## О3 – Образователно съдържание за учители

# О3/А2 – Ръководство за прилагане на подхода CODING4GIRLS, базиран на сериозни игри

Превод на документа от английски език

Цитирайте:

Daniela Tuparova, Boyana Garkova, Joze Rugelj, Mateja Bevcic, Spela Cerar, Matej Zapushek, Michela Tramonti, Alden Meirzhanovich Dochshanov, Carlos V. Carvalho, Rita Durão, Ivanichka Nestorova, Rositsa Georgieva, Hariklia Tsalapata, Olivier Heidmann, Kostas Katsimentes, Christina, Taka Roxani, Sotiri Evangelou, Nadia Vlachoutsou, Nataša Hoić-Božić, Martina Holenko Dlab, Ivona Franković, Marina Ivašić Kos, Luigi Tramonti, , Ahu Şimşek, Kadir Fatih Mutlu, Abdurrahman Saygın, USER GUIDE ON THE CODING4GIRLS SERIOUS GAME APPROACH, Erasmus+ no. 2018-1-SI01-KA201-047013, 2020, https://www.coding4girls.eu/results_03.php

## Информация за документа

**Етап от проекта**: O3/A2 – Ръководство за CODING4GIRLS подход с дидактични игри

**Интелектуален продукт № - Наименование**: O3 – Упътване за учителя

**Водещ екип за този интелектуален продукт:** Югозападен университет „Неофит Рилски” (България)

**Участници:**

**South-West University “Neofit Rilski”, Bulgaria**

Daniela Tuparova, Boyana Garkova, Ivanichka Nestorova, Rositsa Georgieva

**UTH – Greece**

Hariklia Tsalapata, Olivier Heidmann, Kostas Katsimentes, Christina, Taka Roxani, Sotiri Evangelou, Nadia Vlachoutsou

**University of Rijeka – Croatia**

Nataša Hoić-Božić, Martina Holenko Dlab, Ivona Franković, Marina Ivašić Kos

**EU-Track – Italy**

Michela Tramonti, Alden Meirzhanovich Dochshanov and Luigi Tramonti

**Virtual Campus - Portugal**

Carlos V. Carvalho, Rita Durão.

**University of Ljubljana - Slovenia**

Joze Rugelj, Mateja Bevcic, Spela Cerar, Matej Zapushek

**GOI - Turkey**

Ahu Şimşek, Kadir Fatih Mutlu, Abdurrahman Saygın

**Забележка за непоемане на отговорност:**
Този проект е финансиран от програма „Еразъм +“ на Европейския съюз.
Информацията и вижданията, изложени в тази публикация, са на автора (ите) и не отразяват непременно официалното мнение на Европейския съюз. Нито институциите и органите на Европейския съюз, нито каквото и да е лице, действащо от тяхно име, не могат да бъдат държани отговорни за използването на съдържащата се в тях информация.

# СЪДЪРЖАНИЕ

# 1. УВОД

Проектът Coding4Girls е насочен към отвореното и иновативно образование и обучение, вградени в дигиталната ера, като се фокусира върху умения за програмиране, които са много търсени в обществото, задвижвано от технологиите. Освен това способностите да се проектират алгоритми, да се кодира и да се програмира са се превърнали в необходими персонални умения, тъй като са свързани с логическото мислене и изграждане на способност за решаване на проблеми. Като такива, уменията в областта на компютърните науки са силно желани, тъй като те са част от способностите, изисквани от работодателите в движени от иновации сектори, които ще стимулират устойчив икономически растеж през следващите години, водещ до по-голяма заетост и в резултат на това, социална интеграция.

В момента обаче има недостиг на квалифициран персонал в областта на компютърните науки и компаниите се сблъскват с трудности при намирането на служители с необходимия набор от умения в областта на ИКТ. Проектът CODING4GIRLS има за цел да преодолее тази празнина чрез ефективно справяне с уменията по компютърни науки в прогимназиална училищна степен и в първите години на гимназиалната степен, което е времето, когато много ученици губят интерес към компютърните науки. Освен това в тази област има широко призната разлика между половете, тъй като момичетата не се интересуват от кариера в компютърните науки.

CODING4GIRLS е насочен към приобщаващото образование и младежта, като насърчава равни възможности между момичета и момчета в изключително перспективната кариера в областта на компютърните науки. Проектът се справя с тази цел, като се насочва към погрешните възприятия и нагласите на учащите, учителите и родителите относно кариерата в областта на компютърните науки и нейните предимства както за момичета, така и за момчета, като демонстрира връзките си с реалния живот и подчертава факта, че такава кариера се отплаща независимо от пола; но също така и чрез справяне с недостатъчните постижения в уменията по компютърни науки, които са част от по-широката група от компетенции в областта на науката, технологиите, инженерните науки и математиката (STEM).

Проектът помага за изграждането на необходимата основа на учениците, която ще им позволи да успеят в бъдещите си начинания, академични (във висше или друго продължаващо образование в областта на компютърните науки) или професионални (в професионални или професионални дейности, свързани с компютърните науки). Но CODING4GIRLS също се занимава с развитието на учителските компетентности и профила на учителската професия, като дава възможност на преподавателите ефективно да изграждат желани умения за компютърни науки у своите ученици.

Освен това дава възможност на преподавателите да ръководят инициативи, които насърчават перспективите за равенство между половете в преследването на академични или професионални пътища в науката. Това ръководство за учители е част от интелектуалните резултати, разработени в рамките на проекта. Учителите ще намерят информация за: основните понятия, върху които се основава методологията Coding4Girls - обучение, базирано на компютърни игри, дидактични игри, проектно мислене; позиция на момичетата в ИКТ и как възприемат компютърните образователни игри; характеристики на игровата среда на Coding4Girls с два компонента – за учителя и за ученика; преглед на подходите за обучение, основано на игри, за преподаване на програмиране и примери за полезна среда за програмиране. Ръководството включва разработените по проекта сценарии на уроцисе под формата на учебни листове и тяхното адаптиране съгласно методологията Coding4Girls и разработената игрова среда.

.

# 2. ОСНОВНИ ПОНЯТИЯ

## 2.1. Сериозни игри

Играта е широко понятие, което затруднява посочването на най-важните характеристики, които превръщат определена дейност в игра. За Тортън най-важната характеристика на играта е интерактивността. Джонстън вижда играта като дейност, която има правила, на които всички участници трябва да се подчиняват, една или няколко цели, взаимодействие и динамични визуални ефекти (Johnston & de Felix, 1993). Malone (Malone T. , 1981) в своята теория излага фантазията, любопитството, предизвикателството и контрола като най-важните компоненти на всяка игра. Най-изчерпателното разпределение е направено от Garris et al. (Garris , Ahlers, & Driskell, 2002) твърдейки, че тези компоненти са конкуренция, предизвикателство, социални взаимодействия, разговор и фантазия. Теорията на обучението, основано на игри, на Пренски е една от най-влиятелните. Той твърди, че играта се състои от тези елементи: правила, цели и задачи, резултати и обратна връзка, конфликтни ситуации (състезание, предизвикателство, опозиция), взаимодействие и фантастична рамка (Prensky , 2001). Авторите на книгата „Сериозни игри“, определят играта като доброволна дейност (форма на свобода), отделена от реалния живот (въображаем свят, който може да има или няма връзка с реалния живот), която поглъща цялото внимание на играча и се играе според установените правила, които всички играчи трябва да спазват (Michael & Chen, 2006).

Една от най-изложените характеристики на играта е интерактивността и тя се отнася до взаимодействие с компютър, противник (контролиран от компютър или човек) или с други играчи в отбора. Компютърните игри се разделят на броя играчи, едновременно ангажирани в дадена игра, на единични, мултиплейър и масово мултиплейър игри. Според жанра познаваме аркадни, екшън, войни, ролеви игри, стратегически, настолни, спортни, симулационни и приключенски игри. Графиките за популярност на компютърните игри показват, че най-играните игри са масово мултиплейър ролеви игри, стратегии и екшън игри. За да разберем наистина силата на играта, нека първо разгледаме детската игра. Детската игра е известна като една от най-важните дейности,

които помагат да се развият важни умения за целия живот. Чрез игра детето усъвършенства своите интелектуални възможности, като открива първите основни понятия от реалния свят и осъществява жизнеспособни връзки между тях. Жан Пиаже разглежда играта като включване на нови материали в съществуващата когнитивна структура и консолидиране на новоученото поведение. Фройд подчертава емоционалните ползи от играта, като твърди, че тя намалява обективната и инстинктивна тревожност - помага на детето да разреши емоционалните проблеми. Привърженците на социалнит конструктивизъм вярват, че играта е важна поради развитието на социални умения.

Ученето с игра е една от най-често срещаните дейности в ранна възраст, но положителните ефекти от играта и игрите върху ученето често се пренебрегват при официалното образование. Направени са много изследвания в областта на обучението, базирано на игри, което показва, че ако учебният материал е представен във формат на компютърна игра, това има положителни ефекти върху мотивацията. Играта може да помогне на учениците да насочат вниманието към учебния материал и в същото време е източник на забавление, което им доставя удоволствие и удоволствие. Притежаването на тези два компонента в учебния процес ни дава спокоен и силно мотивиран ученик, който следователно е по-готов да учи. Други педагогически предимства са съвместното обучение (мултиплейър възможности), експерименталното и активното обучение, базираното на проблеми обучение и автентичните дейности. Днес използването на ИКТ е често срещано в официалното образование, което ни дава възможност да разработим качествени материали, за да насърчим ученето.

Сериозните игри обикновено се отнасят до игри, използвани за обучение, реклама, симулация или обучение. Те позволяват на учащите да преживяват ситуации, които са невъзможни в реалния свят по различни причини, като безопасност, разходи или време. Предполага се, че те имат определени резултати от обучението и се твърди, че имат положително въздействие върху развитието на новите знания или различни умения на играчите. За да проектираме добра сериозна игра, трябва да можем да проектираме и произведем добър софтуер. Но сериозните игри са много повече от софтуер. Също така трябва да можем да проектираме и да създадем добри инструкции. Потенциалната

стойност на сериозните игри за учене изглежда висока, но все още остава съпротива срещу използването на игри в класната стая. Разумен начин да убедите повече учители да опитат игри е чрез педагогика, свързваща елементи на съществуващите дизайни на игри с приети теории за учене и преподаване.

Rieber, Smith and Noah (Rieber , Smith , & Noah, 1998) заявяват още през 1998 г., че има две различни приложения на игрите в образованието: игра на игри и проектиране на игри. Играта е традиционният подход, при който човек предоставя готови игри на учениците. Проектирането на игри предполага, че самият акт на изграждане на игра е път към учене, независимо дали играта се оказва интересна или не за други хора. Идеята за „учене чрез проектиране“ се основава на предположението, че активното участие в процеса на проектиране и разработване е най-добрият начин да научите нещо. Този подход придоби все по-голямо значение поради разпространението на компютърно базирани инструменти за проектиране и създаване.

## 2.2. Обучение, базирано на играта

Има няколко причини, които привличат вниманието на педагога към игрите s (Gee, 2007). Във формалното образование ние преживяваме преминаване от традиционния дидактически модел, който е фокусиран върху обучението, към ориентиран към учащия модел, който подчертава ролята на активния обучаем. Също така се промени възгледа за целите на обучението от по-ниски таксономични нива, като просто припомняне на информация, на по-високи нива, като намиране и използване на информация в нови ситуации (Rugelj, 2016).

Prensky (Prensky , 2001), Gee (Gee, 2003), and Whitton (Whitton, 2009) определят обучението, базирано на игри, като процес на обучение с използване на дигитални игри. Игрите могат да осигурят мотивация за учене, като по този начин увеличават шанса да бъдат постигнати желаните резултати от обучението. Обучението се дефинира като придобиване на знания или умения чрез опит или практика и какъв по-добър начин за учене, отколкото чрез игра (Pivec & Kearney, 2007), (Pivec, Koubek, & Dondi, 2004) Почти всички проучвания за обучение, основано на игри, показват, че учениците са силно мотивирани, когато учебните материали се представят във формат на компютърна игра и че това има положителни ефекти върху ускоряването на учебния процес (Zapušek &

Rugelj, 2013). Студентите се нуждаят от мотивация, за да се съсредоточат върху това, което трябва да се научи, но за да се получи качествено обучение това не е достатъчно. Сравняването на резултатите от обучението на ученици, които са се учили с компютърни игри, и тези, които учат с друг тип учебни материали, показва, че няма съществена разлика между тях. Това обикновено се дължи на неподходящ дизайн на играта. Игрите могат да бъдат много привлекателни за учениците, но ако те забавляват, а не преподават, използването на игри в образованието няма много смисъл. И така, трябва да разберем кои са елементите, които правят компютърната игра образователна компютърна игра.

Gross (Gross, 2003) твърди, че дигиталните игри с образователна цел трябва да имат добре дефинирани учебни цели и да насърчават развитието на важни стратегии и умения за увеличаване на когнитивните и интелектуални способности на учащите.

Според Malone (Malone T. W., 1981b) and Garris et al. (Garris , Ahlers, & Driskell, 2002), елементите, допринасящи за образователните ценности на цифровите игри, са чувствени стимули (визуални и звукови изображения на учебния материал), фантазия (контекст, представен във въображаема обстановка), предизвикателство (взискателна или стимулираща ситуация) и любопитство (желание да се знае или да се научи ). Тези елементи трябва да бъдат включени в интегрирана платформа, за да структурират цели и правила, контекст на смислено обучение, привлекателна история, незабавна обратна връзка, високо ниво на интерактивност, предизвикателство и конкуренция, случайни елементи на изненада и богата среда за учене.

Играта може да бъде създадена за учене, тъй като включва умствена (а понякога и физическа) стимулация и развива практически умения - тя принуждава играча да решава, да избира, да дефинира приоритети, да решава проблеми и др. Непосредствената награда (и обратна връзка) е основен мотивационен фактор, независимо дали се превежда като игрови субекти (повече жизнена сила, достъп до нови нива и т.н.) или като неврологични импулси (щастие, чувство за постижение и т.н.). Игрите могат да бъдат социална среда, като понякога включват големи разпределени общности. Те предполагат (Baptista & Vaz de Carvalho, 2010) способности за самообучение (от играчите често се изисква да търсят информация, за да овладеят

самата игра), позволяват трансфер на учене от други реалности и по своята същност имат опит с участието на множество сетива.

Van Eck (van Eck, 2006) твърди, че игрите и играта могат да бъдат ефективна учебна среда не защото са забавни, а защото са поглъщащи, изискват от играча да взема чести важни решения, да има интелигентни цели, да се адаптира към всеки играч поотделно и да включва социална мрежа.

Ако разгледаме модел на базирано на игри обучение от Garris, Ahlers, and Driskell (Garris , Ahlers, & Driskell, 2002), основната характеристика на образователната игра е, че учебното съдържание е размито в рамките на характеристиките на играта. Учениците играят играта и се забавляват, забравяйки за „учебната“ част от преживяването, въпреки че постоянно им се представят нови концепции, които трябва да адаптират, за да имат успех в играта. Трябва да стимулираме мотивацията с дизайна на играта, насърчавайки повтарянето на циклите в контекста на играта. Очаква се играчът да предизвика желано поведение въз основа на емоционални и когнитивни реакции, които са резултат от взаимодействие и обратна връзка от играта.

Gee (Gee, 2003) твърди, че характеристиките на видеоигрите с висок учебен потенциал се разделят на две категории: „неигрови“ функции, които могат да се появят и в неигрови контексти, и „игрови“ характеристики, които се отнасят повече до „игровите“ функции на игрите. Въпреки това разграничение, не бива да се предполага, че „неигровите“ функции биха работили и за обучение, ако са откъснати от „играта“.

„Неигровите“ функции на игрите с висок потенциал за обучение са:

- Емпатия към сложни системи - да разгледаме сложната система отвътре, за да разберем как нейните променливи си взаимодействат.
- Симулации на опит и подготовка за действие. Във видеоигрите играчите виждат виртуалния свят от гледна точка на това как той предлага видовете действия, които трябва да предприемат, за да постигнат целта да спечелят.
- Разпределена интелигентност чрез създаването на интелигентен инструмент Добрите видео игри разпространяват интелигентност между човек от реалния свят и изкуствено интелигентни виртуални герои, за да представят макро и микро изглед на ситуацията.

- Междуфункционална работа в екип.

  Добрите видео игри могат да научат на сътрудничество и междуфункционална работа в екип за институции като училища и работни места. Играчите взаимодействат помежду си не от гледна точка на техните реални характеристики, а чрез техните функционални игрови идентичности. Те могат също така да изберат да използват своята реална раса, класа, култура и пол като стратегически ресурси, но не са принудени да бъдат предварително зададени расови, полови, културни или класови категории.

- Ситуативно значение

  Диалогът и опитът са от съществено значение, за да могат хората да свързват думите с действителните действия, функции и решаване на проблеми. Тъй като видеоигрите са симулация на опит, те могат да разположат езика в специфичен контекст.

- Отвореност: смесване на личното и социалното
- В добрите игри с отворен край играчите конструират свои собствени цели, които се основават на собствените им желания, стилове и произход. Комбинацията от лични и в игрите цели създава състояние на висока мотивация.

Характеристиките на „добра игра“, свързана с ефективното обучение, са следните:

Мотивация

Видеоигрите са дълбоко мотивиращи за играчите и е важно да се разберат източниците на тази мотивация, ако искат да осигурят основа за учене.

• Ролята на провала

Цената на неуспеха е намалена и често се разглежда като начин да научите основния модел. Тези характеристики на неуспех в игрите позволяват на играчите да поемат рискове, които може да са твърде скъпи.

• Състезание и сътрудничество

Много геймъри, включително младите, се радват на конкуренция с други играчи в игрите, но може да не видят състезанието като приятно и мотивиращо в училище. Конкуренцията във видеоигрите се възприема от геймърите като социална, колкото за игрите, толкова и за печелене и загуба.

• Дизайнът на игрите, който се отнася до:

• Интерактивност - играчът не просто пасивно консумира знания, но има контрол върху съдържанието;

• Персонализация - базирана на стилове на обучение и предоставяща множество пътища за успех;

• Силни идентичности - добрите игри предлагат на играчите идентичности, които предизвикват дълбока инвестиция от страна на играча и които са ясно свързани с функциите, уменията и целите, които човек трябва да постигне във виртуалния свят;

• Добре секвенирани проблеми - при свързващите подходи към обучението се твърди, че последователността е от решаващо значение за ефективното обучение в сложни области;

• Приятно ниво на разочарование - коригиране на предизвикателствата по такъв начин, че редица играчи да могат да преживеят играта като предизвикателна, но изпълнима;

• Цикъл на експертиза - повтарящи се цикли на разширена практика и тестове за майсторство;

• „Дълбоко“ и „честно“ - играта трябва да бъде предизвикателна, но настроена по начин, който води до успех.

Първоначално елементите на игра трябва да бъдат лесни и лесни за научаване и да стават по-сложни, колкото повече играчът идва да ги овладее.

Компютърните игри, използвани в образователни условия, доказано увеличават мотивацията. Но високо мотивираните обучаеми не са достатъчни, за да се получи качествено обучение. Също така се нуждаем от добри учебни материали, така че учащите действително да получат нови знания от материали, представени под формата на компютърна игра.

Въпреки положителното влияние на играта върху мотивацията, учителите все още се колебаят да използват сериозни игри в преподаването. Както те посочиха в нашето разследване, игрите могат да отнемат много време за използване в класната стая. Поради това те често използват сериозни учебни игри като домашна учебна дейност или като мотивация в уводните уроци.

## 2.3. Дизайн мислене (Design Thinking)

Идеите за дизайн се появяват в края на 60-те години, но като концепция „дизайн мислене“ се появява в Станфордския университет в края на 80-те и 90-те години. В света на бизнеса методът става широко разпространен, след като е популяризиран от Тим Браун от IDEO, който посочва: „Дизайнерското мислене е ориентиран към човека подход към иновациите, който черпи от инструментите на дизайнера за интегриране на нуждите на хората, технологичните възможности и изискванията за успех в бизнеса.“ (IDEO Design thinking, 2008). В началото на 21 век дизайнерското мислене стана изключително популярно и ИТ компаниите започнаха да прилагат подхода към разработването на своите продукти. От 2005 г. например SAP прилага дизайн мисленето като философия за решаване на проблеми и като иновативен подход, насочен към крайния потребител "(Innovation Starter, 2015), и в резултат на това разработването на нови продукти води до желания резултат. Компании като P&G, IBM Design, GOOGLE, AMAZON и Cisco интегрират дизайн мисленето в целия си бизнес и организация. (Naiman, 2019)

Концепцията, разработена от Станфордския университет, определя дизайнерското мислене като „водеща методология за творческо решаване на проблеми и създаване на иновации и се използва ежедневно от хиляди компании и организации по целия свят. Целта е да се развият у подрастващите уменията на 21 век - работа в екип, решаване на проблеми, креативност, съпричастност, увереност, търпение, концентрация, експериментиране. " (Karakehayova, 2019).

Моделът „Станфорд” се състои от 5 етапа (Fig. 2.1.)

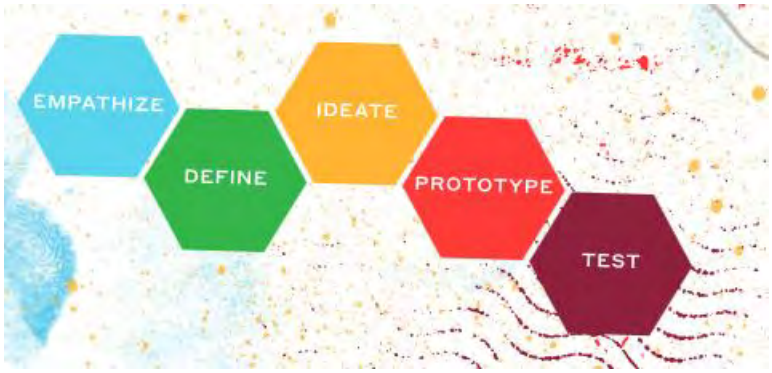



*Fig. 2.1. Етапи в модела на проектно мислене „Станфорд” (Doorley, Holcomb, Klebahn, Segovia, & Utley, 2018)*

Основните етапи и характеристики на дизайн мисленето могат да бъдат обобщени, както следва (Таблица 2.1.) *(Georgieva & Tuparova, 2020)*

*Таблица2.1. Етапи и характеристики на дизайн мисленето, базирани на (Innovation Starter, 2015), (Naiman, 2019), (A project of the K-12 Initiative at Stanford University, 2009)*

| **Практики** | **Принципи** | **Етапи** | | |
|---|---|---|---|---|
| | | **Кратък процес** | **Удължен процес** | **dSchool Станфорд** |
| Съчувствие и задълбочено разбиране на нуждите на потребителя | Откритие - посрещане на ново предизвикателство. Как да го решим? | Наблюдение и изучаване на поведението и опита на потребителите, изследване и дефиниране на проблем и гледна точка чрез съпричастност. | | Проявяване на емпатия |
| Формиране на разнородни екипи | Интерпретация - Какво научих и как да го интерпретирам? | Генерирайте много идеи за кратък период от време. | | Дефиниране |
| Комуникация, базирана на диалога | Идея - виждам възможност, как да я оформя в идея? | Подбор и класификация на идеите | | Оформяне на идея |
| Генериране на решения чрез експериментиране; | Експериментиране - имам идея как да го изградя? | Създаване на прототип - продукт, който получава бърза обратна връзка от потребителите. | | Създаване на прототип |
| Използване на структуриран и улеснен процес. | Еволюция - опитах нещо ново, как да го разработя? | Тестване / обратна връзка - как да подобрим прототипа, ако е необходимо | | Тестване |
| | | | Проверка дали работи идеята | |
| | | | Обучение и усъвършенстване въз основа на обратна връзка | |

Дизайн мисленето е от полза не само за бизнеса, но и за неправителствения, образователния и здравния сектор. Все повече образователни институции се насочват

към дизайнерското мислене. Постепенно добрите практики за прилагане на дизайнерското мислене в образованието се увеличават (Georgieva & Tuparova, 2020) . При прилагането на подход на дизайн мислене, учителят трябва да направи връзка между отделните етапи в модела на дизайн мисленето и подходящите методи на преподаване, които могат да бъдат приложени. Дизайн мисленето изисква прилагането на широк спектър от методи на обучение.


*Фигура 2.2. Методи на преподаване и проектно мислене (Georgieva & Tuparova, 2020)*

## 2.4. Теории з аученето и игрово базирано обучение

Много образователни компютърни игри са проектирани според поведенческата теория на ученето в миналото (Rugelj & Lapina, 2019). Те бяха изпълнени под формата на програмирано обучение. На обучаващите се беше предложен стимул под формата на въпрос или друг вид задача или проблем, които трябва да бъдат решени. Учащите реагират незабавно, като изберат един от предлаганите отговори. Ако отговорът е верен, играта предоставя някакъв положителен отговор под формата на положителна реакция на характера или щастлива мелодия, която стимулира положителните емоции. Тази двойка „действие-реакция“ налага връзка между въпрос и правилния отговор. В

случай на грешен отговор се осигурява реакция под формата на отрицателни стимули и връзката се отслабва. Игрите и викторините с насочване и щракване имат вградена концепция за тренировка и практика и са типични представители на игрите, базирани на бихейвиоризма. Те са подходящи за изучаване на основни аритметични операции или за подпомагане на запаметяването на факти, т.е. постигат се цели за обучение на най-ниските нива на таксономията на Блум..

Теорията за когнитивното учене набляга на когнитивната дейност на обучаемия и формирането на подходящи умствени модели. Обучаващите се учат основни понятия от нейния учител или формират учебни ресурси и след това използват логическа дедукция, за да получат нови знания. Пъзели и стратегически игри като среда за вземане на решения примери за когнитивисткия подход. Най-напредналите форми на когнитивна теория базирани игри се основават на интелигентни системи за обучение, където се използват алгоритми за машинно обучение за моделиране на знания на ученици и експерти, за да се предоставят персонализирани учебни материали..

Конструктивизмът е алтернативен възглед, предполагащ, че обучаемите сами конструират своите знания; редица индивидуално конструирани представления на знанията, всички еднакво валидни. Обучението е активен процес на конструиране (по-скоро придобиване на знания), изграден рекурсивно върху знания, които потребителят вече има. В процес на изграждане сензорните данни се комбинират със съществуващите знания, за да се създадат нови жизнеспособни умствени модели, които от своя страна са основата за по-нататъшно изграждане.

Конструктивисткото обучение набляга на откриването и проучването, като се аргументира, че учениците трябва да бъдат поставени в среда (която може да бъде моделирана с компютърна игра), където те изграждат свои собствени знания. Три основни принципа определят конструктивисткия възглед за ученето:

- всеки човек формира свое собствено представяне на знанието;
- ученето се случва, когато изследването на обучаемите разкрие несъответствие между настоящото им представяне на знания и техния опит;
- обучението се осъществява в социален контекст и взаимодействието между учащите и техните връстници е необходима част от учебния процес.

Учебните материали осигуряват инструкции, които се състоят от подпомагане на изграждането на знания, вместо от деклариране на знанията по поведенчески начин. Симулациите на компютърни игри възпроизвеждат различни сценарии от реалния живот във формат на компютърна игра. Те представят модел на абстрахирана реалност, в който обучаемият изпълнява определена роля. Задачата на учителя е да осигури насоки и обратна връзка, когато ученикът учи, т.е.изграждане на жизнеспособни умствени модели.

Игрите могат да доведат до промени в нагласите, поведението и уменията на играча.. Shute and Ke (Shute & Ke, Games, Learning, and Assessment, 2012) установяват, че има сближаване между основните елементи на добрата игра и характеристиките на продуктивното обучение. Дизайнът на игрите може много да ни научи за ученето, а съвременната теория на обучението може да ни научи за проектирането на по-добри игри (Shute, Rieber, & Van Eck, 2012). Marshall McLuhan, канадски философ на комуникационната теория, който предвиждаше World Wide Web през шейсетте години на миналия век, когато говореше за глобалното село, заяви: „Всеки, който прави разлика между игрите и ученето, не знае първото нещо и за двете.” (Shute, Rieber, & Van Eck, 2012)

Whitton and Moseley (Whitton & Moseley, 2012) предлагат рамка за добри практики в дизайна на сериозни игри от активна учебна гледна точка. Според неговите насоки средата на играта трябва да подпомага активното учене, като насърчава изследването, решаването на проблеми и проучването, поражда ангажираност с изрични и постижими цели, да бъде подходяща за учебния контекст, да подкрепя и предоставя възможности за размисъл, да предоставя равни възможности за всички ученици , осигуряват постоянна подкрепа с постепенно въвеждане на нарастваща сложност и ще бъдат подкрепяни с някаква помощ или съвети.

Няколко проучвания показват, че сериозните игри могат да подкрепят ученето с мотивация, ангажираност и забавление (Hijon-Neira, Velazquez-Iturbide, Pizarro-Romero, & Carrico, 2015). Ученето по програмиране изисква много компетенции като логическо мислене, решаване на проблеми и способност за разбиране на абстрактни понятия. Поради тази причина много ученици смятат, че компютърното програмиране е трудно

за научаване. Този факт може да доведе до ниска мотивация за изучаване на встъпителни курсове по програмиране. За да се подобри мотивацията и да се подобри отношението на учениците към ученето към програмирането, учителите търсят стимулиращи подходи към ученето.

Програмирането се научава най-добре чрез практика и, ако учениците искат да учат ефективно, поне част от тази практика ще трябва да бъде самостоятелно насочена или в сътрудничество с връстници. Ключовата роля на учителя е да убеди учениците да направят това и по този начин да ги мотивира (Feldgen & Clua, 2004).

Студенти в уводния курс за компютърно програмиране проектират и разработват програми (Rugelj & Lapina, 2019). Това е типичен подход за активно учене. Ако въведем работа по проекти в групи, която се оказа много ефективен начин за учене на програмиране (Nančovska Šerbec, Kaučič, & Rugelj, 2008), всъщност можем да говорим за триалогичния принцип на обучение. Триалогичният план за обучение се състои от форми на обучение, при които учениците съвместно разработват, променят или създават общ артефакт (т.е. компютърна програма в нашия случай) в систематичен процес (Kafai, 1995). Той се фокусира върху взаимодействието, което се случва при създаването на конкретни артефакти, не само между хората („диалогичен подход“), или в съзнанието на човек („монологичен“ подход) (Paavola & Hakkarainen, 2005).

# 3. МОМИЧЕТАТА, ИКТ И СЕРИОЗНИТЕ ИГРИ

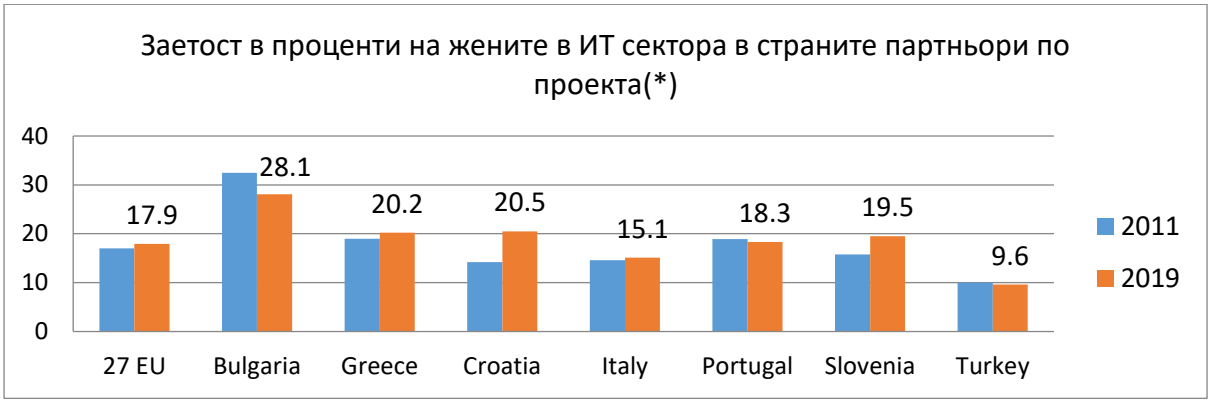
## 3.1. Позициите на жените в ИКТ сектора – актуално състояние

Дигиталното общество поставя нарастващи изисквания към компетенциите на съвременните хора. Нараства необходимостта от хора, които са не само цифрово грамотни, но и притежават уменията да създават и поддържат софтуерни приложения в различни области на човешката дейност. Медиите постоянно коментират недостига на квалифицирани специалисти в сектора на информационните и комуникационни технологии (ИКТ). Наблюдава се непрекъснат растеж на този сектор.

През 2019 г. около 7,8 милиона души са работили като ИКТ специалисти в целия Европейски съюз (ЕС). (Eurostat, 2020). Дигиталното общество поставя нарастващи изисквания към компетенциите на съвременните хора. Нараства необходимостта от хора, които са не само цифрово грамотни, но и притежават уменията да създават и поддържат софтуерни приложения в различни области на човешката дейност. Медиите постоянно коментират недостига на квалифицирани специалисти в сектора на информационните и комуникационни технологии (ИКТ). Наблюдава се непрекъснат растеж на този сектор.

Според (Eurostat, 2020) в ЕС-27 през 2011 г. жените в ИТ сектора са били 17%, а през 2019 г. - около 17,9%. Според данни на Евростат за периода 2007-2019 г. средно за 27-те държави-членки на ЕС жените в ИТ сектора са намалели от 22,5% на 17,9%. През 2019 г. жените са 17,3% от общия брой на заетите в ЕС лица с ИКТ образование, в сравнение с 20,2% през 2009 г. (Eurostat, 2020).

Настоящото състояние на заетостта на жените в сектора на ИКТ в страните партньори по проекта Coding4girls е представено на фиг. 3.1




Фигура 3.1. Наети жени в сектор ИКТ (в проценти) през 2011 и 2019 г.

## 3.2. Отношение на момичетата към сериозните игри

Резултатите от проучване, проведено от Vermeulen et al. (Vermeulen, Van Looy, Courtois, & De Grove, 2011) показа, че разликите между половете присъстват постоянно, но предишният опит значително влияе на констатациите (Rugelj & Lapina, 2019). ). Жените играят игри по-често, но по-кратко време от мъжете. Те предпочитат абстрактни, кратки и лесни за овладяване игри като ежедневни (например тетрис) и игри в социални мрежи, докато мъжете са по-склонни да играят „основни“ жанрове. Тази категория се отнася до игри, базирани на умения, които отнемат много време и обикновено се отличават с висококачествени триизмерни графики като стрелци, бой, екшън-приключения, спорт, състезания, стратегии, ролеви игри и MMO игри. Неядрените жанрове включват платформа, приключения, симулация, парти, сериозни, класически и ежедневни игри. Основните жанрове се играят повече от мъже, отколкото от жени. Проучването установи, че жените са любители на пъзел игри, докато състезания, ритъм, симулация и виртуални игри се играят както от жени, така и от мъже. Друго проучване разкрива, че жените предпочитат парти игри (като музикални и танцови игри) и класически „ретро“ игри.

Alserri et al (Alserri, Zin, & Wook, 2018) са направили обширно проучване на свързани статии за ефективни елементи на сериозната игра, като мотивационни елементи на играчите на дигитални игри, ефективни елементи на образователна игра и елементи на женските предпочитания. Те използват резултатите, за да създадат концептуален модел за ангажираност, основана на пола в сериозните игри. Този метод използваме в проекта Coding4Girls, за да увеличим мотивацията на момичетата за програмиране чрез

подходящ подбор на игри, които ще бъдат проектирани и внедрени от учениците в обучение, базирано на създаване на игри.

Авторите заявяват, че изследването разкрива, че мотивацията за игра на специфичен тип дигитална игра зависи от пола. Стереотипът, че жените не играят компютърни игри, вече не е валиден. Разликите в предпочитанията на пола за цифровите игри, установени в това изследване, са много сходни с вече представените разлики, идентифицирани от Vermeulen et al. (Vermeulen, Van Looy, Courtois, & De Grove, 2011). Жените предпочитат да играят изследователски и творчески повече от мъжете. Те играят по-често, но за по-малко време от мъжете, а предпочитанията им към жанровете на играта също са различни. Установено е, че мъжете прекарват повече време в компютърни игри, отколкото жените. Жените също предпочитат пъзел игри, социални игри с награди, предлагани в игрите, образователни игри, жанрове на симулационни игри, както и съвместни игри и игри за изследване, виртуален живот, виртуален свят и парти игри. Освен това, жените обичат да играят приключенски игри, но те предпочитат първо да наблюдават другите, преди да играят себе си. Елементите на мотивационните предпочитания в игрите за жени са предизвикателство, бягство, забавление, социално взаимодействие, мотивация, фантазия, състезание и възбуда. И мъже, и жени като състезания, симулации и виртуални игри.

Phan et al. (Phan, Jardina, Hoyle, & Chaparro, 2012) стигнаха до много сходни заключения в своето проучване относно различията между половете между мъжете и жените геймъри по отношение на използването, предпочитанията и поведението на компютърните игри.

Според изследването на Тупарова, Тупаров, Велева (Tuparova, Tuparov, & Veleva, 2019 b), има разлики в предпочитанията на момчетата и момичетата към характеристиките и елементите на игрите като: Графичен дизайн; Звук; Възможност за игра на играта на различни устройства: компютър, таблет, смарт телефон и др .; Възможност за включване на много играчи; Възможност за онлайн игра; Възможност за преминаване към по-сложни нива на играта. Най-предпочитаната характеристика на игрите за момичета е възможността за преминаване към по-сложни нива на играта. За момчетата най-предпочитано бъдеще е логиката и сюжетът / сценария на играта. Функцията с най-

нисък интерес за момичета е звукът в игрите, докато за момчетата - награди в играта. Що се отнася до крайно използваното устройство, смартфонът е най-използваното игрално устройство както от момчета, така и от момичета. За момичетата второто в употреба устройство е лаптопът, последван от таблет. За момчета вторият в употреба са настолен компютър и лаптоп. Авторите отбелязват, че момчетата показват по-голямо предпочитание от момичетата към сюжетни игри и момичетата имат по-голямо предпочитание от момчетата към игрите с памет. Независими от пола са предпочитания за следните видове игри: игри със социални елементи; внимание / концентрация, игри за бързина на реакциите, ролеви игри, пъзели, приключенски игри. Типът игра с най-нисък интерес както за момичета, така и за момчета е пъзелът.

Трябва да вземем предвид конкретната ситуация в проекта Coding4Girls, който всъщност се основава на обучение, основано на създаване на игри, и да подготвим задачи, които са интересни и мотивиращи за момичетата. Програмна среда, която представя програмирането като средство за създаване на анимационни филми (т.е. разказване на истории) или прости игри, може да бъде подходяща за момичета, тъй като повечето от тях могат да измислят идея за история или проста игра, която биха искали да създадат (Wolz, Barnes, & Bayliss, 2009). И двете са естествено последователни и е малко вероятно да изискват незабавни разширени програмни концепции, те са форма на себеизразяване и предоставят на момичетата възможност да експериментират с различни роли, а приятелите, които не са програмисти, могат лесно да разберат и оценят анимирана история или игра, която предоставят възможност за положителна обратна връзка. Kelleher, Pausch и Kiesler (Kelleher , Pausch, & Kiesler, 2007) използват средите на Storytelling Alice и Generic Alice съответно и за двата случая. Те твърдят, че няколко проучвания на деца програмисти са установили, че когато момичетата и момчетата имат подобен опит с компютърното програмиране, те са еднакво заинтересовани и ефективни в ученето да програмират. Но ефективността на програмирането е свързана с количеството време, прекарано от потребителите в програмирането, и предишния им опит в програмирането.

# 4. 1. ОБУЧЕНИЕ ПО ПРОГРАМИРАНЕ ЧРЕЗ ИГРИ

Този раздел ще бъде фокусиран върху представянето и сравняването на различни подходи и платформи / игрови среди, които могат да се използват за преподаване на умения за програмиране на деца.

## 4.1. Подходи за обучение на програмиране чрез игри

В момента има голямо разнообразие от подходи за обучение на програмиране за начинаещи програмисти.

Подходите могат да бъдат класифицирани като:

- Изучаване на програмиране чрез проектиране на игри - познато още като обучение, основано на игрови дизайн, съответства на използването на езици за програмиране, за да се научите как да кодирате чрез проектиране и създаване на игри;
- Учене на програмиране в компютърно базирана игрова среда.
- Изучаване на програмиране чрез проектиране на игри в среда, базирана на игри. Методологията Coding4Girls принадлежи към тази категория учебни среди за програмиране.

Един пример е използването на блоково-базирани езици за визуално програмиране, които са особено удобни за потребителя, лесни за работа (на базата на плъзгане и пускане), предотвратяват грешки по отношение на синтаксиса и логиката (Ouahbi, Kaddari, Darhmaoui, Elachqar, & Lahmine, 2015) представят съдържание в много интуитивна среда. Scratch (Resnick, et al., 2009), https://scratch.mit.edu/ , Snap! (Weintrop & Wilensky, 2015) https://snap.berkeley.edu/ , or Alice 2/3 https://www.alice.org/about/ са добре познати и успешни примери за езици за визуално програмиране. В следващата са сравнени характеристиките на най-известните езици и среди за визуално или блоково програмиране (и среди)..

Таблица 4.1. Характеристики на някои блоково-базирани среди за програмиране

| | ПЛАТФОРМА | ЦЕЛЕВА ГРУПА | ЕЗИЦИ | БЕЗПЛАТНИ/ПЛАТЕНИ |
|---|---|---|---|---|
| Scratch | Уеб-базирана (но с офлайн версия) | 8-16 | Визуален език за компютърно програмиране | Безплатна |
| Snap! | Уеб-базирана (но с офлайн версия) | 12-20 | Визуален език за компютърно програмиране | Безплатна |
| Alice | Windows, Mac OS X, Linux | 12-20 | Визуален език за компютърно програмиране | Безплатна |
| Tynker | Web-based, iOS | 7+ | Визуален език за компютърно програмиране, JavaScript, Python | Платена |

Компютърните игри могат да се използват и за научаване как да се кодира, или чрез насърчаване на учениците да разработват и създават свои собствени видео игри (обучение чрез игрови дизайн) или като им позволяват да играят сериозни игри, чиито учебни резултати обхващат учебни резултати, свързани с програмирането (учене, базирано на игри). Компютърните игри, предназначени за образователни дейности, имат потенциала да създадат мотивираща и забавна учебна среда, тъй като съдържат дейности, които отговарят на образователните стандарти, учебни цели, осигуряват обратна връзка и могат да постигнат високи образователни резултати. (Georgieva & Tuparova, 2019).

Идеята за използването на игри за преподаване на програмиране идва от факта, че те, като са по-ангажиращи и мотивиращи, позволяват на учениците да се научат на изчислително мислене и умения за програмиране в забавна и позната среда, преди да прехвърлят тези умения в изучаването на език за програмиране (Kazimoglu, Kiernan, Bacon, & Mackinnon, 2012), 2012). Все още според (Kazimoglu, Kiernan, Bacon, & Mackinnon, 2012), сериозните игри за учене на програмиране и изчислително мислене трябва да бъдат създадени не само за забавление, но също така трябва да обмислят учебна програма и умения, като правят разлика между различните програмни

конструкции и насърчават добрите програми за програмиране (чрез начини на игра, като постигане на висока оценка).

Сериозните игри също могат да бъдат полезни за програмиране, като превръщат неприятните операции в интересни преживявания —Shabanah et al (Shabanah, Chen, Wechsler, Carr, & Wegman, 2010) създават Algorithm Game Designer целящ да улесни създаването на алгоритмични игри, с цел подобряване на ангажираността в системите за визуализация на алгоритми. Grivokostopoulou et al (Grivokostopoulou, Perikos, & Hatzilygeroudis, 2016) разработиха игра за обучение по алгоритми в областта на изкуствения интелект, базирана на играта Pacman, така че чрез визуализации на алгоритми и анимации учениците могат да бъдат подпомогнати в процеса на обучение, като демонстрират начина, по който алгоритъмът работи, неговите функции и как да се вземат правилни решения въз основа на конкретни параметри.

С развиващото се широко разпространение на по-бързите скорости на интернет връзка и нарастващата наличност на компютри в домовете на всички, започват да се появяват платформите, базирани на онлайн игри като (https://codecombat.com/) и LightBot (https://lightbot.com/). Combéfis et al. (Combéfis, Beresnevičius, & Dagiene, 2016) правят преглед на основните видове онлайн платформи, използвани за преподаване на умения за програмиране, и стигнаха до извода, че следните фактори могат да направят една сериозна игра по-мотивираща и успешна: 1) процес на обратна връзка и оценка; 2) естетика; 3) аспекти на сътрудничество и мултиплейър; 4) насочване през играта; 5) няма негативни последици; 6) музика; 7) предизвикателство на средно ниво, за да се запази интереса на играчите.

Що се отнася до резултатите от обучението на сериозни игри за програмиране на обучение, според обширен преглед на 49 сериозни програми за игри от Miljanovic et al (Miljanovic & Bradbury, 2018), най-сериозните игри за програмиране се фокусират върху решаването на проблеми и основни концепции за програмиране, като малко игри се фокусират върху структурите на данни, методите за разработка и софтуерния дизайн.

Има и игри, които, макар и да не преподават умения за програмиране, са способни да преподават по косвен, но ефективен начин ключови умения за изчислително мислене, като абстракция (напр. Тетрис), декомпозиция (напр. Судоку), мислене по

алгоритъм (напр. Пъзел) игри), оценка (напр. Игри с памет) и обобщение (напр. Pattern Games) (Frankovic, Hoic-Bozic , & Nacinovic-Prskalo, 2018).

При обучението, базирано на създаван на игри, често имаме комбинацията от използването на езици за визуално програмиране и базирани на блокове среди в процеса на проектиране и кодиране на игри. В проучване, което сравнява използването на среди за програмиране Scratch и Pascal (процедурен език чрез команден ред или текстов редактор) с начинаещи програмисти, е установено, че обучаемите са много по-мотивирани да продължат обучението си по компютърни науки, когато използват Scratch (Ouahbi, Kaddari, Darhmaoui, Elachqar, & Lahmine, 2015). ). Изследването също така установява, че създаването на игри и разказването на истории прави учениците по-креативни и автономни, докато учат програмиране. Друго проучване, проведено от Weintrop et al (Weintrop & Wilensky, 2015) , сравнявайки използването на Snap! и Java от ученици в гимназията установяват, че подходът, базиран на блокове, има по-лесна крива на обучение от Java - по този начин е в съответствие с виждането, че базираното на блокове програмиране (като Scratch и Snap!) е по-достъпно за начинаещи програмисти. Според резултатите от проучването това се дължи на факта, че блоковете са по-лесни за четене, поради визуалния характер на блоковете, те са по-лесни за композиране и служат като помощни средства за паметта.

Интегрираният подход за преподаване на програмиране и влиянието му върху постиженията на 14-годишни ученици е разгледан в (Tuparova, Nikolova, & Tuparova, 2020). Този подход съчетава два етапа::

- „Използване на съществуващи образователни компютърни игри: за придобиване на знания и умения, мотивация на учебни дейности, развитие на алгоритмично мислене; за експериментиране със съществуващи образователни компютърни игри за проучване на правилата на играта, елементите на интерфейса и техните свойства и събития;
- Проектиране и разработване на образователни компютърни игри, включително математически модел на играта, дизайн на графичен потребителски интерфейс (GUI); кодиране, тестване и проверка."

За изпълнение на подхода се използва C # среда за програмиране. Моделът е тестван със 126 студенти със специализация по информатика, разделени в две групи - Експериментална и Контролна група. Постиженията на учениците се измерват чрез практическа задача и тест на хартия. Резултатите показват, че няма разлика между постиженията на момчетата и момичетата в експериментална група. Но авторите установяват, че момичетата в експериментална група постигат по-високи резултати от момичетата в контролната група.

## 4.2. Игрово-базирани ззреди за обучение по програмиране

Представяме няколко примера за използването на различни среди, които използват игри за обучение на програмиране и програмиране.

### 4.2.1. Обучение чрез създаване на игри

### Scratch

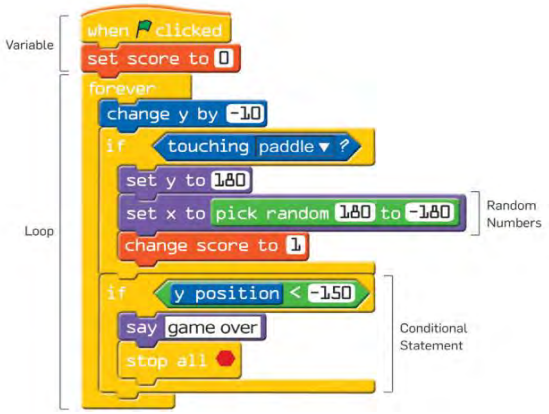



*Fig.4.1* Sample Scratch Script

Scratch (https://scratch.mit.edu/) е базирана на блокове среда за програмиране, т.е. позволява на обучаващите се да сглобяват програми, като сглобяват кодови блокове и получават визуална обратна връзка (Weintrop & Wilensky, 2015). По този начин обучаемите се запознават с основите на програмирането, без да се притесняват за синтаксиса (Ouahbi, Kaddari, Darhmaoui, Elachqar, & Lahmine, 2015).

Scratch може да се използва като въведение в програмирането, особено сред по-малките ученици, но и като начин за улесняване на прехода към други програмни среди, като по този начин въвежда и насърчава интереса към компютърните науки (Wolz, Maloney, & Pulimood, 2008). Според изследване, проведено от Meerbaum-Salant et al.

(2013), повечето ученици могат да постигнат разумно ниво на усвояване на концепции за компютърни науки чрез Scratch. Трудности обаче възникват при преподаване на конкретни теми, които изискват по-голямо ниво на абстракция. Това обаче може да бъде решено, ако учениците са придружени в процеса от учителя.

Уебсайтът на Scratch поддържа световна мрежа от потребители, хостваща общност от програмисти, която позволява на потребителите да споделят проекти (Meerbaum-Salant , Armoni, & Ben-Ari, 2013). Scratch се предлага на повече от 50 езика, включително български, хърватски, английски, гръцки, италиански, португалски, словенски и турски. Той също така има офлайн редактор, който позволява програмата да бъде изтеглена на компютър и да работи без връзка с интернет..

### Snap!

Snap! (https://snap.berkeley.edu/) .) също е визуален програмен език, базиран на блокове, чийто дизайн се основава на Scratch, но добавя някои допълнителни функции. Snap!, подобно на Scratch, е уеб-базиран, но също така има възможност да се стартира офлайн чрез браузър. Освен функциите на Scratch, Snap! добавя списъци с класове, процедури за класове, спрайтове на класа, костюми на класа, звуци на класа и продължения на класа, като по този начин го прави по-подходящ за по-напреднала аудитория от Scratch. Snap! се предлага на над 40 езика, включително български, хърватски, английски, гръцки, италиански, португалски, словенски и турски.

*Фигура.4.2* Snap! Потребителски интерфейс

## Alice


*Фиг. 4.3* Alice 3 Code Editor

Alice https://www.alice.org/ е блок-базирана среда за програмиране, която има за цел да мотивира обучаващите се да програмират чрез насърчаване на тяхната креативност чрез създаване на анимации, интерактивни разкази или прости 3D игри (Kelleher , Pausch, & Kiesler, 2007). Тя позволява създаването на виртуални светове с редактор на виртуален свят, където обучаемите могат да добавят 3D обекти и да добавят функции и методи към съществуващи 3D обекти. След като се създаде виртуалният свят, обучаемият може да напише код, за да развие логиката на играта / разказа / анимацията (Sykes, 2007). Alice е добър език за програмиране за учащи без предишен опит, тъй като им позволява да виждат анимираните програми да се изпълняват, докато пишат своя код (Cooper, Dann, & Pausch, 2000).

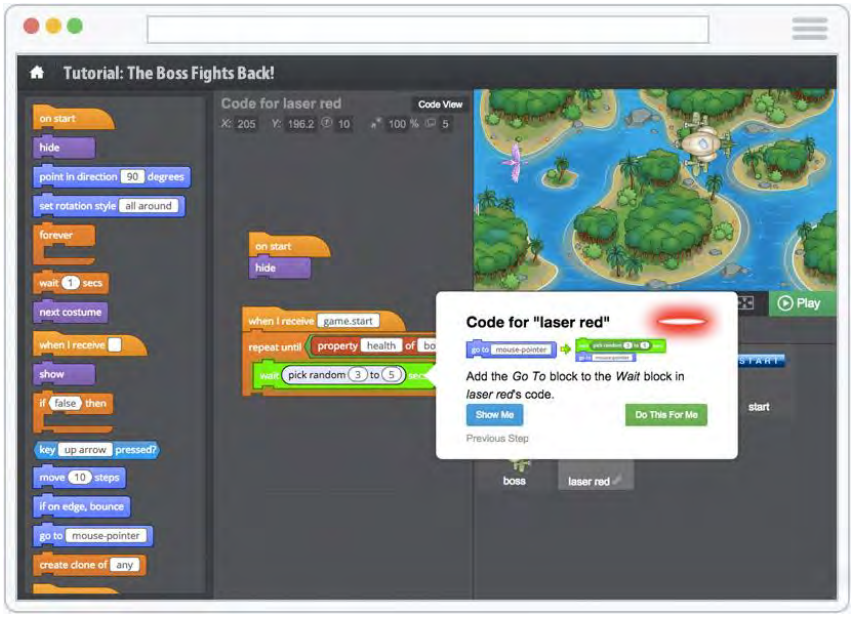
Освен това, друго проучване, което се фокусира върху Storytelling Alice (програмна среда, базирана на Alice 2 с допълнителни творчески инструменти за писане на истории), установява, че макар и Generic Alice, и Storytelling Alice да са еднакво успешни в преподаването на програмни концепции за момичета, с Storytelling Alice момичетата са по-мотивирани и увеличиха времето си за програмиране, което оказа положително въздействие при програмирането.

Alice 3, най-новата част от поредицата, се предлага на 13 езика, включително английски, португалски, гръцки, словенски и български, английски, испански, португалски и немски.

### Tynker

Tynker https://www.tynker.com/ е образователна платформа за програмиране, базирана на визуален език за програмиране с плъзгане и пускане. За разлика от другите представени програмни среди, тази е търговски продукт. Един аспект, който добавя към Scratch, се състои в това как той подхожда към кодирането като игра, в която потребителите трябва да създадат програма, която да накара героите да се движат, да си взаимодействат и да изпълняват различни задачи. Той представя проблеми, които играчите трябва да решат, така че децата да започнат да разпознават модели (Geist, 2016). Също така предлага вграден наставник, който дава стъпка по стъпка инструкции, така че обучаемият да може да се научи как да прилага концепции за кодиране. Обучаваният също така може да визуализира блоковете в JavaScript кода, като им позволява да разберат блока в текстов език за програмиране.

Tynker предлага над 23 курса по програмиране, 11 курса за iPad и 2000+ дейности по кодиране и предоставя индивидуални планове и семейни планове (за до 4 члена). Предлага се само на английски език.

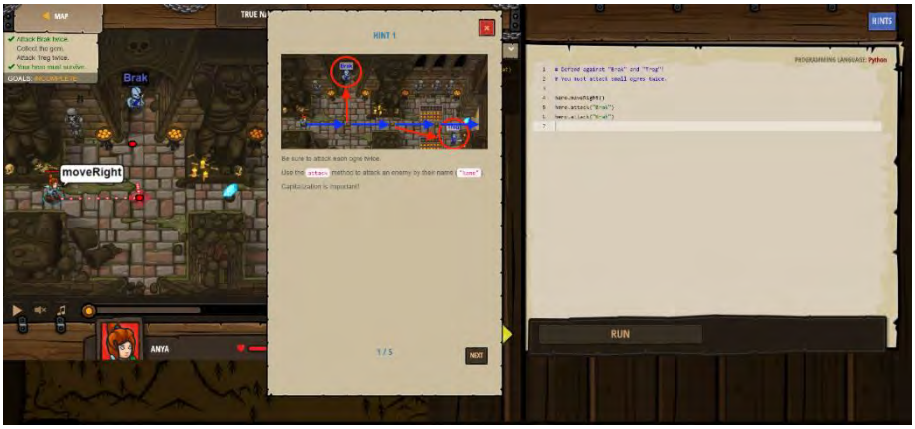



*Фиг. 4. 4* Tynker Потребителски интерфейс

### 4.2.1. Обучение по програмиране в среда, базирана на игри

Тук представяме някои платформи, базирани на игри, предназначени да развият способностите за кодиране.

#### Code Combat




*Фиг.. 4. 5* Code Combat Gameplay

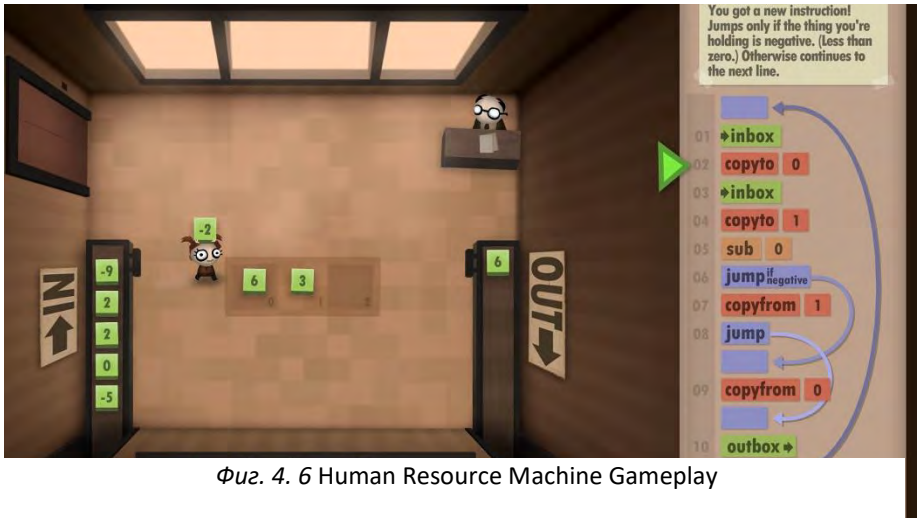
Code Combat https://codecombat.com/ е уеб базирана приключенска игра. Той има планове както за отделни ученици, така и за класове, като предоставя и ресурси за учителите, които да използват по време на часовете. Той има повече от 100 нива безплатно за игра и само за абонати. Предлага се на над 50 езика, включително български, хърватски, английски, гръцки, италиански, португалски, словенски и турски.

От обучаемите се изисква да напишат свой собствен код (или в JavaScript, или в Python), за да накарат героите да се движат, да си взаимодействат и да постигат различни задачи. Той не е базиран на блокове, изисква от учащите да напишат свой собствен код. Той дава подсказки по пътя и въвежда понятия постепенно. Играта е разделена на 5 различни свята, въвеждайки различни концепции, докато играчът напредва през играта: 1) Kithgard Dungeon - синтаксис, методи, параметри, низове, цикли и променливи; 2) Backwoods Forest - If / else, логическа логика, релационни оператори, функции, свойства на обекта, обработка на събития, обработка на входа; 3) пустинята Сарвен - аритметика, броячи, while-цикли, прекъсване, продължаване, масиви, сравнение на низове, намиране на мин / макс; 4) Cloudrip Mountain - литерали на обекти, отдалечен метод, извикване, for-цикли, сложни функции, чертеж, по модул; 5) Ледник Kelvintaph - Разширени техники за напреднали.

Според Miljanovic et al. (Miljanovic & Bradbury, 2018) образователното съдържание на играта включва различни концептуални аспекти на проектирането на алгоритми и решаването на проблеми, фундаментални програмни концепции (като Синтаксис и семантика, променливи и примитивни типове данни; Изрази и задания, вход и изход, условни и итеративи, функции и параметри, и рекурсия) и основни структури от данни (като масиви, хетерогенни агрегати и абстрактни типове данни)).

### Human Resource Machine

Human Resource Machine http://tomorrowcorporation.com/humanresourcemachine (Машина за човешки ресурси) е пъзел игра, базирана на визуален език за програмиране.




*Фиг. 4. 6* Human Resource Machine Gameplay

В тази игра играчът трябва да автоматизира задача чрез програмиране на офис работник, като напредва през все по-трудни пъзели (Johnson & all, 2016). Играчът трябва да създаде последователност от ходове, като плъзга и пуска инструкции, като постепенно се въвеждат нови команди, за да изпълняват по-сложни операции. Чрез този визуален език за програмиране той преподава основни концепции за програмиране чрез сравнение на алгоритми.

**LightBot**

LightBot https://lightbot.com/ е пъзел игра с подобна механика на машината за човешки ресурси, изискваща логическо мислене, за да преминете през нивата. Играчът трябва да ръководи робота, за да запали плочки и да реши 40 нива. Командите, използвани в Lightbot, се показват като икони.

Въпреки че няма изрично изучаване на език за програмиране, играчите „... развиват разбиране за последователност и прилагане на алгоритми“ (стр. 6) чрез фокус върху уменията за решаване на проблеми” (Miljanovic & Bradbury, 2018)(p.6) Въпреки това, чрез играта учениците изучават различни основни концепции за програмиране, като условни условия, итерации, стратегии за рекурсия и отстраняване на грешки. Тъй като пространството за плочките е ограничено, обучаемият трябва да обмисли и ефективността на кода.

Изследване, проведено от Mathrani et al (Mathrani, Christian, & Ponder-Sutton, 2016) използва Lightbot с ученици от средното образователно ниво и установява, че учениците се радват да играят игрите и че това е ефективен начин за изучаване на програмни конструкции като функции, процедури, условни условия и рекурсии. Повечето от участващите ученици също се съгласиха, че подходът е ефективен начин за тях да разберат понятия, които иначе биха били по-трудни за разбиране.

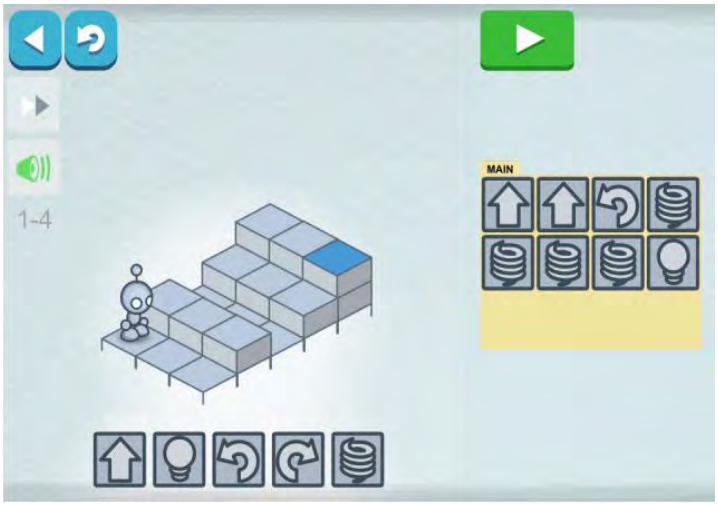



*Fig. 4. 7* Lightbot Interface

### May's Journey – Пътуването на Мая

May's Journey е единствената игра от този списък, която е насочена директно към момичетата от средното училище. Това е 3D пъзел игра, в която играчите решават лабиринт чрез персонализирания език за програмиране на играта, който е вдъхновен от Java. Играта е предназначена да привлече момичета за компютърни науки, като

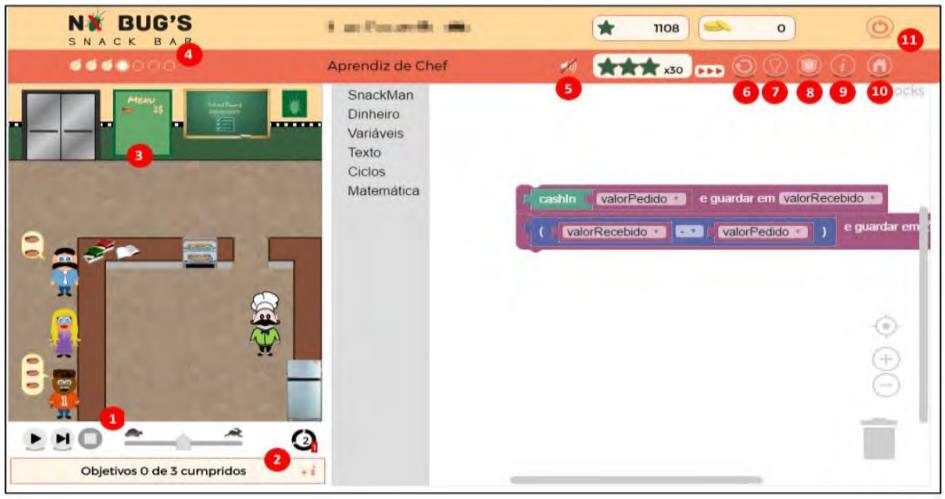

*Fig. 4. 8* May's Journey Gameplay

преподават основите на програмирането. Разработчикът използва псевдо-код, за да улесни прехода между визуалното програмиране и реалните програмни езици. Учебното съдържание включва основни инструкции и логика на последователността, цикли, променливи, ако изрази, сравнители и логическа логика, операции над цели числа и операции върху низове (Jemmali , 2016).

В играта има две фази, фаза на проучване с механиката на типична игра и фаза на кодиране. Във фазата на кодиране интерфейсът е разделен, така че играчът може да въведе кода, докато получава визуална обратна връзка за изпълнението на програмата. Съвети за кода също се предоставят по време на фазата на проучване. В историята героят е момиче, което живее в свят, който се разпада и е отделен от приятеля си, има нужда да поправи света на играта и да разгадае мистерии. Всяка част от мистерията се разкрива постепенно, мотивирайки играчите да играят повече. Тъй като пре-тийнейджърите са основната целева група, те направиха главния герой да изглежда като момиче от средното училище, така че играчът да може да се проектира върху момичето. В дизайна на играта са взети предвид предпочитанията на момичетата към дизайна (висока лекота, топла цветова схема и по-малко агресивна илюстрация. Изглежда, че тестовата група също оценява сюжета.

### No Bug’s Snack Bar

No Bug’s Snack Bar е изследователска сериозна игра с плъзгане и пускане на блок-базиран подход, фокусиран върху решаването на проблеми. Резултатите от обучението




*Фиг. 4. 9* No Bug's Snack Bar Gameplay

са свързани с манипулация на променливи, последователност от действия, условни условия и итерации и стратегии за отстраняване на грешки (Vahldick, 2017).

В играта главният герой работи в Снек бар и трябва да изчиства различни мисии, използвайки код. Една иновация, интегрирана в тази игра, е инструмент за учителите да следят как напредват учениците. Съществува и подход на геймификация, като ученикът печели точки, докато напредва през игрите и намира начини да подобри своите решения на представените проблеми. Играта включва асистент с подсказки и административна система за учителя, за да види кои са обучаващите се, които се нуждаят от помощ, така че той / тя да може да им изпраща персонализирани подсказки. Следователно присъствието на учител се счита за основно.

**Robot ON!**

Robot ON! е друга изследователска игра тип пъзел, насочена към студенти, които изучават C ++. В тази игра играчът е учен, който трябва да активира „Mech Suit“ чрез поредица от задачи. След като играчът завърши ниво, той активира нова система от роботи. Учителите могат да създават свои собствени нива, използвайки други езици за програмиране. Плейърът има различни инструменти, които са цветно кодирани, като цветът на кода съответства на различните инструменти. Играчът се запознава с различни инструменти един по един, придружени от уроци (Miljanovic & Bradbury, 2016).

Вместо да се фокусира върху писането на код, тази игра се фокусира върху разбирането на програмирането, за да преподава умения за отстраняване на грешки и


*Fig. 4. 10* Robot ON! Gameplay

да разбира кода, написан от други. С напредването на играта играчът получава следните инструменти: 1) инструмент за активиране (за да научи контролния поток); 2) инструменти за коментар и un-коментатор (за да научите поведението на кода); 3) намер инструмент (за да научите предназначението на променливите); и 4) инструмент за проверка (за да научите потока от данни).

### Educational Pacman Game

Образователната игра Pacman е изследователска сериозна игра, предназначена да преподава алгоритми за търсене, позволявайки на учениците да видят как се държат различните алгоритми за търсене чрез тяхното графично изобразено изображение(Grivokostopoulou, Perikos, & Hatzilygeroudis, 2016).

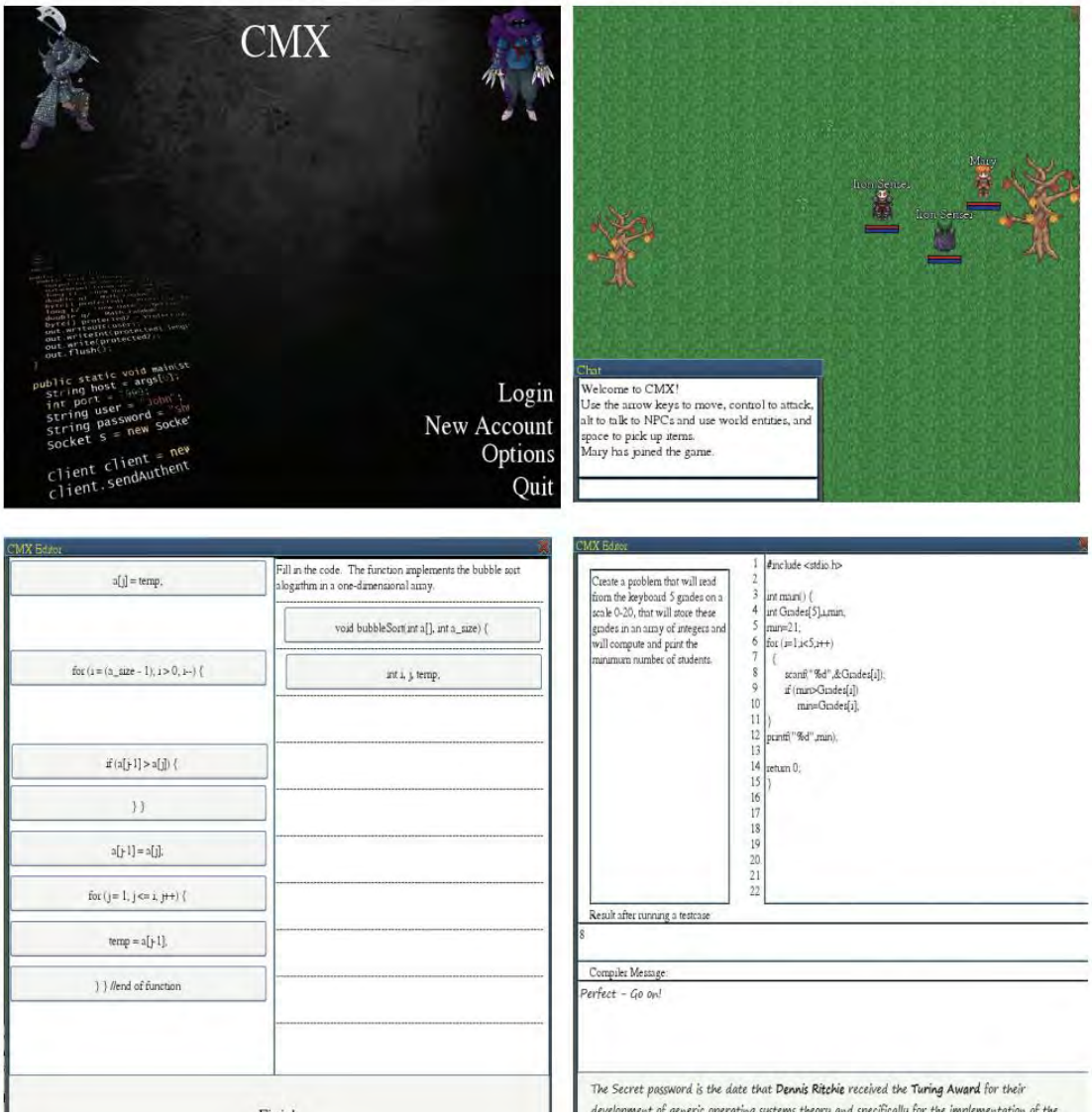

*Фиг. 4. 11* Educational Pacman Game

Играта има два режима:

1) Образователен режим: ученикът може да прочете текстово описание и графичната блок-схема и псевдокод на алгоритъм по свой избор. Играта също така позволява на ученика да научи алгоритъма чрез визуализации на различния Pacman Maze.
2) Режим на игра: ученикът трябва да решава нивата на лабиринта при различни условия и с ограничение във времето. Тези нива са направени така, че ученикът трябва да приложи специфичен алгоритъм за търсене, за да премести Pacman в лабиринта - от ученика се изисква да стигне от произволна позиция до веселие или включване чрез преместване на героя по конкретния алгоритъм.

## CMX

CMX CMX е масивна мултиплейър онлайн ролева игра (MMORPG) за обучение по програмиране. В играта има два отбора - бисквити и хакери, които се състезават помежду си, за да намерят паролите в глобална фабрика за токсични отпадъци. Те са подпомогнати от Senseis, които им помагат в изучаването на езика за програмиране C. Играта включва три нива на Senseis, достъпни чрез отключване на парола на предишното ниво. Учениците като че ли повишават нивото си на разбиране на C - те могат не само да отговарят на теоретични въпроси, но и да поставят изпълними програми чрез плъзгане и пускане на инструменти, както и писане на програми на C (Malliarakis, Satratzemi, & Xinogalos, 2014).




*Фиг. 4. 12* CMX Game Environment

ТРаблица 4.2. Сравнение на игровите среди за обучение по програмиран

| GAME | PLATFORM | TARGET GROUP | LANGUAGE(S) | LEARNING UNITS | TYPE OF GAME |
|---|---|---|---|---|---|
| Code Combat | Web-based | 9+ | JavaScript, Python | syntax, methods, parameters, strings, loops and variables, If/else, Boolean logic, relational operators, functions, object properties, event handling, input handling, Arithmetic, counters, while-loops, break, continue, arrays, string comparison, finding min/max, Object literals, remote method, invocation, for-loops, complex functions, drawing, modulo | Commercial Game (*Freemium*) |
| Human Resource Machine | Windows, Mac OS X, Linux, iOS, Android | 9+ | Visual Programming Language-based | Input & output and conditionals & iteratives<br>algorithm comparison | Commercial Game (*Paid*) Puzzle Game |
| Lightbot | iOS, Android, Windows, Mac OS X | 9+ | Visual Programming Language-based | Conditionals and Iteratives and Recursion<br>Debugging Strategies | Commercial Game<br>*Paid*<br>Puzzle Game |
| May's Journey | Windows | 12-18 | Custom Programming Language (inspired from an Object-Oriented Language like Java) | Basic instructions and sequence logic, loops, variables, if statements, comparators and Boolean logic, operations on integers, operations on strings | Research Game<br>Puzzle Game |
| No Bug's Snack Bar | Web-based | 18+ | Visual Programming Language | Variable Manipulation, Sequence of Actions, Conditional & Iteratives, Debugging Strategies | Research Game |
| Robot ON! | Windows | 18+ | C++ | Syntax & Semantics, Variable & Primitive Data Types, Expressions & Assignments, Conditionals & Iteratives, Program Comprehension | Free/Research Game |
| Educational 'Pacman' Game | Windows | 18+ | Game for learning AI search Algorithm | Algorithms (Basic textual description of Algorithms and corresponding graphical flowchart along with Pseudocode) | Research Game |
| CMX | Windows | 18+ | C | Theory on arrays, outputs of variables after an array's program's execution, syntax of array programs, entry of variable values in an array, calculation of the array's sum, calculation of the maximum value of the array, | Research Game<br>MMORPG |

# 5. ИГРОВАТА ПЛАТФОРМА CODING4GIRLS ЗА УЧИТЕЛИ И УЧЕНИЦИ

## 5.1. Coding4Girls платформата и подхода дизайн мислене

Перфектна комбинация от ИКТ и сериозната игра е показана в резултата от проекта O2 - Насърчаване на развитието на умения за програмиране сред момичета чрез сериозни игри, насочени към изграждане на умения за програмиране сред млади хора, главно момичета, в основно и средно образование чрез използване на софтуер разработен в рамките на проекта.

Този софтуер е предназначен за преодоляване на съществуващата пропаст между мъжкото и женското участие в обучението по компютърни науки и кариерата чрез въвеждане на методически интервенции за обучение, които правят компютърните науки привлекателни за всички. Този софтуер и учебните дейности, които трябва да бъдат проведени, са разработени с цел да стимулират участието на момичетата в компютърните науки. Те са проектирани следвайки подхода на дизайн мисленето, замислен като творчески процес, който помага на учениците да проектират смислени решения съвместно със своите връстници.

Този процес на проектиране е много ефективен благодарение на своята структурирана рамка за идентифициране на предизвикателства, събиране на информация, генериране на потенциални решения, усъвършенстване на идеи и тестване на решения.

Процесът е рекурсивен по природа и изисква взаимодействие. Всеки етап от процеса изисква преразглеждане и призоваване през целия учебен опит, за да се насърчат експериментирането, осъществимостта на решението и размисълът.

Проектираните мисловни дейности дават възможност за съвместен опит в и извън класната стая. Учениците са пряко ангажирани със събирането на информация, генерирането на знания, комуникацията и представянето.

В проекта Coding4Girls, подходът дизайн мислене използва основните предимства на обучението, базирано на игри, за да насърчи и мотивира учениците в техния учебен процес. Той използва някои важни игрови елементи (напр. Точки и предизвикателства)

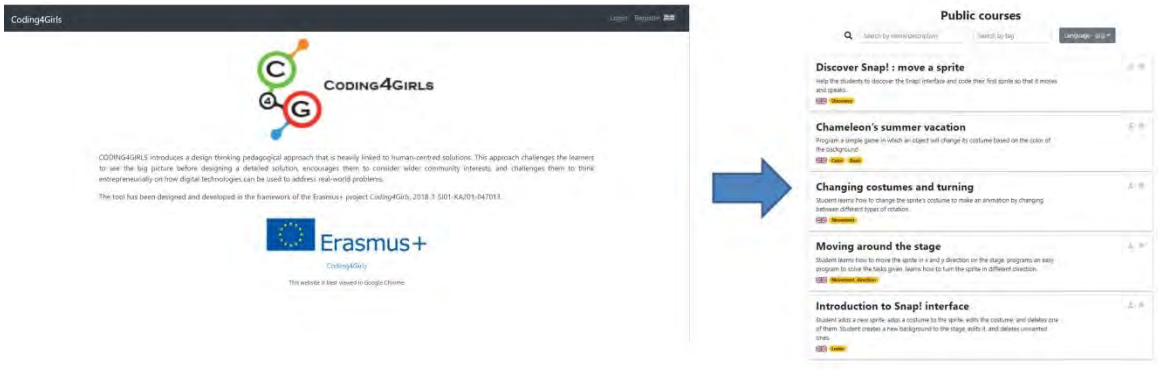
и сценарии, които правят определена дейност по-забавна чрез увеличаване на степента на участие, мотивация и постигнати резултати.

По-специално, използването на предизвикателствата, докато помага на учениците да видят общата картина, преди да проектират подробно решение, ги насърчава и предизвиква да мислят предприемачески за цифровите технологии и как те могат да бъдат използвани за решаване на реални проблеми.

Софтуерът, разработен в рамките на проекта, се състои от две различни взаимосвързани части: платформата на учителите и средата за ученически игри

## 5.2 Платформата за учителя

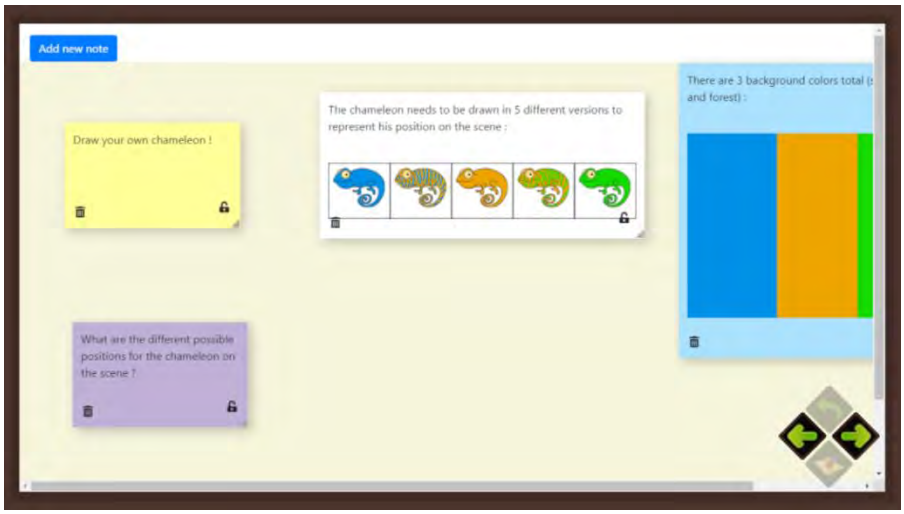
Платформата за учителя е уеб базирана платформа, където учителите могат да проектират и подготвят специфични курсове за развиване на умения за кодиране на своите ученици чрез Snap! Платно. Вътре в платформата всеки учител има на разположение както частна, така и обществена зона. В първата те могат да подготвят своите курсове въз основа на нуждите на своите ученици. Вторият е хранилище на всички създадени курсове от други колеги учители. Целта е да се споделят, използват или се вдъхновяват от учебните дейности, които вече са подготвени от други колеги (Фиг. 5. 1). Разбира се, всеки учител може да персонализира всички публични курсове според спецификата на своите класове.




Фиг. 5.1 Платформата за учителя и някои курсове за кодиране, налични в нея.

Тези курсове са съставени като място за групиране на тематично свързани дейности, всички свързани с общ проблем. Проблемът е представен в самото начало на курса пред студентите, които могат да направят мозъчна атака заедно, за да разработят съвместно предварителни решения. Това е много важна фаза от учебния процес, където учениците могат да генерират нова идея и да бъдат творчески потопени в процеса за решаване на проблеми.

Платното за мозъчна атака може да бъде подготвено от учителите в платформата на учителите, като се използват някои ключови въпроси или някои забележки и размисли, които да се използват по време на дискусията на учениците. Софтуерът предлага възможността да го подготвите на виртуална дъска, като използвате текст, изображения или видео, както е показано на следващата фигура Фиг. 5.2:



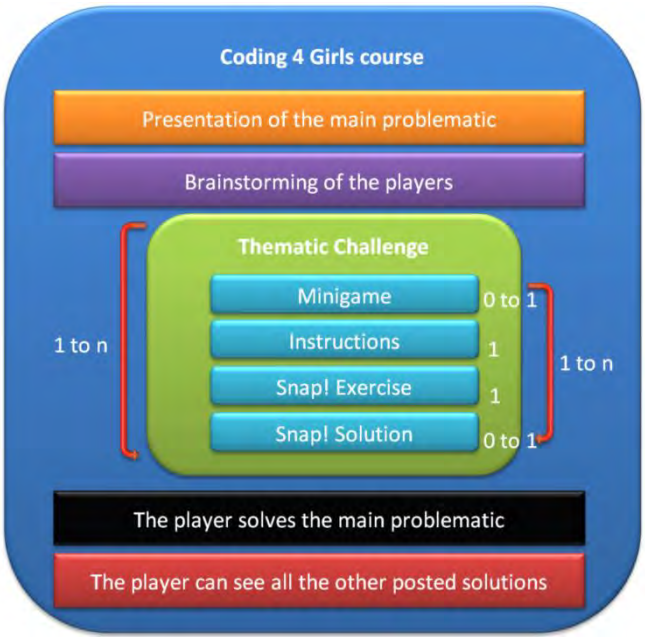

Фиг.5.2. Зона за мозъчна атака в средата за ученически игри за принос и дискусия на учениците.

Всички публикации имат икона на катинар, която може да бъде отворена, когато публикацията е редактируема или заключена, когато публикацията не може да бъде редактирана. Само учителите имат възможност да заключват и отключват след него, учениците могат да знаят само дали даден пост е заключен или не, без да могат да действат по него.

След фазата на мозъчна атака учениците се дават поетапно; специфични дейности (представени в последователен ред), предназначени да представят инструментите,

необходими за решаване на всеобхватния проблем. Тези дейности за кодиране ще бъдат представени чрез използване на Snap! платно, чрез полузапечените задачи за предизвикване на учениците в решението на проблема и презентацията на крайното решение.

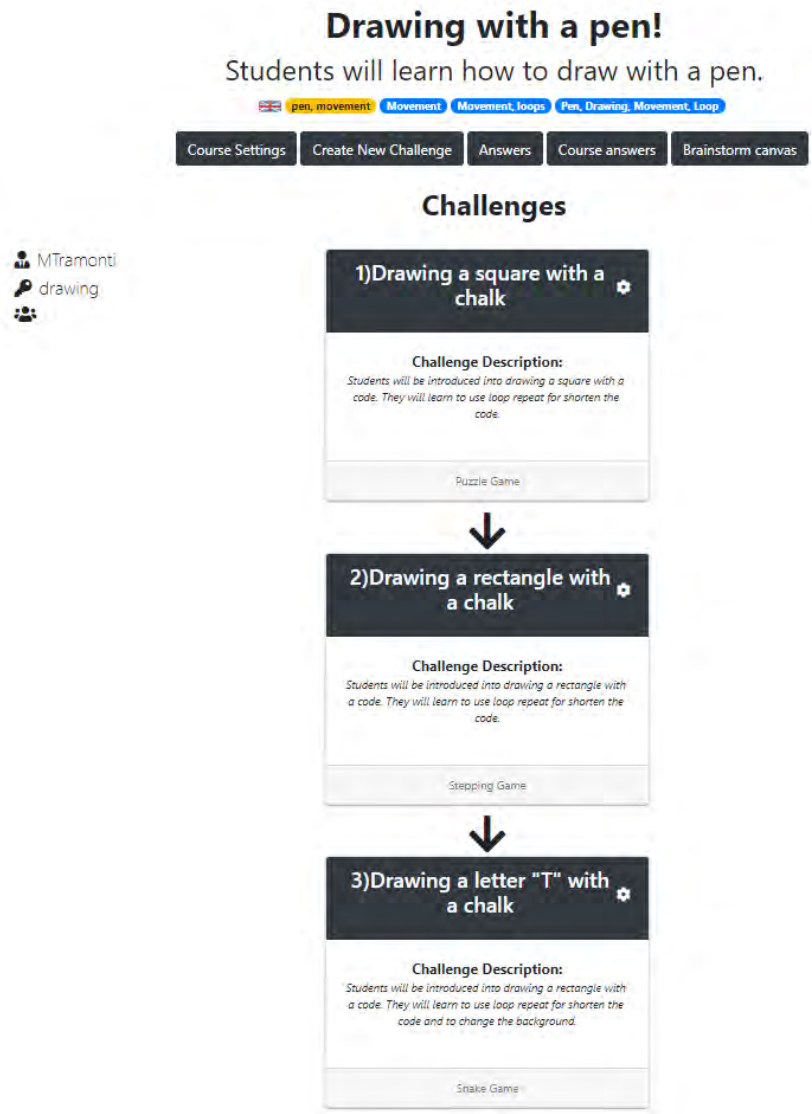
На Фиг. 5.3 е показана структурата на всеки C4G курс, публикуван в платформата за учители..




Фиг 5.3 Структура на C4G course

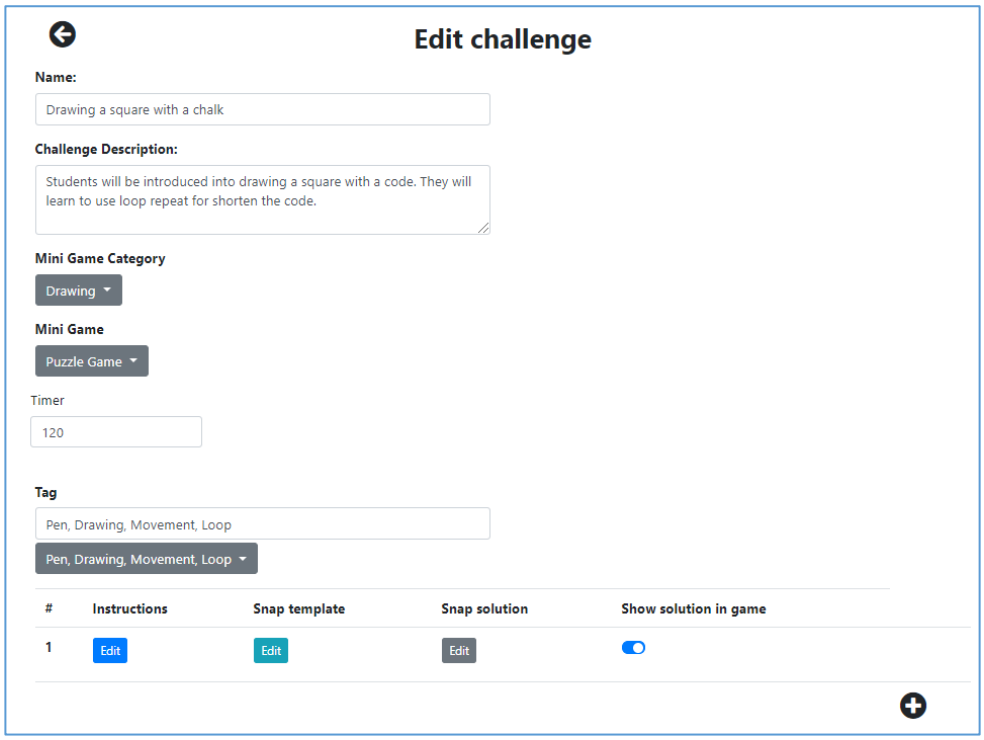
Всички групови дейности в курса се наричат предизвикателства, които се представят на учениците в определен ред, предварително, от учителя (Фигура 4). Например, ако учителят иска да създаде курс за основни познания по програмиране, първата дейност може да се отнася до концепцията за булеви числа, втората условна структура и последната - циклите. Това ще позволи на учителите да подготвят персонализирани учебни стъпки за своите ученици в Учителската платформа и учениците да бъдат постепенно доведени до крайната учебна цел в средата на играта на учениците.

Разбира се, учениците трябва да играят и да решават всички предизвикателства по реда, установен от учителите, преди да вземат окончателното решение.




Фиг. 5.4 Пример за предизвикателства в курс по кодиране в платформата за учители.

Всяко предизвикателство може да бъде свързано с мини-игра, разработена от екипа на проекта. Целта е да се помогне на учениците да разберат по-добре какъв може да бъде крайният резултат от конкретно изследвано свойство за кодиране. Например, ако предизвикателството е свързано с изучаването на свойството "loop", Match-3 е една от разработените мини-игри, които могат да бъдат свързани с него. Следователно, всяко предизвикателство може да има, освен Snap! платно за решаване на задачата, мини-игра, която ученикът ще трябва да играе, следвайки страница синструкции,определени от учителя по време на подготовката им в платформата за учители. (Фиг. 5.5).

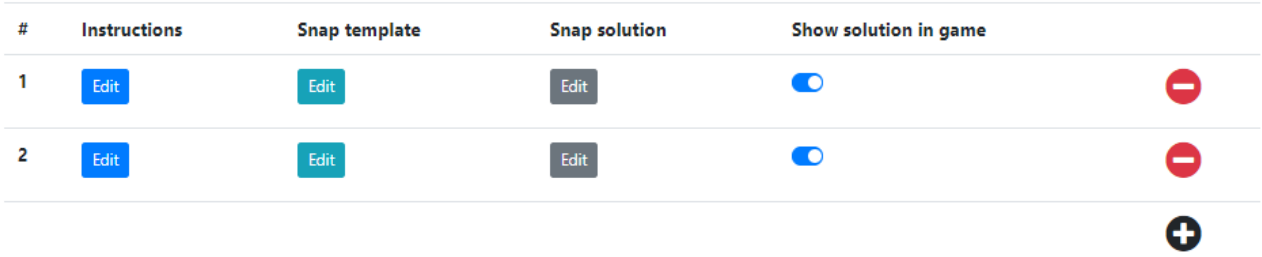



Фиг. 5.5 Изглед отвътре на предизвикателство в платформата за учители

Всички игри, предлагани (до момента 11) на учениците, са свързани и опростяват действителните концепции за програмиране, върху които са проектирани курсовете и сценариите C4G. Не е задължително обаче да включите мини-игра в предизвикателството. Това е само избор на учител. Всяко предизвикателство се състои от три части:

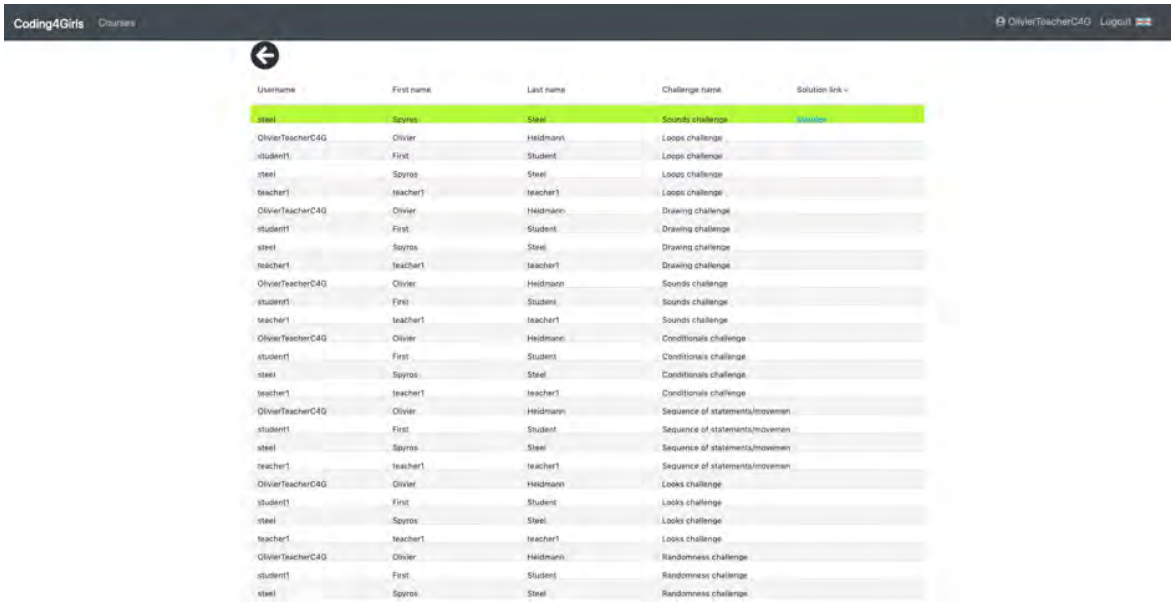
1. Инструкции, при които учителят може да даде указания или обяснения относно задачата, която трябва да се изпълни от техните ученици.
2. Snap шаблон, където учениците ще се опитат да решат възложената задача, като използват блоково кодиране на Snap с отворен код! в средата на играта на учениците
3. Бързо решение, при което учителите могат да решат да покажат или не окончателното решение на задачата.

Ако предизвикателството е сложно, по отношение на знанията за програмиране, то може да бъде структурирано в повече от една задача (ФФиг. 5.6). По този начин учителите ще могат да водят, стъпка по стъпка, своите ученици, през различни фази, за решаване на сложна дейност, разпадаща се на по-малки части.




Фиг. 5.6 Разбиване на по-малки задачи на сложно предизвикателство

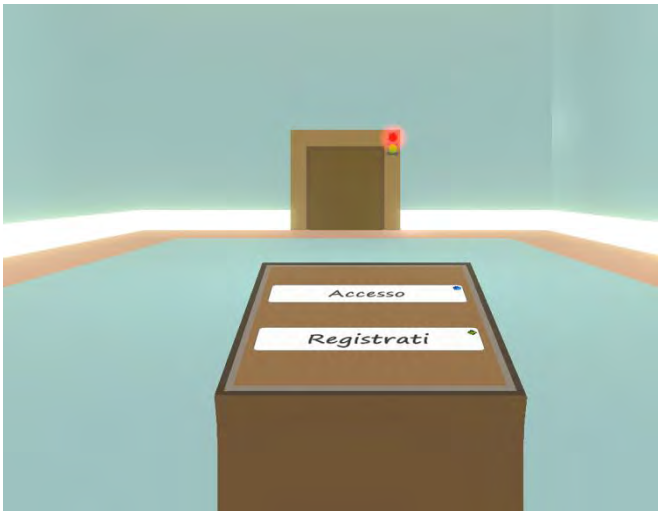
Освен това учителите могат да съберат всички снимки! отговори на техните ученици след подаването им във всяко предизвикателство, показани в таблица в платформата за учители. Таблицата съдържа списък на потребителите и подробностите за всяко предизвикателство, което съществува в курса. Всеки ред се отчита потребителско име, собствено име, фамилия, име на предизвикателство или решени предизвикателства. Ако няма изпратено решение, колоната „Връзка към решение“ ще бъде празна. В противен случай редът ще бъде маркиран и думата „Решение“ ще се появи в последната колона и, щраквайки върху него, ще се покаже изпращането на потребителя (Фиг.5.7).




Фиг. 5.7 Отговорите на предизвикателството от потребители с решение

## 5.3. Игровата среда за учениците

Учениците ще имат достъп до тези предизвикателства чрез Student Game Environment, която е Unity 3D видеоигра за изтегляне, където учениците могат да открият и завършат курсовете, подготвени от техните учители по забавен, увлекателен и игрив начин. Курсовете, използващи елементи от подхода на дизайн мисленето, представят на учениците всеобхватен проблем за решаване и представят инструментите за неговото решаване чрез подход стъпка по стъпка. Те, както бе споменато по-горе, се изготвят от учители и се публикуват в Учителската платформа, но учениците могат да имат достъп до тях чрез Игровата среда за учениците




Фиг. 5.8 Първата стая за достъп до средата на играта на учениците

Учениците могат да се присъединят към курс, като въведат правилния код в терминала на портала. Кодът е уникален и трябва да бъде предоставен от учители, които вече са подготвили курса на платформата за учители. След като курсът е изпратен, името (Фиг. 5.9).


Фиг. 5.9 Втора стая с два терминала: един за присъединяване за първи път към курса и един за избор на курс от вече регистрирани

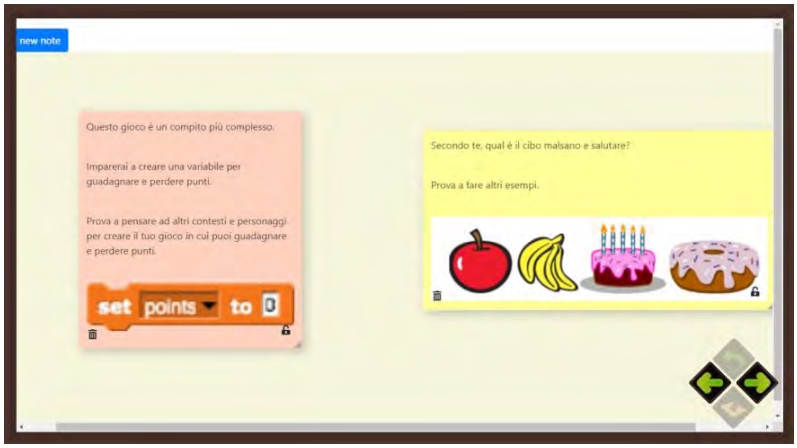
Учениците се насърчават да проектират и кодират игри, които отговарят на специфични нужди или проблеми (в зависимост от избора на учителите). Това е „подход с висок таван с нисък вход“, който позволява на учениците да започнат с лесни проблеми, докато се заемат с по-предизвикателни задачи. Учителите проектират и предоставят на своите ученици „полготови“ сценарии, в които решението е частично готово, като ги предизвиква да изпълнят задачата. Следователно учителите могат да подготвят малки и управляеми модули, като използват софтуера Coding4Girls (C4G), подходящ за нуждите и знанията на техните ученици.

Следвайки подхода на Design Thinking, екипът на проекта е подготвил някои курсове и учебни сценарии, предназначени да предизвикат учениците при решаването на конкретен проблем с кодирането. Понастоящем 22-те сценария на обучение, събрани в „Колекция от учебни листове, базирани на създаване на игри, насочени към учители“, са адаптирани към подхода на дизайн мисленето, следвайки структурата на софтуера C4G.

Сценариите имат различни нива от основно до напреднало за по-способни ученици и се предлагат на английски, словенски, италиански, хърватски, български и гръцки.

Задачите могат да се решават индивидуално и съвместно чрез възбуждане на групова дискусия в класната стая или на практика в средата за игра на учениците. Софтуерът предлага пространство за мозъчна атака, където учениците могат да

обсъждат и споделят своите идеи чрез поставяне и управление на мултимедийни публикации, както е показано на фигура Фиг. 5.11. Всички ученици, участващи в курса, могат да напишат своя принос, който е видим за всички записани в конкретния курс.




Фиг. 5.11: Зона за мозъчна атака в сценария за обучение *Купуване на храна за пикник* в средата за игра на учениците

Следващата фигура показва панела от предизвикателства в средата за ученически игри на курса, създаден от учителя в платформата за учители. Числото от 0 до 13 представлява предизвикателствата. В този панел предизвикателството номер 0 представлява общите инструкции, дадени в началото на играта и отварянето ще отведе учениците до глобалните инструкции за проблеми, свързаните с Snap! шаблон, последван от платно за мозъчна буря. Докато другите предизвикателства, от номер 1 до 13, представляват задачите, подготвени от учителите в тяхната платформа.


Фиг. 5.12. Панелът от предизвикателства в игралната среда за учителя

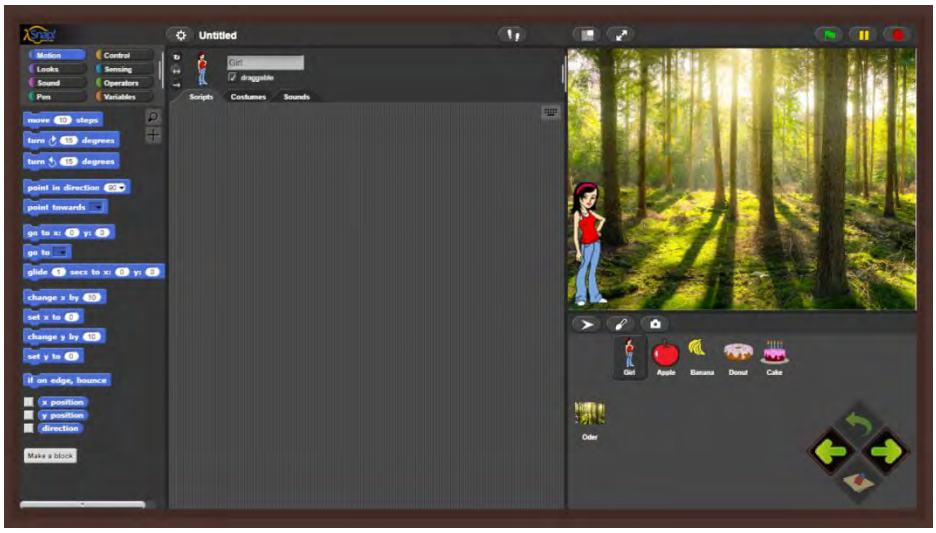
Чрез избора на един от тях, детайлите на предизвикателството ще се появят в долната част на екрана, с името на предизвикателството вляво, свързаната с предизвикателството мини-игра в средата и бутон за стартиране на предизвикателството. Фигура 5.12 показва името на предизвикателството, съответстващо на темата, която ще се изучава, в случая „Условно предизвикателство“. Освен темата за предизвикателството, системата показва свързаната с нея 3D мини игра, напр. "Намерете своя път". Учителите могат да решат коя мини-игра (по избор) ще играят техните ученици, като избират една от съществуващите мини-игри, като например Match3 игра, игра на зарове, игра на инвентара, игра за намиране на пътя, стъпка, звукова игра, игра на змия, Пъзел игра, игра за съвпадение на шаблони и викторина с въпроси с множествен избор.

Фиг. 5.13 Пример за налична мини игра - Игра на зарове

Всяко предизвикателство в игралната среда на учениците е структурирано по следния начин: една уводна мини-игра, илюстрираща концепцията за кодиране; една страница с инструкции за задачата, която трябва да бъде изпълнена в Snap; две Snap! платно - едното да се използва за решаване на задачата, а другото за показване на решението за дейност. Страницата с инструкции е в HTML (Фиг. 5.14), обикновено обогатена с изображения или видеоклипове, за да представи контекста и конкретната цел на учебната задача, която трябва да бъде изпълнена в Snap!

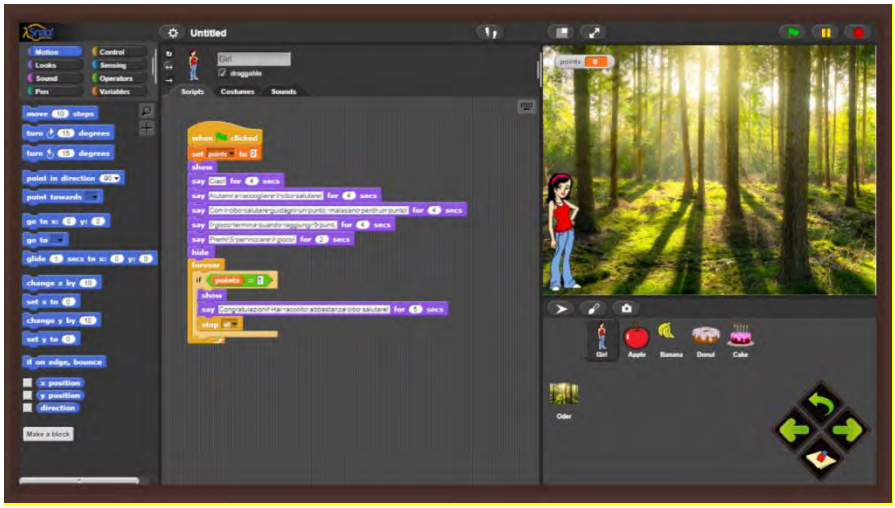

Фиг. 5.14 Пример за учебната страница на предизвикателството в средата за игра на учениците

Тази страница ще бъде последвана от Snap! платно (Фигура 5.15), въз основа на шаблон, предоставен от учителя, съдържащ кодиращата дейност или проблем, който да бъде изпълнен или решен от учениците




Фиг. 5.15 Пример за шаблон на Snap!, предоставен от учителя за решаване на задачата за кодиране в средата за игра на учениците

Още едно Snap! платно ще бъде предоставено за показване на решението на предложеното предизвикателство. Само учителят обаче може да реши дали се показва платно с решение или не. Това ще зависи от управлението на учебния процес, избрано от инструктора.




Фиг. 5.16 Пример за Snap! платно, предоставящо решението на предизвикателството от учителя за техните ученици в средата за игра на учениците

Тъй като всеки курс събира повече предизвикателства, тези стъпки ще се повтарят толкова време, колкото броят на общите предизвикателства, идентифицирани от учителите. Това позволява разбиването на една сложна кодираща дейност на прости елементарни стъпки, при които отговорът и / или решението на предходната дейност се превръща в шаблон, в който следва да бъде изпълнена следващата дейност. В крайна сметка курсът се състои от определен брой предизвикателства, които представят обучението по програмиране чрез постепенно изграждане на скеле. След като учениците завършат всички предизвикателства на курса (Фиг. 5.16), те се връщат към първоначалния проблем с кодирането и ще бъдат помолени да синтезират новите знания, които току-що са придобили. В самия край на курса играчите могат да видят и всички решения на проблема, предложени от останалите ученици, като предизвикват моменти на конфронтация. Този C4G подход и инструменти позволяват на учениците да развиват, усъвършенстват и подсилват уменията за кодиране чрез персонализирани тренировъчни дейности, съчетаващи забавни и учебни задачи.

# 6. ПРИМЕРНИ УРОЦИ (УЧЕБНИ ЛИСТОВЕ)

## 6.1. За учебните листове със сценарии на уроци;

В рамките на проекта Coding4Girls са разработени 22 учебни сценария. Те описват открай дейностите по реализирането на подходите на CODING4GIRLS и дизайн мисленето. Достъпни са на английски език и на езиците на партньорите по проекта – Български, Хърватски, Гръцки, Италиански, Португалски, Словенски и Турски. Можете да ги откриете на сайта на проекта. (https://www.coding4girls.eu/results_03.php)

Подготвените учебни листове със сценарии, представят в съкратен формат информация, която ще помогне на учителите да интегрират предложените сериозни игри и методологии за учене чрез подхода на дизайн мисленето в техните преподавателски практики. Те следват CODING4GIRLS подхода, основан на активно учене, дизайн милене и проектиране на обучението и включват информация за всяка учебна дейност, която трябва да се разработи за изграждане на умения за програмиране за момичета и момчета.

Налична е следната информация:

- Обща образователна цел на съответния урок;
- Понятия, обхванати от урока;
- Специфични учебни цели;
- Очаквани резултати от обучението;
- Използване стъпка по стъпка на подхода за обучение CODING4GIRLS, основан на игрово базирано обучение и дизайн мислене;
- Методи за оценка на усвоените знания;
- Въпроси за иницииране на дискусия между учащите се в контекста на съвместни дейности в клас и работа в екип

Учителите могат да използват сценариите и игрите в предложената последователност или могат да ги избират свободно според своите предпочитания и нужди. Трябва да се вземе под внимание адаптацията на уроците към съответната възрастова група по отношение на предварителни по математика – координатна система, отрицателни числа, реални числа и др.

Учебните листове обхващат както общата функционалност на предложената сериозна игра, включително процеси на взаимодействие с потребителите и генериране на обратна връзка, така и описания на всички учебни дейности, които ще бъдат приложени в предложената сериозна игра..

Подготвените учебни листове варират от основни, с въвеждане и затвърждаване на една концепция за програмиране, до по-напреднали с използване на множество концепции за програмиране. Следващата таблица представя предложения ред на дейностите.

Table 6.1. Списък на уроците, разработени в рамките на проекта Coding4Girsl

| **Основни обучителни сценарии** | | |
|---|---|---|
| 1 | **Въведение в Snap! интерфейс**<br>Запознаване със Snap! среда за визуално блоковопрограмиране | UL |
| 2 | **Време е да оживите вашия спрайт**<br>Намерете блокове за програмиране, свържете ги, преместете спрайта, накарайте спрайта да каже нещо. | UL |
| 3 | **Придвижване по сцената**<br>Създаване на смислена последователност от блокове за движение на героите (спрайтовете). | UL |
| 4 | **Смяна на костюми и обръщане** | UL |
| 5 | **Звуци от фермата**<br>Добавяне, импортиране, запис и възпроизвеждане на звук. | UL |
| 6 | **Лятната ваканция на Хамелеона, опростена версия**<br>Запознаване със събития, чувствителност на цветовете, логически стойности, проверка и реагиране на две различни състояния на играта | UL |
| 7 | **Помогнете на принца и принцесата да открият техните животни**<br>Използване на условни, рисуване | UL |
| 8 | **Рисуване с креда (тебешир)**<br>Използване на цикли/loop/, обръщане, промяна на фона (сцената) | UL |
| 9 | **Прибиране на боклука и почистване на парка**<br>Запознаване с променливи, дублиране на спрайтове, части/blocks/ от код | UL |
| 10 | **Хранене на котките**<br>Използване на променливи (вътре / извън цикъла), цикли, произволни числа, конкатенация на низове, оператори, вход. | UL |
| 11 | **Познай броя на котките в приют**<br>Използване на случайни числа, въвеждане на променливи, условни блокове, оператори за сравнение, брояч. | UL |

| Сценарии за напреднали | | |
|---|---|---|
| 12 | **Улов на здравословна храна**<br>Използване на променливи, условни, цикъл, точка в посока, произволни | UL |
| 13 | **Разказване на приказки** | SWU |
| 14 | **Рисуване** | UNIRI |
| 15 | **Хвани мишката**<br>Използване на цикли, условни условия, променливи | UL |
| 16 | **Купуване на храна за пикник**<br>Използване на променливи, условни условия, оператори | UL |
| 17 | **Операции**<br>Използване на случайни числа, смяна декор на сцена, вход, условни блокове | SWU |
| 18 | **Рециклиране** | SWU |
| 19.1 | **Свири на пиано 1** | SWU |
| 19.2 | **Свири на пиано 2** | UNIRI |
| 20 | **Тест** | SWU |
| 21 | **Опростена игра PACMAN**<br>Използване на движение на обект въз основа на събития, сензори на цветовете, логически стойности, проверка и реагиране на две различни състояния на играта | UL |

Всичките 22 учебни листове са представени в Приложение 1. В Приложение 2 са дедени кодовете за всички игри, представени в публичната секция на средата Coding4Girlst.

## 6.2. Examples with use of game environment developed in the frame of Coding4Girls project

Игровата среда за учениците е сериозна видеоигра, базирана на Unity, достъпна за Windows и Mac OS X.

Когато играе играта, потребителят първо открива встъпителна сцена, представяща необходимите европейски отказ от отговорност, като проектът и логото на Erasmus + се показват за 4 секунди.

Играчът пристига след това в голяма празна стая, наречена лоби, контролира невидим аватар и разглежда света на играта от гледна точка на първо лице. В средата на стаята е поставена конзола, която позволява на потребителя да влезе в сървъра C4G, използвайки своите идентификационни данни или чрез създаване на нов акаунт. След

като влезете, вратата се отваря и играчът може да продължи към друга стая, където може да се абонира за нов курс или да осъществи достъп до вече активиран курс. След като бъде избран желаният курс, потребителят ще премине през плаващ портал, за да продължи с играта и да открие точното съдържание на избрания курс.

Курсът се формира от колекция от свързани предизвикателства за кодиране с помощта на Snap! Платформа, всяка илюстрирана от миниигра. В идеалния случай всяко предизвикателство ще илюстрира аспект или стъпка от урока, който трябва да се научи в курса.

По подразбиране C4G е дефинира някои основни концепции за кодиране, които са илюстрирани от мини-игра: Цикли, условни условия, променливи, изявления, паралелизъм, оператори, събития.

Освен това учителите могат също така да решат да заменят мини-играта с тест с множество възможности за избор или изобщо с нищо, оставяйки само Snap!.


Фиг. 6.1 Категорията мини-игри, достъпна в платформата за учители

Категорията мини-игри, достъпна в платформата за учители (Фиг. 6.2).


Фиг. 6.2 Панел на предизвикателство в средата на играта на учениците

Интегрирането на мини-играта в конкретно предизвикателство е да се илюстрира концепцията за програмиране зад задачата, проектирана от учителите. Не е задължително. Това означава, че учителят може да реши да има или не мини-игра за своите ученици в предизвикателството и коя мини-игра да играе, като ги избере от списъка на съществуващите мини-игри.

Необходимостта да се свърже мини игра с предизвикателството трябва да се има предвид, ако тя води до добавена стойност от гледна точка на разбиране, доказателство за концепция, визуализация на механиката и, разбира се, ангажиране и мотивиране на учениците по-добре в техния учебен път..

Към момента съществуват 11 различни мини-игри, вариращи от игра Match3 до тест с въпроси с множествен избор..


Фиг. 6.3 Мини-игрите, налични в платформата на учителите

Когато потребителят въведе предизвикателство, свързано с мини-игра, той ще бъде телепортиран до мястото, където се провежда мини-играта. Има много места за различните мини-игри, вариращи от пясъчни плажове до заснежени планини.


Фиг. 6.4 Пример за местоположение, свързано с мини-игра

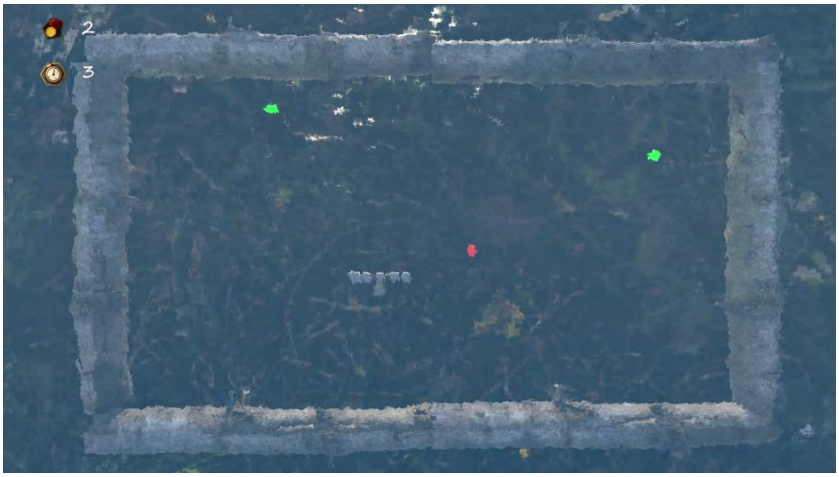
Сред наличните мини-игри има игра Змия, игра Match3, игра на зарове, игра с инвентар, игра за намиране на пътя, игра със стъпки, игра със звукове, игра с пъзели, игра за съвпадение на шаблони и игра с въпроси с избираем отговор.

Всяка мини-игра има своите трудности и гамата от налични мини-игри обхваща различните възможни възрастови групи на учениците и / или темите по програмиране, преподавани в курсовете.

Една от най-простите мини игри е играта със змия, разположена на брега на реката, и играчът трябва да следва дървения знак със стрелка до червен кръг, за да започне. (Фиг. 6.5.)


Фиг. 6.5 Пример за местоположението, свързано с игра със змия

Играта със змия (Фигура 5.7) се провежда във водата. Вие управлявате змията с помощта на стрелката на клавиатурата 4 и се опитвате да оцелеете възможно най-дълго. Змията се увеличава по размер, като яде яркозелените точки, но може да умре, ако се извива върху опашката си, ако удари границата на екрана или изяде червена точка. Всяка изядена зелена точка добавя една точка към резултата.




Фиг. 6.6 Игра „Змия“

Тази проста мини игра предлага добра илюстрация на няколко програмни концепции като цикли, движения, променящи се графики и т.н.

Друг пример е играта Match3. Провежда се в заснежен, планински район, по който играчите могат да се разхождат свободно


Фиг. 6.7 Местоположението, свързано с игра Match3

Мини играта Match3 се основава на типична дейност Match3, при която трябва да плъзгате и пускате съседни квадратчета, за да ги накарате да изчезнат, ако образуват група от три плочки или повече от същия тип


Фиг. 6.8 Match3 mini-game

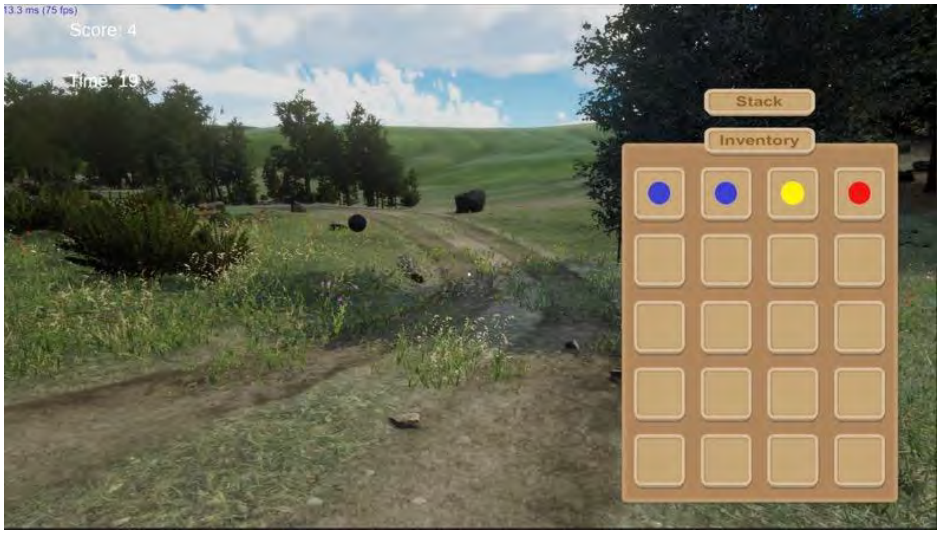
The dice game takes place in dark woods. The interactive part of the mini-game is inside an abandoned cabin, on the chair across the door. It is a wooden board with 2 dice on it.

Играта на зарове се провежда в тъмна гора. Интерактивната част на мини-играта е в изоставена хижа, на стол срещу вратата има дървена дъска с 2 зара върху нея.

И играчът, и изкуственият интелект трябва да хвърлят заровете, когато им дойде редът и този с най-голямата сума от двата зара печели рунда.

Fig. 6.9 Игра на зарове

Играта Събиране на запаси/инвентар се провежда в полетата, до малка гора. В района има плаващи цветни сфери и играчът трябва да ги събере, съгласно инструкциите, написани от учителя върху дървения знак.




Фиг. 6.10 Inventory game

Мини играта Стъпки се провежда на един плаж на острова по залез слънце. Играчът трябва да отговори на въпроса, написан на голямата скала в центъра на плажа, като настъпи буквата, изписваща верния отговор.

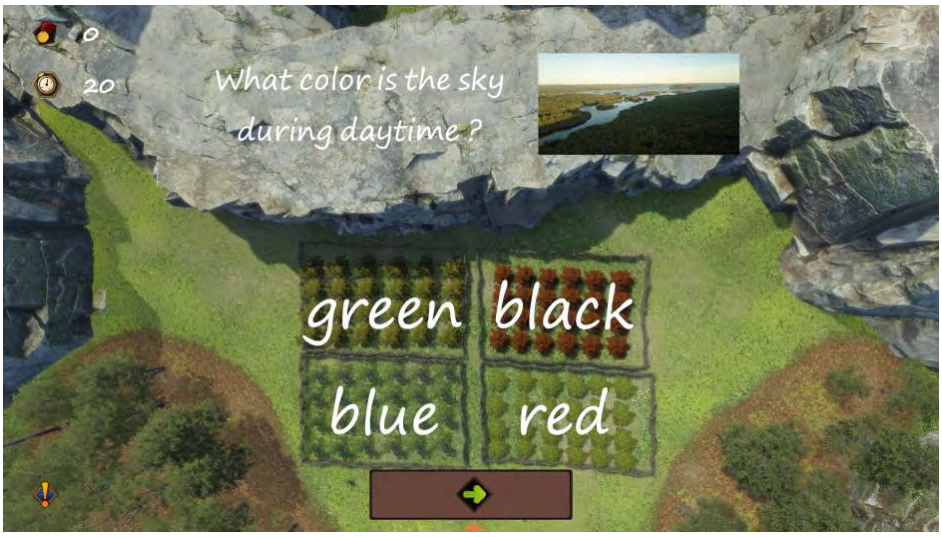
Когато играчът стъпи върху един от камъните с буква, формираща отговора, тя ще бъде маркирана в зелено и ще се появи на втората голяма скала до тази, на която е написан въпросът. Ако потребителят стъпи на грешен камък, той ще бъде маркиран в червено и ще трябва да започне отново изписването на отговора


Fig. 6.11 Игра Стъпки

Тестът с множество възможности за избор на отговор се поставя в поле отстрани на скала и играчите могат свободно да се разхождат в началото и за да започнат мини-играта.

Въпросите се появяват в горната част на скалата и всяко поле ще съдържа четирите възможни отговора, заедно с изображенията, ако има такива.

Иконата за медал представлява текущия резултат на ученика въз основа на верните отговори и часовникът отброява секундите, за да прекрати това предизвикателство. Бутонът със стрелка в долната част ви позволява да преминете към следващия въпрос.




Фиг. 6.12 Тест с въвпроси с избираем отговор

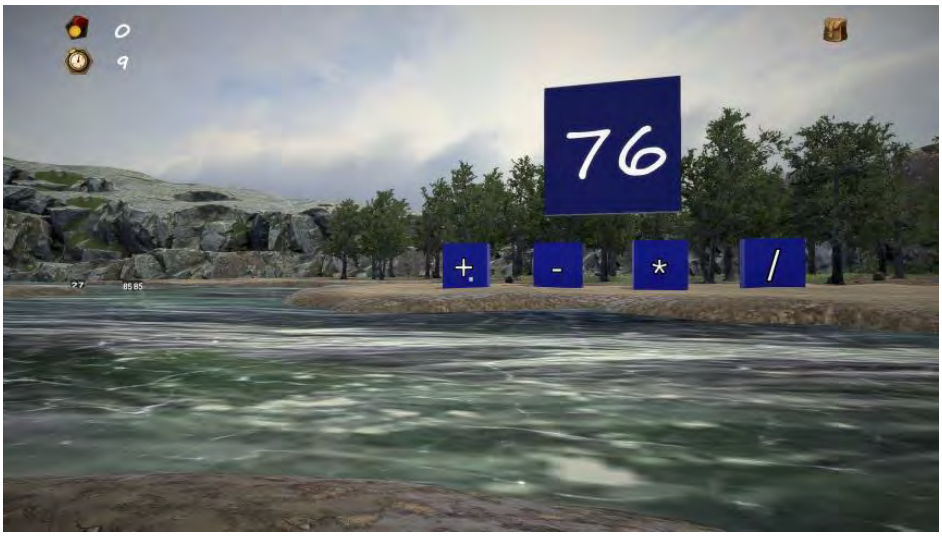
Декорът на пъзела е скалист район на картата. Целта е да се намерят разпръснатите табла в района и да се решат пъзелите, като се създаде линия, която започва от белия кръг на панела и отива към червения квадрат.

Фиг. 6.13 Табло в играта Пъзел

Играта за съвпадение на шаблони се провежда на брегoвете на река. Има няколко кутии с цифри, които започват да се носят по реката към морето. Целта на играта е играчът да ги вземе, за да извърши определена математическа операция.

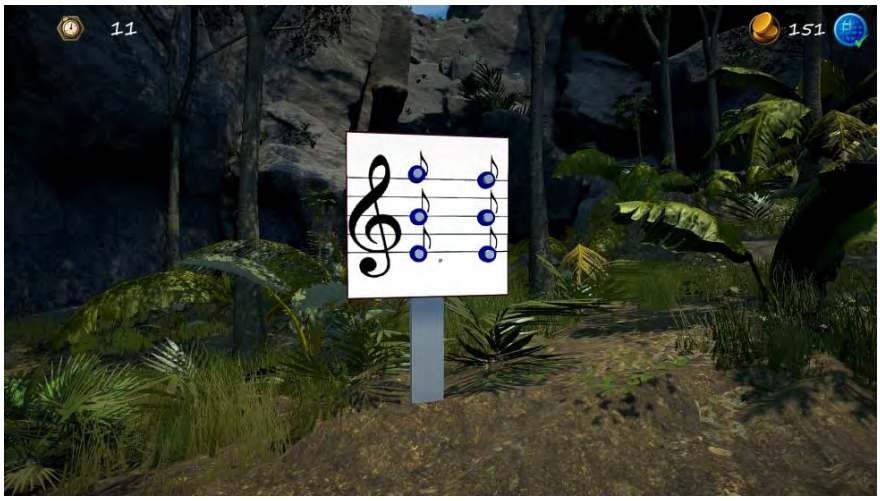
С други думи, играчите трябва да използват плаващите числа и наличните операции, за да достигнат определеното число, записано на една синя маса. Цифрите на брега и плаващите числа се генерират произволно, но по начин, който гарантира, че винаги има възможна комбинация за достигане на цифрата на брега. Наличните операции зависят от учителя и могат да бъдат едно от следните:

- Разширени операции (степен, квадратен корен, квадратичен остатък по модул)
- Тригонометрия (Cos, Sin, Tan)
- Булеви оператори (AND, OR, XOR)




Фиг.6.14 Игра за съвпадение на шаблони

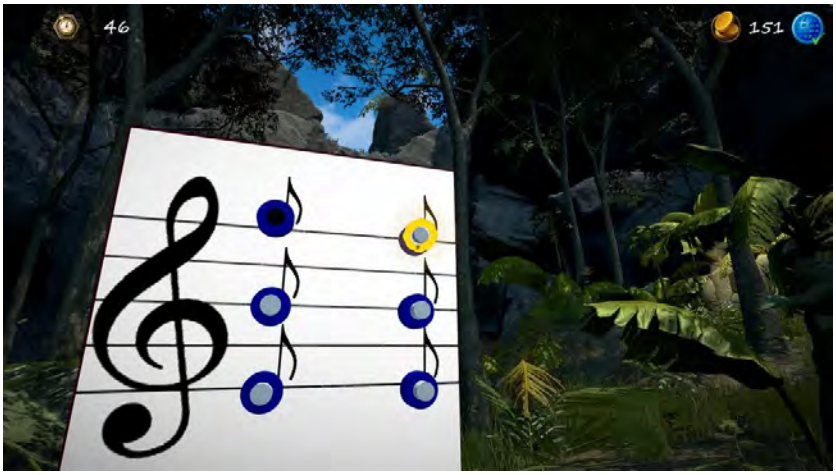
Друг пример за мини игра е звуковата игра, която изисква много внимание от играчите, за да успее в задачата си. Провежда се в тропическа гора и играчът трябва да намери петте разпръснати панела и да реши за всеки от тях пъзел въз основа на звуците, които чува в джунглата.




Фиг. 6.15 Един от панелите в звуковата игра

Всяка колона на панела представлява един звук и всеки ред представлява тона на звука. Бутоните отгоре са за високите тонове, а тези отдолу за ниските. Когато потребителят е близо до панел, ще се възпроизведе комбинация от 3 възможни звука (една птича песен с нисък тон, една със средна височина на тона и една с висок тон). Комбинацията от птичи песни се повтаря в цикъл, като всяко повторение се отделя от останалите с 3 секунди тишина.

Играчът ще трябва да обърне голямо внимание на реда, в който се чуват песните на птици, и да го възпроизведе съответно на панела. Например на първия панел (Фиг. 6.16) играчът може да чуе първо нисък звук и след това висок. Правилният отговор е следователно да натиснете бутона за ниско отляво на панела и бутона за висок отдясно.




Фиг. 6.16 Отговорът на учениците на панела в играта Звук.

За да избере определен бутон, играчът просто трябва да насочи курсора на мишката към него и бутонът ще бъде маркиран в жълто. Ако натиснатият бутон е правилен, той ще стане зелен, в противен случай ще стане черен.

Звукът често е важен за всяка игра, но е от решаващо значение в тази и в центъра на всяко ниво на играта. Това е така, защото играчите трябва да разпознават между високи, средни и ниски тонове и следователно трябва да се концентрират върху тях.

Игровата среда може да се използва от всички ученици, най-вече на възраст от 10 до 15 години, и има за цел да насърчи равнопоставено участие в дейности свързани с програмиране в класната стая и извън нея. Той показва концепциите за програмиране чрез 3D виртуална среда, която учениците могат да изследват, мини-игри, демонстриращи концепции, среда за сътрудничество за споделяне на идеи между учениците с цел потенциално решение, прегледи на решенията на други участници в играта за изграждане на по-нататъшна стратегия за справяне с даден проблем , и сравнение на решенията с това, което се въвежда от учителя.

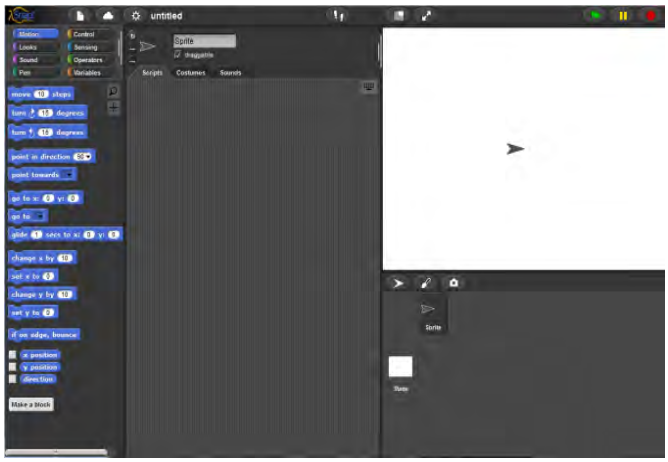
Може да се предвиди бъдещо развитие и усъвършенстване за обогатяване на софтуера C4G с много повече мини-игри, които да помогнат за илюстриране на допълнителни концепции за програмиране. Модулният характер на учебната игра C4G и начинът, по който тя е проектирана в Unity, позволяват този вид бъдещи разширения.

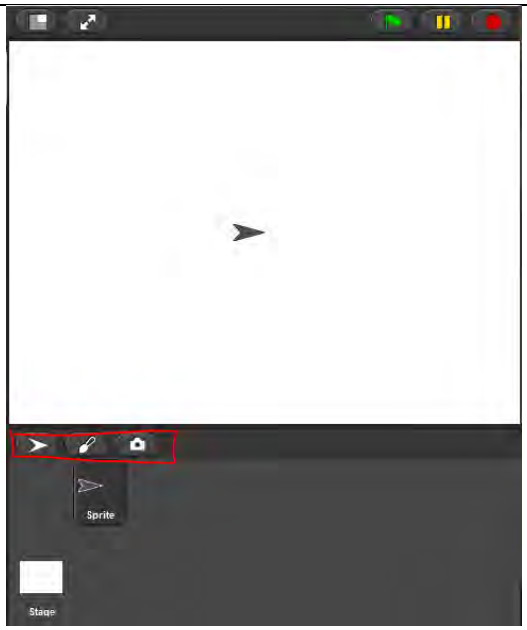
# ПРИЛОЖЕНИЕ 1. УЧЕБНИ ЛИСТОВЕ

## Основни учебни сценарии

### Учебен сценарий 1 - Въведение в Snap! интерфейс

| | |
|---|---|
| **Учебен сценарий Име** | Въведение в Snap! интерфейс |
| **Предишен опит в програмирането** | Не се изисква |
| **Очаквани резултати** | Общи резултати от обучението:<br>● Запознаване със Snap! среда за визуално блоково програмиране<br>Конкретни резултати от обучението:<br>● ученикът може да добави нов спрайт<br>● ученикът може да добави костюм към спрайт и да го редактира<br>● ученикът е в състояние да центрира спрайта, така че въртенето да работи правилно<br>● ученикът може да добави нов фон за сцената и да го редактира |
| **Цел, задачи и кратко описание на дейностите** | Ученикът добавя нов спрайт, добавя костюм към спрайта, редактира костюма и изтрива един от тях. Ученикът създава нов фон на сцената, редактира го и изтрива нежеланите.<br>**Цел: До края на часа учениците ще нарисуват любимия си герой и жизнената му среда, реална или въображаема, за да го използват в игра. В научните изследвания рисуването на спрайтиве е единтефицирано като подходящо за целевата група. Това прави дейността по-мотивираща за всички ученици.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Учителят демонстрира дейността<br>Самостоятелна работа |
| **Форми на обучение** | Работа с класа/групата (фронтални дейности)<br>Индивидуална работа |

| | |
|---|---|
| **Резюме на обучението** | (Мотивация-въведение, прилагане, осмисляне и оценка)<br>В края на часа учениците ще нарисуват любимия си герой и неговата среда на живот, реална или въображаема, за да го използват в бъдеща игра.<br>[Стъпка 1]<br>Покажете на учениците уеб страницата, където могат да намерят Snap! (https://snap.berkeley.edu/).<br>Покажете им различни части на интерфейса: раздел с блокове, раздел, където те могат да сглобяват скриптове, да сменят костюми, да добавят звуци, сцена със спрайт върху него, списък на спрайтове.<br>  <br>[Стъпка 2]<br>Можете да създадете нов спрайт, като щракнете върху един от трите бутона: |




Ще се опитате да нарисувате нов спрайт, затова щракнете върху четката и ще се отвори изскачащ прозорец, където можете да нарисувате вашия спрайт по подобен начин като в Paint.

Задача за ученици: Създайте първия си спрайт. Имате 10 минути.

След изготвянето на спрайта трябва да се уверите, че центърът на въртене на спрайта е там, където искате да бъде.

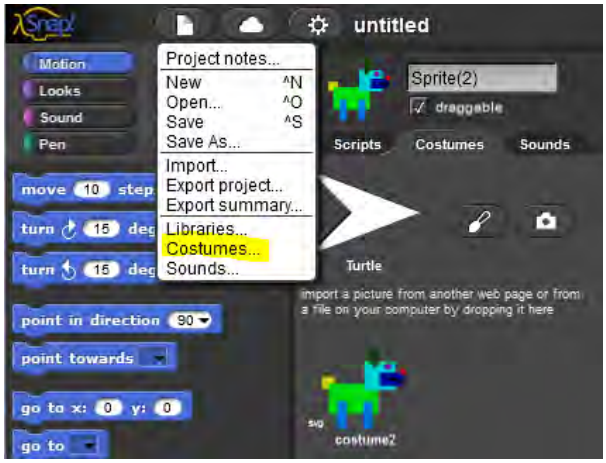
Задача за ученици: центрирайте вашия спрайт

[Стъпка 3]

За да редактирате вашия спрайт, изберете раздела **Костюми**, който се вижда само когато щракнете върху вашия спрайт. Щракнете с десния бутон върху костюма, който искате да редактирате, и изберете редактиране. Можете също да дублирате костюма си или да го изтриете със същото меню.


[Стъпка 4]

За да импортирате вече съществуващ костюм, щракнете върху иконата с нарисуван лист хартия и изберете Костюми ...

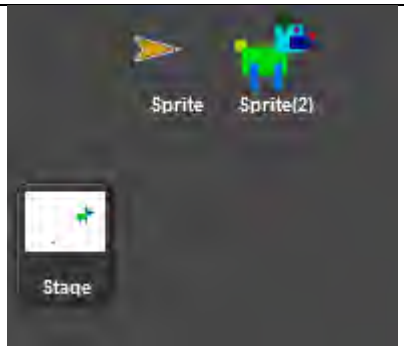
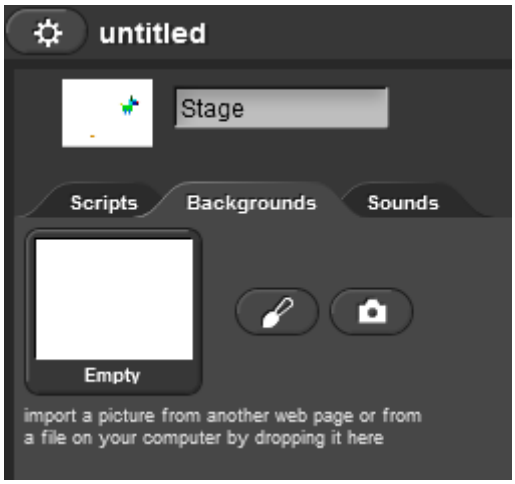



Отново, тази опция ще се покаже само когато вашият спрайт бъде щракнат под сцената.

<u>Задача за учениците:</u> изберете костюм и го добавете към спрайта

[Стъпка 5]

Сега имате своя герой и трябва да добавите малко фон към сцената. За да направите това, първо щракнете върху сцената вместо върху героя под сцената. За да добавите нов фон, изберете раздела Фонове:

| | |
|---|---|
| |  <br>  <br>Задача за учениците: нарисувайте своя фон.<br>Задача за учениците: претърсете съществуващите фонове и добавете един от тях, така че да имате два.<br>Задача за учениците: Намерете начин да редактирате фона си. Намерете начин да изтриете един от вашите фонове, така че да остане само един.<br>Рефлексия и оценка: Успяли ли са учениците да нарисуват своя характер и обкръжение там, където живее? Имали ли са проблеми? Как ги решиха? |
| **Инструменти и ресурси за учителя** | https://snap.berkeley.edu/ |
| **Инструменти и ресурси за учениците** | Инструкции за учениците (C4G1_InstructionsForStudent_BG.docx) |

## Учебен сценарий 2 - Време е да оживите вашия спрайт

| | |
|---|---|
| **Учебен сценарий Име** | Време е да оживите вашия спрайт |
| **Предишен опит в програмирането** | / |
| **Очаквани резултати** | Общи резултати от обучението:<br>• Ученикът знае къде да намери определени блокове за програмиране и как да ги свърже в последователност<br>• Ученикът знае как да премества спрайт<br>• Ученикът знае как да накара спрайта да каже нещо<br>• Специфични резултати от обучението, ориентирани към алгоритмично мислене:<br>• Създаване на смислена последователност от блокове |
| **Цел, задачи и кратко описание на дейностите** | Ученикът открива къде се съхраняват блоковете за програмиране и как да намери подходящите, какви категории блокове има и как да свърже блоковете в последователност |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Учителят демонстрира дейността<br>Самостоятелна работа |
| **Форми на обучение** | Работа пред класа/групата<br>Самостоятелна работа |
| **Резюме на обучението** | (Мотивация-въведение, прилагане, осмисляне и оценка)<br>Ще накарате героя си да се движи и да каже нещо през този час. Можете да им покажете пример за програма, която те ще програмират през този час.<br>[Стъпка 1]<br>Първо нека да разгледаме къде са налични програмните блокове, които можете да използвате. Къде са те? |

От лявата страна можете да намерите различни категории блокове: Motion, Looks, Sounds, Pen, Control, Sensing, Operations и Variables. Първо ще използваме блокове 

.

Задача за ученици: Първо намерете блока и след това щракнете двукратно върху него. Какво се случва?

[Стъпка 2]

За да започнете да свързвате блоковете в програма, трябва да плъзнете и пуснете блока 

в раздела Скриптове.

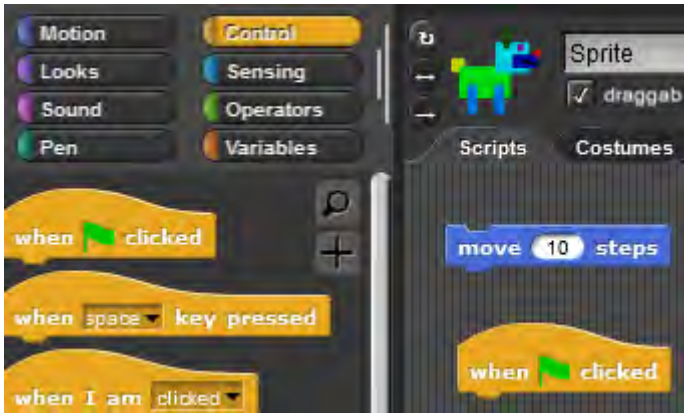

Можете да щракнете двукратно върху блока в раздела **Скриптове**, за да изпълните кода.

[Стъпка 3]

Програмите в Snap! обикновено започват с щракване върху зеления флаг.

Задача за учениците: щракнете върху различни типове категории и се опитайте да намерите блок, който стартира програмата, ако се щракне върху зеления флаг.

Решение:



Ако искате програмата да работи в правилна последователност от стъпки, блоковете трябва да бъдат свързани както при пъзелите.

Като този: 


Сега всеки път, когато щракнете върху зеления флаг, спрайтът ще се движи за 10 стъпки, но от различна позиция на картината.

[Стъпка 4]

Ако в даден блок има празно (бяло) пространство, това означава, че можете да промените цифрите или буквите, написани там.

<u>Задача за ученици:</u> Уверете се, че вашият герой се движи по 30 стъпки наведнъж, вместо само 10.

[Стъпка 5]

Накарайте героя си да каже нещо. Къде ще намерите блока да каже? Изпробвайте каква е разликата между 

и 

, и го обяснете на съученика си.

[Step 6]

Открихте двете команди Say (Кажи) в категорията Looks (Външност). Разликата в тях е, че при 

няма да има изчакване за определен брой секунди преди да програмата да продължи със следващите блокове и текстът ще се визуализира по всяко време.

[Стъпка 7]

Вземете вашия герой от предишния час. Чрез плъзгане на сцената го преместете в лявата част на сцената и напишете програма, която придвижва героя 

от позицията му отляво до дясната страна на сцената. След всеки ход персонажът трябва да каже нещо. Направете повече от едно движение.

| | |
|---|---|
| | Изпробвайте. Завърща ли се персонажът на точно същото положение всеки път, когато програмата ви се изпълнява? Можете ли да намерите блок, който да гарантира, че вашият герой ще започва винаги от една и съща позиция и няма да излиза извън сцената?<br>Съвет за учителя: ако героят „избяга“ от сцената, можете да го извикате обратно на сцената, като щракнете върху него с десния бутон на мишката и изберете show.<br>Блокът, който търсите, е  . За да определите кои стойности за x и y са подходящи, можете да преместите героя си на мястото, на което искате да бъде, и щракнете върху x позиция и позиция y (в долната част на блоковете в категория Motion) и текущите x и y ще се покажат. Просто трябва да ги запишете в белите полета в go to блок.<br>Рефлексия и оценка:<br>Колко пъти вашият герой трябваше да повтори кодът за движение и да изпълни последователността „кажи“, за да изпълни задачата? Еднакво ли е числото за всички в класа? Защо е така? |
| **Инструменти и ресурси за учителя** | **Примерна програма:**<br>https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_dog_goes_home |
| **Инструменти и ресурси за учениците** | • Инструкции за ученика(C4G2_InstructionsForStudent_BG.docx)<br>• Ако ученикът не е нарисувал собствен спрайт и фон, той може да използва:<br>• https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_dog_goes_home_tmp |

## Учебен сценарий 3 - Придвижване по сцената

| | |
|---|---|
| **Учебен сценарий Име** | Придвижване по сцената |
| **Предишен опит в програмирането** | Ученикът знае къде да намери блокове за програмиране и как да ги свърже в последователност |
| **Очаквани резултати** | Общи резултати от обучението:<br>• Създаване на смислена последователност от блокове<br>Специфични резултати от обучението, ориентирани към алгоритмично мислене:<br>• Ученикът позиционира спрайта на сцената<br>• Ученикът променя x и y позицията на спрайта<br>• Ученикът използва цикъл repeat n<br>• Ученикът научава, че посоката на движение на спрайта в move __ steps е спрямо посоката, към която е обърнат спрайтът. |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Ученикът се учи как да премести своя спрайт в позиции x и y на сцената, програмира лесна програма за решаване на задачите, научава се как да завърти спрайта си в различна посока и как това се отразява на блока move __ steps.<br>**Задачи:** Създайте програма, която премества спрайт по позиция x, създайте програма, която премества спрайт в позиция y, създайте програма, която комбинира движение в позиции x и y.<br>**Цел: Ученикът прави разлика между движението в позиции x и y на сцената и използва цикъл.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Учителят демонстрира дейностите пред класа/групата<br>Самостоятелна работа |
| **Форми на обучение** | Фронатална работа/ групова работа<br>Самостоятелна работа |
| **Резюме на обучението** | (Мотивация-въведение, прилагане, рефлексия и оценка)<br>Ще помогнете на различни животни да постигнат целите си. За целта ще трябва да им дадете инструкции как да се движат по сцената. |

[Стъпка 1]

Отворете *Catch the ball* и добавете код на кучето, така че то да хване топката. Използвайте блоковете 

и 

за да направите анимация на куче, което се движи към топката.

Възможно решение на задачата:


Както се вижда, x се променя, когато се движите наляво или надясно. Ако x е 0, вашият спрайт е в средата на сцената. Всичко, което е вляво от центъра на сцената изисква пред числото за x да поставите знак минус „-“. Колкото героят е по-далече от центъра, толкова, по-голямо е числото за позиция x. Вдясно от средата, стойностите на x са числа, по-големи от 0.

**Съвет:** Ако се направи с по-големи ученици, които познават десетичните дроби, времето за изчакване може да бъде по-кратко, напр. 0,1. Ако те знаят какво е координатна система, някои обяснения могат да бъдат пропуснати или описани с понятийния апарат за координати и координатна истема.

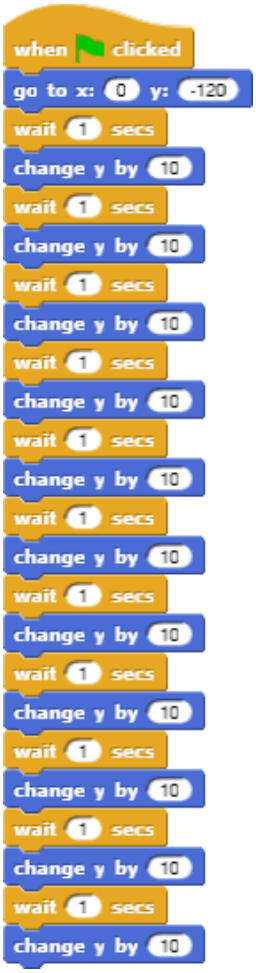
[Стъпка 2]

Отворете Помогни на маймуната да се изкачи на дървото (Ip monkey climb the tree) и добавете код към маймуната, за да донесе бананите.

Използвайте блокове 

и 

,за да направите анимация на маймуна, която се катери на палмата. Възможно решение на задачата:



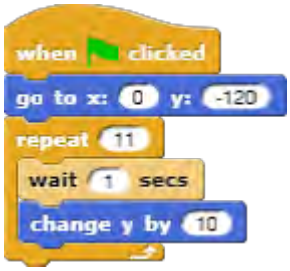

Както се вижда, y се променя, когато се движите нагоре или надолу. Ако y е 0, вашият спрайт е в средата на сцената. Всичко, което е по-високо от средата, има y по-голямо от 0. Ако искате вашият спрайт да бъде под средната линия на сцената, то е точно като да се гмуркате: казвате, че сте под водата, като поставите знака минус - отпред на числото и кажете, колко „метра“ под водата сте и на сцената казвате - колко стъпала под средната линия сте. Ако искате да се спуснете обратно от дървото, използвайте 

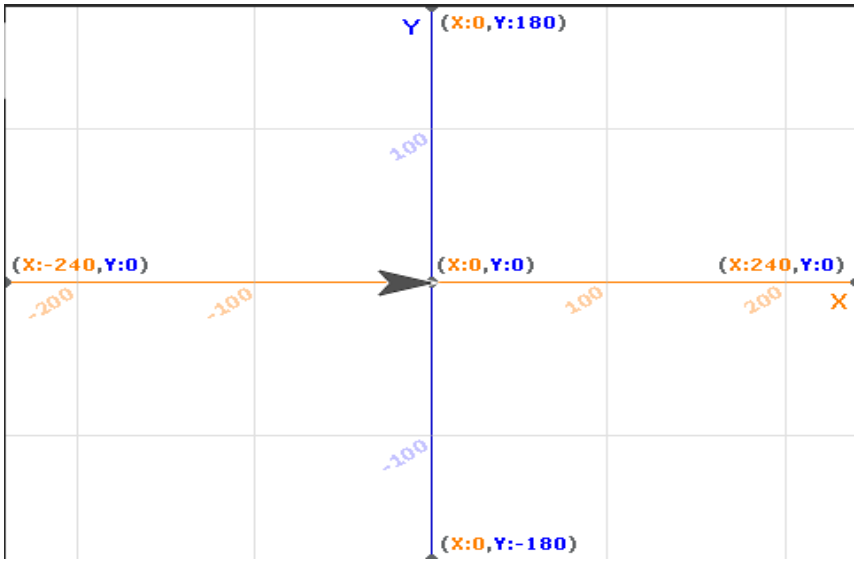
.

[Стъпка 3]

И в двете стъпки трябваше да използвате взаимозаменяемо два блока. Колко пъти трябваше да повторите кода?

Има по-кратък начин за писане на този код, като кажете на компютъра да повтори вашия код определен брой пъти. Това е

| | |
|---|---|
| | блок за цикъл **repeat __** . Можете да го използвате, когато едно и също действие или последователност от действия се повтарят повече от веднъж. Опитайте се да промените кода си и за двете задачи, така че да използвате   . Кодът, който искате да повторите, трябва да бъде поставен вътре в този блок и трябва да напишете колко пъти трябва да се повтори в празното пространство.<br>Код за кучето:<br>  <br>Код за маймуната:<br>  <br><u>Задача:</u> Опитайте се да накарате кучето да тича към топката и обратно.<br><u>Задача:</u> Опитайте се да накарате маймуната да се изкачи на дървото и да се върне надолу. |

| | |
|---|---|
| | Какво ви хареса най-много? Можете да си помогнете с x и y позицията на спрайта, като използвате XY Grid background в Snap:<br>  |
| **Инструменти и ресурси за учителя** | • Възможно решение за Catch the ball: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_moving_x<br>• Възможно решение за Help monkey climb a tree: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_moving_y |
| **Инструменти и ресурси за учениците** | • Catch the ball (Хвани топката): https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Catch_the_ball<br>• Help monkey climb the tree (Помогни на маймуната да се изкачи на дървото): https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Help_monkey_climb_the_tree<br>• Инструкции за ученика (C4G3_InstructionsForStudent_BG.docx) |

## Учебен сценарий 4 - Смяна на костюми и обръщане

| | |
|---|---|
| **Учебен сценарий Име** | Смяна на костюми и обръщане |
| **Предишен опит в програмирането** | Движение |
| **Очаквани резултати** | Общи резултати от обучението:<br>• Създаване на смислена последователност от блокове<br>Специфични резултати от обучението, ориентирани към алгоритмично мислене:<br>• Ученикът променя костюма на спрайт, за да направи анимация<br>• Ученикът променя завъртането на героя (спрайта). |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Ученикът се учи как да промени костюма на спрайта, за да направи анимация. Той също така се научава как да променя различните видове завъртане на спрайта.<br>**Задача:** Създайте програма, която променя костюма на спрайта; във всяка програма задайте подходящ тип завъртане за всеки спрайт<br>**Цел:** Да се усвои начинът за смяна на костюма на спрайта и как да се зададе подходящ тип завъртане на спрайт. |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Демонстрация пред класа/групата<br>Самостоятелна работа |
| **Форми на обучение** | Презентация пред класа/групата<br>Самостоятелна работа |

| | |
|---|---|
| **Резюме на обучението** | (Мотивация-Въведение, Приложение, Рефлексия и Оценка)<br>Ще научите как да направите анимация на спрайт, така че да изглежда като ходене, танци, ...<br>[Стъпка 1]<br>  <br><br>Отворете нов празен проект, щракнете върху иконата, която прилича на бял лист хартия, и изберете **Костюми ...**<br>Щракнете върху Балерина a (Ballerina) и щракнете върху **Импортиране** (import). Направете същото с Балерина b, Балерина c и Балерина d.<br>В раздела Costumes (Костюми) на вашия спрайт вече имате 4 костюма на балерина. Можете да преименувате Sprite на Ballerina, като промените текста над раздела **Costumes**:<br>Сега се върнете в раздела **Scripts** (**Сценарии)** и се опитайте да създадете код, който ще започне, когато се щракне зеления флаг, и 15 пъти променя всяка секунда променя външния вид на Балерина. Ще трябва да използвате блок next costume. Уверете се, че вашата Балерина започва и завършва танца си с двата крака на пода. Началната и крайната позиция не са част от нейния танц. |

Решение:


[Стъпка 2]

Нашата балерина не иска да бъде постоянно на една и съща позиция, затова прави малки движения всеки път, когато сменя костюма. Добавете това движение към нейния танц.

Възможно решение:


[Стъпка 3]

Отворете нов празен проект и импортирайте всички „ходещи“ костюми на Awery (Ейвъри). Добавете подходящ фон, по който да върви Ейвъри. Създайте анимация на Ейвъри, ходеща от лявата страна на сцената до дясната страна на сцената. Опитайте се да разберете как да анимирате Ейвъри по начин, че нейните стъпки да изглеждат свързани, както в реалния живот.

Възможно решение:


[Стъпка 4]

Досега винаги пишехте програма, при която спрайтът се движи само в една посока. В тази задача ще трябва да завъртите „мишката“, за да стигнете до „сиренето“. За да я накарате да се обърне,


а) можете да изберете **point in direction**: където ѝ казвате в коя посока трябва да гледа

или

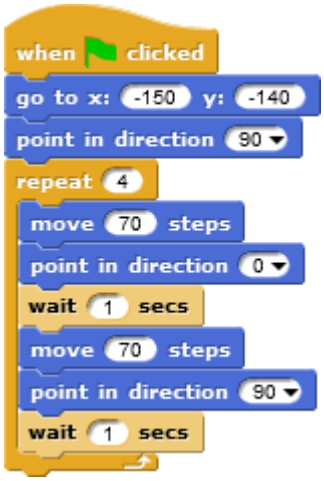
б) можете да ѝ кажете да се обърне под определен ъгъл по посока на часовниковата стрелка 

на определен ъгъл в градуси или обратно на часовниковата стрелка 

Пълният кръг има 360 градуса, така че ако потърсите да се обърнете в посока, противоположна на текущата посока, трябва да се обърнете на 180 градуса. Ако искате да завиете наляво, завъртете на 90 градуса обратно на часовниковата стрелка. Ако искате да се обърнете надясно, завъртете на 90 градуса по посока на часовниковата стрелка.

Задача: Отворете https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Find_cheese.

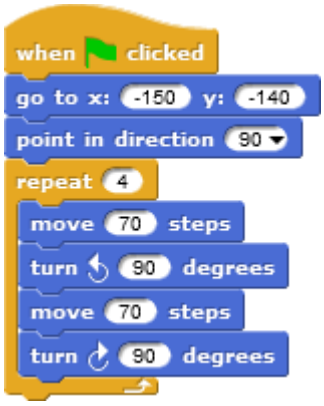
Напишете програма, която мишката трябва да следва, за да стигне до сиренето, ако трябва да ходи само по зелената площ. Направете точка на мишката в посоката, в която тя се насочва и използвайте блока **Move …..steps** (**Премести __ стъпки)**. За да видите как се движи мишката, използвайте блока **Wait 1 seconds** (**Изчакай 1 секунда)** между редовете.




Решение:

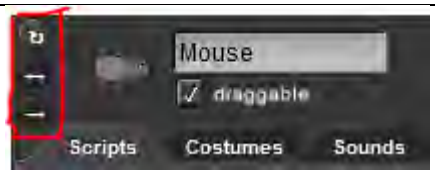
Сега се опитайте да напишете програма със завъртане на 90 градуса.

Решение:




[Стъпка5]

Както видяхте, мишката се обръща в различни посоки, за да стигне до сиренето. Понякога не искате спрайтът ви да се обърне с главата надолу, а просто да се обърнете наляво или надясно, за да не върви „по главата си“. За да сте сигурни, че вашият спрайт се върти така, както искате, трябва да щракнете върху съответната икона отляво на вашия спрайт:

| | |
|---|---|
| |  <br>Кръглата стрелка означава, че вашият спрайт може да се обръща във всяка посока (като мишката).<br>Стрелката <-> означава, че спрайтът ви ще се обръща само наляво или надясно (това е, което бихте използвали кучето да не ходи „на главата си“)<br>Последната -> стрелка означава, че спрайтът винаги ще изглежда такъв, какъвто е (можете да използвате това за маймуната)<br>Опитайте се да пренапишете кодовете си за кучето и маймуната, така че те първо да преминат обекта и да се върнат обратно, като се обърнат. Уверете се, че сте променили правилно стила им на въртене. |
| **Инструменти и ресурси за учителя** | ● Програмни решения за балерина: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_dancing<br>● Avery ходене: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Avery_walking<br>● Решение за намиране на сиренето: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Find_cheese_solution |
| **Инструменти и ресурси за учениците** | ● Намиране на сиренето: https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Find_cheese<br>● Инструкции за ученика (C4G4_InstructionsForStudent_BG.docx) |

## Учебен сценарий 5 - Звуци от фермата

| | |
|---|---|
| **Учебен сценарий Име** | Звуци от фермата |
| **Предишен опит в програмирането** | • Ученикът може да добави фон.<br>• Ученикът може да добави нов спрайт.<br>• Ученикът знае как да накара спрайта да каже нещо. |
| **Очаквани резултати** | Общи резултати от обучението:<br>• добавяне звук от медийната библиотека на Snap,<br>• импортиране на звук от други носители,<br>• запис на нов звук,<br>• възпроизвеждане на звук при натискане на клавиш.<br>Специфични резултати от обучението, ориентирани към алгоритмично мислене:<br>• ученикът добавя звук от медийната библиотека на Snap и го възпроизвежда при натискане на определен клавиш,<br>• ученикът импортира звук от компютър и го възпроизвежда при натискане на определен клавиш,<br>• ученикът записва нов звук и го възпроизвежда при натискане на определен клавиш. |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Програмирайте проста игра, в която играчът научава звуците на животни, като натиска определени клавиши.<br>**Задачи:** В първата стъпка ученикът трябва да избере фона на сцената. След това трябва да програмира жената фермер да казва инструкциите:<br>1) Ако искате да чуете кучето, щракнете върху бутона "D" !;<br>2) Ако искате да чуете кравата, щракнете върху бутона "C" !;<br>3) Ако искате да чуете овцете, щракнете върху бутона "S" !;<br>4) Ако искате да чуете прасето, щракнете върху бутона "P" !;<br>5) Ако искате да чуете коня, щракнете върху бутона "H" !.<br>След това ученикът трябва да програмира задачата според указанията на жената фермер.<br>**Цел:** Учениците ще бъдат запознати как да добавят нов звук и как да го използват. Те също така ще се научат как да използват |

| | |
|---|---|
| | звуковия блок („възпроизвеждане на звук [name_of_sound]“) и контролния блок („при натискане на бутона [When the the_key is pressed]“). |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно учене; обучение, основано на проектиране на игра/игрови дизайн |
| **Форми на обучение** | Преподаване пред класа/групата<br>Самостоятелна работа |
| **Резюме на обучението** | (Мотивация-въведение, прилагане, осмисляне и оценка)<br>**Мотивация-Въведение**<br>Мотивираме учениците, като играем играта (те не виждат кода). Целта на урока е да се създаде подобна игра.<br>  <br>[Стъпка 1]<br>Първата стъпка е да се определи предисторията на играта. Фонът трябва да съдържа различни животни. Имаме три възможности:<br>1. Учениците сами рисуват фона;<br>2. Учениците търсят безплатно изображение онлайн;<br>3. Осигуряваме предварителни ресурси за учениците (ако искаме да спестим време).<br>  <br>Учениците вече знаят как да добавят фон на сцена, така че го правят индивидуално.<br>[Стъпка 2]<br>Втората стъпка е да добавите жената фермер. Имаме същите опции като в първата стъпка:<br>1. Учениците сами рисуват жената фермер; |

| | |
|---|---|
| | 2. Учениците търсят безплатно изображение на жената фермер онлайн;<br>3. Предоставяме образ на жената фермер за ученици (ако искаме да спестим време).<br>Учениците вече знаят как да добавят нов спрайт, така че го правят индивидуално.<br>[Стъпка 3]<br>След това учениците трябва да програмират инструкциите за плейъра. Инструкциите са дадени от жената фермер. Учениците правят това, като използват блокове Say от Looks и Изчакай … сек.. Учениците вече знаят как да правят това, така че го правят индивидуално.<br>  <br>**Изпълнение**<br>След това показваме на учениците как да добавят звук в играта. Имаме три възможности:<br>1. Импортиране на звук от медийна библиотеката на Snap!;<br>2. Импортиране на звук от нашия компютър чрез плъзгане в Snap!;<br>3. Запис на нов звук в Snap!<br>Учителят показва на учениците и трите варианта под формата на фронтално обучение. Когато ги представи всички, учениците започват да програмират поотделно следните задачи (с подкрепата на учителя). |


След това те избират звука на кучето (Куче 1 или Куче 2).


Учениците трябва да програмират звука на кучето, който ще се възпроизведе при натискане на бутон „D“. Те правят това, като от група **Control**, използват блок **Когато клавишът** [the_key] е натиснат и блок Sound / play sound [name_of_sound].


[Стъпка 5]


Учениците трябва да програмират звуци на животни. Първо, те трябва да добавят звуци от компютъра си. Те правят това, като плъзгат звуците в раздела за звуци на фона.

След като импортираме звуците, можем да щракнем с десния бутон върху звуците, за да ги преименуваме. В нашия случай те се наричат: крава, прасе, кон и овца.

След това учениците трябва да добавят звука в скриптове на сцената. Те правят това, като използват *Control / When key is pressed [the_key]* и *блокиране на звук / възпроизвеждане на звук [name_of_sound]*.


[Стъпка 6]

Следващата стъпка е да програмирате поздрава на жената фермер. Когато играчът започне играта, жената фермер трябва да каже: „Добре дошли в моята ферма!“. Първо,


учениците трябва да запишат поздрава на жената фермер. Правят го със звукозапис (червен бутон), намиращ се в раздела (Звуци на жената фермер). Когато записват звука, те трябва да го запазят (бутон Запазване - Save).


След като запазим звука, можем да щракнем с десния бутон върху него, за да го преименуваме. В нашия случай се нарича ферма. Сега учениците трябва да добавят звука в сценариите на жена фермер. Те правят това, като използват блок Play sound [name_of_sound]. 

от група Sound.

[Допълнителна задача]

Ученикът може да надгради фермата, както му харесва, като добави нови спрайтове (фермер, кокошка, трактор, ...) и звуци.

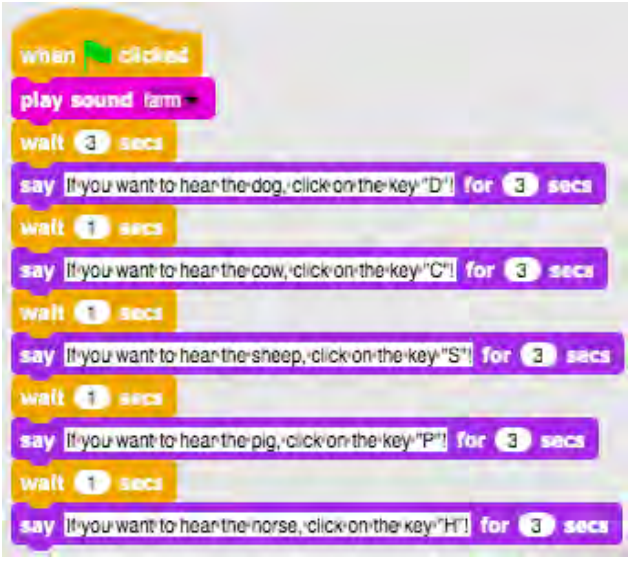
**Рефлексия и оценка**

Учениците обобщават:

- как са добавили звуци в кода си;
- кои блокове са използвали за вмъкване на звук в кода;
- кои контролни блокове са използвали в своя код;
- защо и как са използвали звукови блокове и контролни блокове.

**[Окончателен код]**

*The woman farmer*




*Фонът*

| | |
|---|---|
| **Инструменти и ресурси за учителя** | • Цяла дейност в Snap !: https://snap.berkeley.edu/project?user=tadeja&project=Farm<br>• Уебсайт с безплатни изображения: https://pixabay.com/<br>• Уебсайт на безплатни звуци: https://www.zapsplat.com/<br>• Лайович, С. (2011). Драскотина. Научете се програмиране и пощенски компютърни мачки. Любляна: Пасадена.<br>• Vorderman, C. (2017). Računalniško programiране за деца. Любляна: MK. |
| **Инструменти и ресурси за учениците** | ● Шаблон в Snap!: https://snap.berkeley.edu/project?user=tadeja&project=Sounds%20of%20the%20farm_0<br>● Уебсайт с безплатни изображения: https://pixabay.com/<br>● Уебсайт на безплатни звуци: https://www.zapsplat.com/<br>● Инструкции за ученика (C4G5_InstructionsForStudent_BG.docx) |

## Учебен сценарий 6 – Лятната ваканция на хамелеона

| | |
|---|---|
| **Учебен сценарий Име** | Лятната ваканция на хамелеона |
| **Предишен опит в програмирането** | Не се изискват предварителни познания |
| **Очаквани резултати** | Общи резултати о т обучението:<br>• движение на обект въз основа на събития;<br>• чувствителност към един или повече цветове;<br>• отчитане на булева стойност в логически изрази;<br>• дефиниране, разграничаване, динамична проверка и реагиране на различни състояния на играта,<br>Специфични резултати от обучението, ориентирани към алгоритмично мислене:<br>• ученикът реализира движението на обекта с клавишите със стрелки, използвайки събития и взема предвид ограниченията,<br>• ученикът използва сензорен цветен блок, за да получи булева стойност за четене на еднократно или многократно сензориране,<br>• ученикът осъзнава, че състоянието на обекта може да бъде изразено с цветовете, които обектът докосва,<br>• ученикът прави разлика между две (основни), пет (пълни) различни състояния и знае как да ги изрази с логически изрази,<br>• ученикът осъзнава, че позицията на обекта се променя динамично и използва цикъл forever, за да проверява многократно текущото състояние, |

| | |
|---|---|
| | • ученикът използва if , за да даде различни отговори въз основа на текущото положение на обекта. |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Програмирайте проста игра, в която обектът ще промени костюма си въз основа на цвета на фона.<br>**Цел: Учениците ще бъдат запознати със сензорния цветен блок и как да го използват в логически изрази, за да разграничат динамично променящите се състояния на играта и да дадат правилните отговори.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно обучение, съвместно обучение, решаване на проблеми |
| **Форми на обучение** | Фронтално обучение<br>Индивидуална работа / работа по двойки / групова работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Размисъл и оценка)<br>Хамелеон замина на лятна ваканция. Той обича да се къпе в морето, да релаксира плажа и когато е прекалено горещо, той обича да отива в сянката на близките дървета, за да се охлади. Тъй като той е хамелеон, той променя цвета си според настоящата му позиция.<br>**[Базова версия]**<br>В основната версия трябва да правим разлика между две състояния.<br>[Стъпка 1]<br>Молим учениците да редактират фона на сцената, така че той да е разделен на две части, син и пясъчен, като всяка представлява различно място. Синият цвят е за морето и пясъчен за плажа. Можем да инструктираме учениците да включват други предмети, за да направят фона по-реалистичен, като: вълни, черупки, пясъчни замъци, чадъри и т.н. ... Те трябва да внимават да не избират |

предмети, които са по-големи и изцяло оцветени с различни цветове от заден план. В този случай блокът за цветно разпознаване няма да може да разпознае в коя част от сцената е героят.

[Стъпка 2]

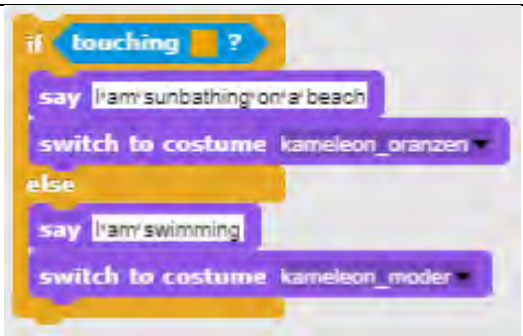
Учениците трябва да нарисуват хамелеон и да оцветят кожата му в два различни цвята:

[Стъпка 3]

Първо трябва да накарат хамелеона си да се движи в четири посоки с помощта на клавиши. Те могат да изберат своя собствена комбинация от клавиши (напр. Клавиши със стрелки или WASD). На този етап предполагаме, че знаят как да го направят от предишни дейности. Трябва да напомним на учениците, че персонажът може да се измести от сцената, ако не използваме подходящ блок при програмиране на движение с блок *bounce if on edge* (отскачане, ако е на ръба).

За да направим движението на хамелеона малко по-реалистично, искаме той да се обърне наляво или надясно към хоризонталната посока, с която сме изправени с блок *point in direction* (посока на движение).

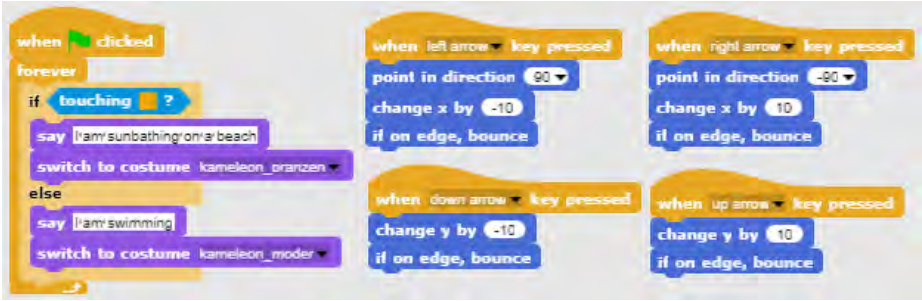
| | |
|---|---|
| | [Стъпка 4]<br>Запознаваме учениците с концепцията за характер, усещащ цвета (цветовете), които той докосва. С блока “touching color?” („докосващ цвят?“) можем да получим информация под формата на булеви стойности - True или False, ако той докосва определен цвят. Тъй като получаваме булева стойност от този блок, можем да я използваме в блока if, където е решено дали ще изпълняваме команди, изброени в тялото му или не.<br>След това обсъждаме с учениците какви са различните позиции на хамелеона на сцената и как можем да ги изразим, използвайки блока touching color?.<br><br>Има две:<br><br>1. Той докосва синия цвят -> Touching color [blue]?<br>2. Той докосва пясъчния цвят -> Touching color [sandy]?<br><br>Когато той докосва определен цвят, ние трябва да променим външния му вид и също така да го накараме да каже къде се намира. Можем да променим външния вид на Спрайт, като превключваме между неговите костюми. Това се прави с блока *Looks/switch to costume[option],* където избираме кой от възможните костюми искаме да покажем. За да накараме хамелеона да говори, използваме *Looks/say[text]* block.<br><br>Защото има само две възможности, които можем да използваме блока за условие “if - else”.<br><br>Можем да изберем кой цвят ще проверяваме и по подразбиране другият цвят ще попадне в случай „друго“. В примерния код избрахме пясъчен цвят: |




[Стъпка 5]

За ситуации, когато трябва да изпълняваме определени команди за цялото време на програмата, която използваме - цикъл завинаги. Всичко, написано под тялото на цикъл forever, ще се изпълнява отново и отново. Обсъждаме с учениците, че в нашия случай точно това е, което искаме / имаме нужда, за да създадем тази игра.

[Окончателен код]




**[Пълна версия]**

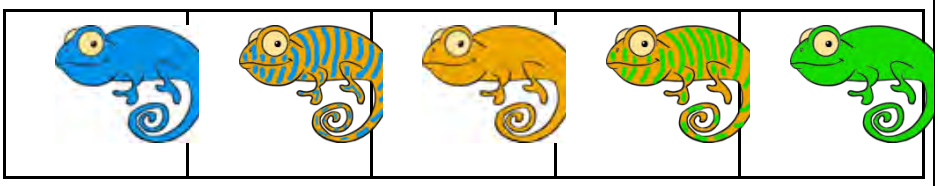
[Стъпка 1]

Молим учениците да редактират фона на сцената, така че той да е разделен на три части от един и същи цвят, всяка от които представлява различно място: син цвят за морето, пясъчен цвят за плажа и зелен за гората. Те могат да добавят други предмети, за да направят фона по-реалистичен като: вълни, черупки, пясъчни замъци, слънчеви чадъри, дървета и др ... но трябва да внимават

добавените предмети да не са по-големи от самия главен герой, защото в в този случай персонажът няма да докосне нито един от трите цвята и функцията за засичане на Snap няма да може да разпознае в коя част от сцената е героят.

[Стъпка 2]

Те трябва да нарисуват хамелеон и да нарисуват кожата му в пет различни комбинации, представящи позицията му на сцената:



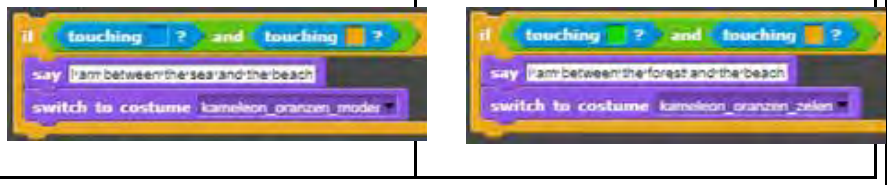
[Стъпка 3]

Първо трябва да накарат хамелеона си да се движи в четири посоки с помощта на клавиши. Те могат да изберат своя собствена комбинация от клавиши (напр. Клавиши със стрелки или WASD). На този етап предполагаме, че те знаят как да го направят от някаква друга дейност. Трябва да предупредим учениците да не забравят, че персонажът може да се измести от сцената, ако не използваме подходящ блок при програмиране на движение (bounce if on edge block).

За да направим движението на хамелеона малко по-реалистично, искаме той да се обърне наляво или надясно към хоризонталната посока, пред която сме изправени (използвайте а *point in direction* block).

| | |
|---|---|
| | [Стъпка 4]<br>Запознаваме учениците с концепцията за характер, усещащ цвета (цветовете), който той докосва. С блока “touching color?” можем да получим информация под формата на булеви стойности - True or False ако той докосва един или дори няколко цвята в даден момент. Тъй като получаваме булева стойност от този блок, можем да я използваме в главата на изречението If, където е решено дали ще изпълняваме команди, изброени в тялото му или не.<br>След това обсъждаме с учениците какви са различните позиции на хамелеона на сцената и как можем да ги изразим с докосващ цвят? блок.<br>Бързо откриваме, че има пет:<br>1. Той е изцяло от синята част -> Touching color [blue]?<br>2. Той е между синята и пясъчната част -> Touching color[blue]? AND Touching color [sand]?<br>3. Той е изцяло от пясъчната част -> Touching color [sand]?<br>4. Той е между пясъчната и зелената част -> Touching color[sand]? AND Touching color [green]?<br>5. Той е изцяло от зелената част -> Touching color [green]?<br>Когато той докосва определен цвят (цветове), ние трябва да променим външния му вид и също така да го накараме да каже къде се намира. Можем да променим външния вид на спрайта, като превключваме между неговите костюми. Това се прави с блока *Looks/switch to costume[option],* където избираме кой от възможните костюми искаме да покажем. За да накараме хамелеона да говори, използваме блока *Looks/say[text]*.<br>Първо се грижим за по-простите ситуации, при които хамелеонът е изцяло в една и съща цветна част на сцената: |


След това формираме логически израз с използването на логически оператор И, защото искаме да проверим дали хамелеонът докосва два цвята едновременно:




Ако комбинираме условните изречения по-горе и ги сложим под блока за събитие *When Green Flag clicked*, забелязваме, че тези условия ще бъдат проверени точно веднъж. Помагаме им да забележат, че тъй като контролираме движението на главния герой, позицията на хамелеон ще се променя през цялото време по време на играта. Ето защо трябва постоянно да проверяваме тези условия не само веднъж, но буквално през цялото време!

[Стъпка 5]

За ситуации, когато трябва да изпълним определени команди за цялото изпълнение на програмата, която използваме - цикъл forever. Всичко, написано под тялото на цикъла forever, ще се изпълнява отново и отново. Обсъждаме с учениците, че в нашия случай точно това е, което искаме / имаме нужда, за да създадем тази игра.

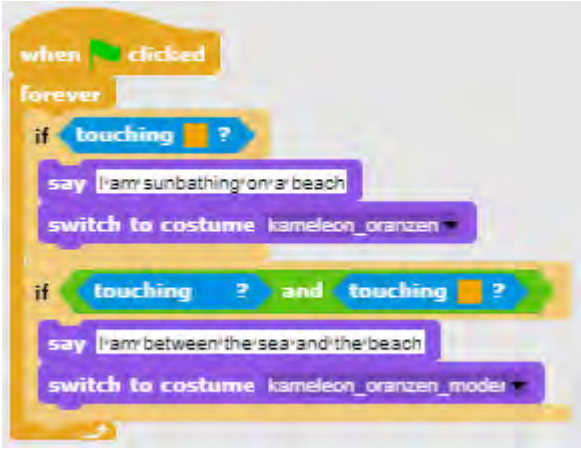
[Окончателен код]


**[Учениците коригират кода]**

За да опростим тази дейност, можем предварително да подготвим част от кода в шаблонния файл и да инструктираме учениците да го попълнят.

Учениците, които следваха предложената учебна пътека, вече научиха за преместването на обекта с ключове. Така че можем да включим кода за движение в шаблонния файл. Те могат да променят настройките на клавишите от клавишите със стрелки към персонализирано подреждане (например WASD).

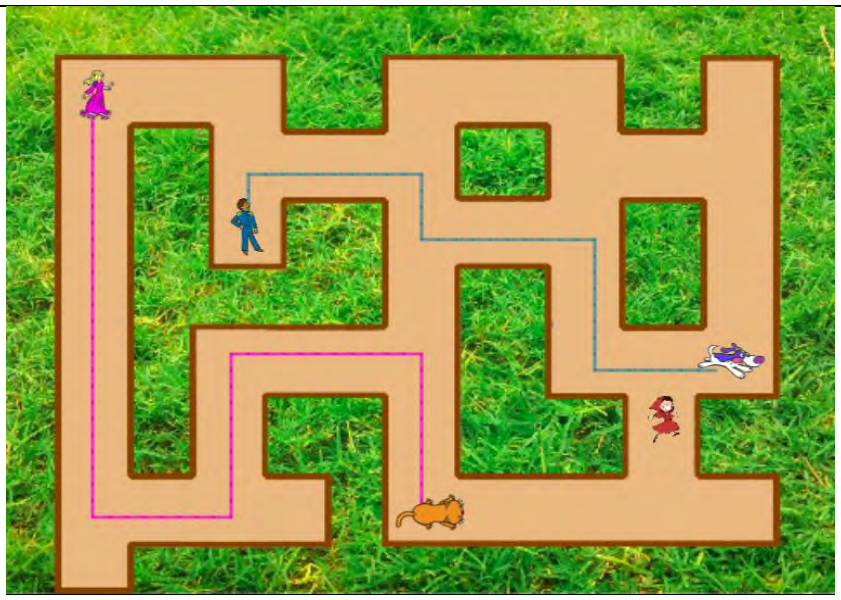
За да им помогнем да разберат понятието цикъл forever и как да го използват за откриване на цвят на фона, можем да включим код за откриване на две ситуации: 1) обектът е изцяло на един цвят, 2)

| | |
|---|---|
| | обектът докосва два цвята едновременно. Инструктирахме ги да попълнят кода за всеки случай.<br>Предложен шаблон за код:<br>  |
| **Инструменти и ресурси за учителя** | • Цялата дейност в Snap!:<br>Basic:<br>https://snap.berkeley.edu/project?user=zapusek&project=chameleon_simple<br>Full:<br>https://snap.berkeley.edu/project?user=zapusek&project=chameleon<br>• Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.<br>• Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Ресурси / материали за учениците** | • Шаблон в Snap!:<br>https://snap.berkeley.edu/project?user=zapusek&project=chameleon_template<br>• Полуготов файл в Snap!:<br>https://snap.berkeley.edu/project?user=zapusek&project=chameleon_half_baked<br>• Инструкции за учениците (C4G6_InstructionsForStudent_BG.docx) |

## Учебен сценарий 7 – Помогнете на принца и принцесата да открият техните животни

| | |
|---|---|
| **Учебен сценарий Име** | Помогнете на принца и принцесата да открият техните животни |
| **Предишен опит в програмирането** | Добавяне на текст за спрайт<br>Движение на обект с клавиши със стрелки с помощта на събития<br>Използването на условие за спрайт докосва друг обект.<br>Използване на събития |
| **Очаквани резултати** | Основни очаквани резултати:<br>● Условия за спрайт да докосва даден цвят<br>● Придвижване до определени координати<br>● Молив горе, Молив долу<br>● Цвят на молива<br>Специфични очаквани резултати свързани с алгоритмично мислене:<br>● Учениците използват оператор if за проверка дали даден обект докосва даден цвят<br>● Учениците задават начални координати на спрайт<br>● Учениците използват блокове „молив горе“ и „молив долу“ за изчертаване на линия/пътека на движение.<br>● Учениците променят цвета в ззависимост от двойката, която свързва.<br>● Учениците реализират начално изтриване на всички очертавания на сцената от предишно изпълнение на програмата. |
| **Цел, задачи и кратко описание на дейностите** | Кратко описание: Момичето трябва да помогне на принцесата да намери котката си, а принцът да намери кучето си. Тя прави това, като отива при принцесата и й показва, с чертане на линия, пътя до котката си; подобно Момичето показва на принца пътя към кучето |

| | |
|---|---|
| | му. По този начин Момичето трябва да избягва срещата между животни, така че пътищата им да не се пресичат.<br>Задачи: В първата стъпка учениците трябва да изберат подходящия фон (лабиринт). Те добавят пет спрайта в лабиринта - техния спрайт (момиче), принцеса, принц, котка и куче. След това те програмират движението с ключове (с използване на събития) за момичето, където трябва да обърнат внимание, че спрайтът не ходи по тревата. По-късно те програмират рисуване с писалка и промяна на цвета на писалката със събития. Те също трябва да програмират стартовото събитие, което изчиства пътя и момичето дава инструкции.<br>**Цел: Учениците ще бъдат въведени в очертаване на движение на героя с клавиши. Освен това те ще се научат как да използват условни блокове и условия, за да предотвратят движеието по целия екран.** |
| **Продължителност** | 30 минути |
| **Стратегия и методи на обучение** | Активно обучение, обучение, основано на игрови дизайн, решаване на проблеми |
| **Форми на обучение** | Фронтално обучение<br>Индивидуална работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)<br>Първоначално се дава на учениците:<br>● Фон<br>● Спрайт на момиче<br>● Код на движение за една посока<br>Момичето решава да помогне на принцесата да намери котката си, а принцът да намери кучето си, като показва (рисува) пътя към техните животни. За да се избегне объркване, пътеките трябва да са с различни цветове и да не се пресичат. |



[Стъпка 1]

Молим учениците да редактират фона на сцената - лабиринт. За прилагане на “if touching color” („ако докосваш цвят“) или фонът (трева) трябва да е монохромен, или пътеката трябва да има монохромна рамка, както в нашия случай. За да избегнем тези „проблеми“ с намирането на подходящ фон, ние им даваме този фон.

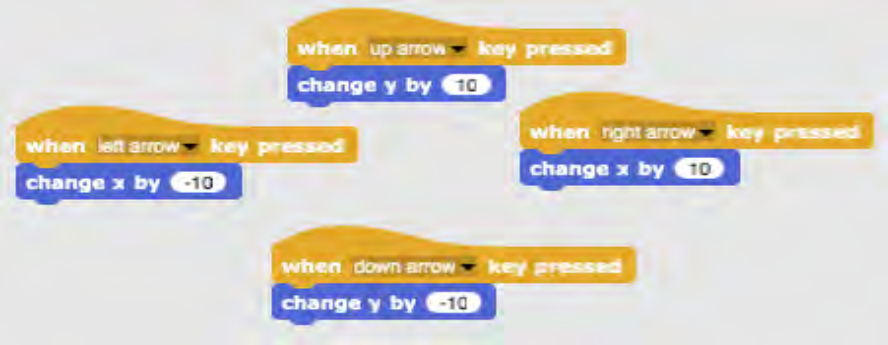
[Стъпка 2]

Учениците вече имат момичешкия спрайт в началото. Трябва да намерят още четири спрайта и да ги сложат в лабиринта. За всички спрайтове те трябва да зададат подходящия размер (който е по-малък от ширината на пътеките в лабиринта. За всеки спрайт те използват кода:


Препоръчителният размер за момичето е 8%, другите спрайтове могат да бъдат по-големи.

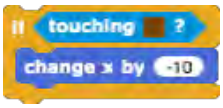
[Стъпка 3]

След това те трябва да направят движението на момичето в четири посоки с помощта на клавиши. Предполагаме, че те вече знаят как да направят това от предишни дейности. Както и да е, ние им даваме кода за едно направление, което им помага да направят още три.




[Стъпка 4]

В следващата стъпка те трябва да предотвратят движението на момичето през поляната. Те правят това, като добавят условен блок, ако докосват кафяв цвят. Ако момичето докосне кафявия цвят (края на пътеката), тя се движи за 10 крачки назад. Не виждаме тези две стъпки и все едно момичето остава на същото място. Това е код за придвижване надясно, така че 10 стъпки назад означава промяна на x с -10.




Те добавят този код под предишния код, напр. за стрелка надясно:


Подобно трябва да се направи и за другите три направления.

[Стъпка 5]

След това програмират рисуване. Те правят това чрез блокове pen up и pen down, използвайки събития *when key pressed*.

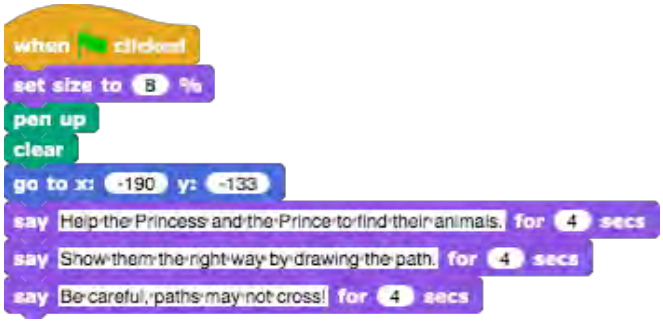

Когато се натисне бутон „D“ и момичето се премести, тя чертае линия. Когато се натисне бутон „E“, чертежът спира.

По същия начин те задават цвят на писалката, като натискат клавиша.

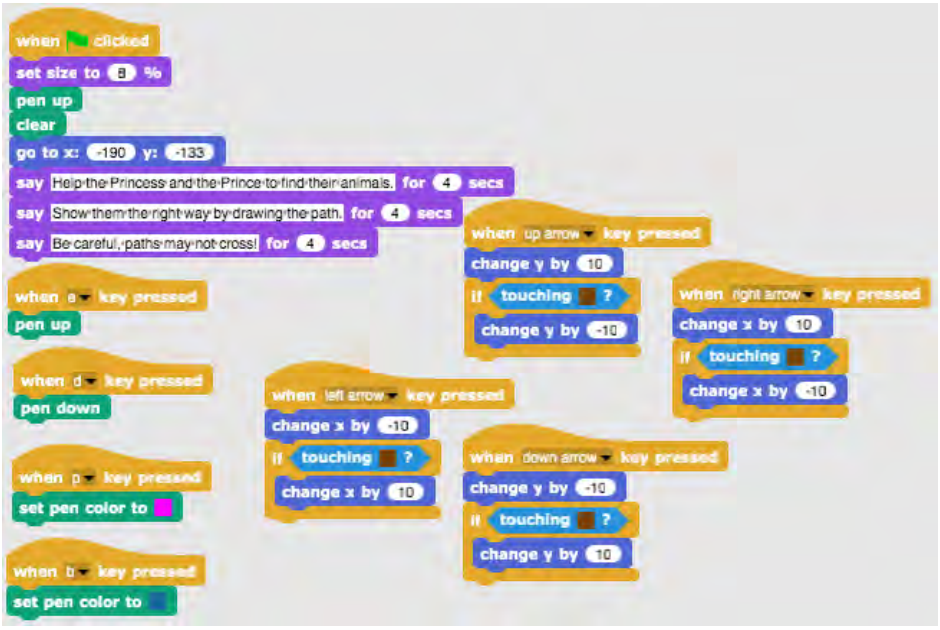

[Стъпка 6]

Накрая те програмират при щракване на събитие със зеленото знаме, където учениците добавят някои инструкции, които момичето казва в началото.

Когато играете играта, спрете я и я играйте отново, учениците ще видят, че е добре да добавите следните блокове: pen up (в случай, че е останала надолу от предишната игра), clear (изчиства пътя от предишната игра) и преместване до x , y (момичето винаги започва от тези координати, които са вътре в пътеката, а не на тревата).




За да определим началните координати за момичето, хващаме момичето с мишката и го пускаме там, където искаме да започне. След това щракваме върху Motion блок, където можем да намерим x позиция и y позиция. Чрез щракване върху x позиция откриваме x позицията на момичето, подобно на y.

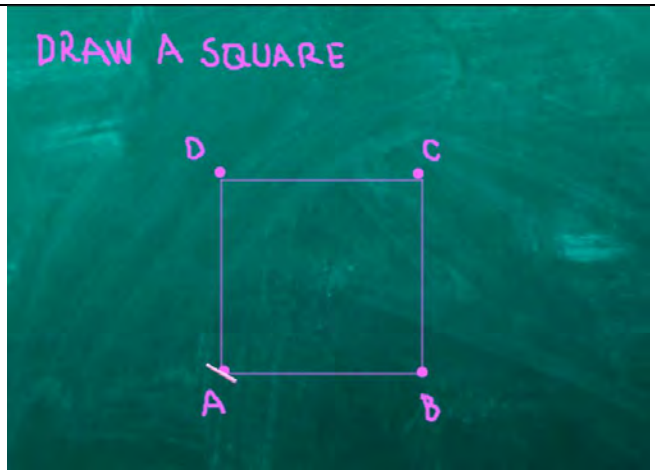
| | |
|---|---|
| | [Окончателен код] Момиче<br>  <br><br>Например Принцесата<br>  <br> |
| | [Допълнителни задачи]<br>Учениците могат да добавят допълнителни задачи според техните желания или могат да следват задачите по-долу:<br>• Задайте начални координати за принца и принцесата и напишете код за тяхното движение. Задайте подходящия размер за тях. Те трябва да нарисуват път към своите животни.<br>• Добавете друг спрайт (животно) за момичето.<br>• Всеки спрайт трябва да рисува с различен цвят.<br>• Регулирайте първоначалните инструкции.<br>• Добавете инструкции за преместване на спрайт и рисуване, като щракнете върху спрайт. Напр. принцесата казва: „Преместете ме с натискане на клавишите W, S, A и D. Начертавам пътеката с натискане на клавиша 3. Спирам да рисувам с натискане на клавиша 4. Помогнете ми да намеря котката си!“ |

| | |
|---|---|
| **Инструменти и ресурси за учителя** | • Цялата дейност в Snap!: https://snap.berkeley.edu/project?user=mateja&project=Helping%20Prince%20and%20Princess%20to%20find%20their%20animals<br>• Дейност в Snap! С допълнителните задачи (възможно решение): https://snap.berkeley.edu/project?user=mateja&project=Helping%20Prince%20and%20Princess%20to%20find%20their%20animals%20%2B%20Add.%20Task<br>• Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.<br>• Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Ресурси / материали за учениците** | • Полуготов файл за работа в Snap!: https://snap.berkeley.edu/project?user=mateja&project=Helping%20Prince%20and%20Princess%20to%20find%20their%20animals%20-%20Part<br>• Указания за учениците (C4G7_InstructionsForStudent_BG.docx) |

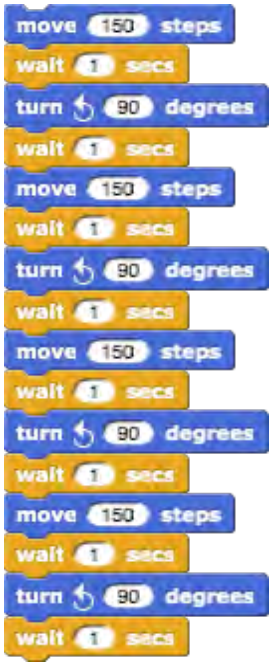
## Учебен сценарий 8 – Рисуване с креда (тебишир)

| | |
|---|---|
| **Учебен сценарий Име** | Рисуване с креда (тебишир) |
| **Предишен опит в програмирането** | • Добавяне на текст към спрайт<br>• Рисуване с молив (pen up, pen down, set color)<br>• Преместване на определен брой стъпки<br>• Използване на цикъл<br>• Използване на събития |
| **Очаквани резултати** | Основни очаквани резултати:<br>• Цикъл repeat<br>• Завъртане на 90 градуса<br>• Определяне посока на движение (Point in direction)<br>• Смяна фон на сцена<br>Специфични очаквани резултати, свързани с алгоритмично мислене:<br>• Ученикът използва цикъл repeat когато се повтарят едни и същи блокове няколко пъти<br>• Ученикът използва завъртане на 90 градуса когато изчертава различни фигури (квадрат, правоъгълник, буква “Т”)<br>• Ученикът описва заначението на установяване на посока на движение 90<br>• Ученикът сменя фон на сцена при използване на събитието *when a key ….. is pressed* |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Играчът получава три различни фона и трябва да свърже точки в три различни форми - квадрат, правоъгълник и буква „Т“.<br>**Задачи:** Учениците избират фона „дъски” и започват с рисуване на квадрат. Тяхната изходна позиция е точката „А“. Когато рисуват |

| | |
|---|---|
| | квадрат, те повтарят определени стъпки 4 пъти, така че вместо да пишат същия код 4 пъти, те могат да използват повторение на цикъла 4 пъти. След това те нарисуват правоъгълник, също с използване на повторение на цикъл, този път повторете 2 пъти. В последната си задача те трябва да свържат точки във формата на буквата „Т“, където трябва да открият броя на стъпките. Те могат да използват повторение на цикъла, когато е възможно.<br>**Цел: Учениците ще бъдат запознати с рисуването на различни фигури с код. Те ще се научат да използват повторение на цикъл, за да съкратят кода и да променят фона.** |
| **Продължителност** | 60 минути |
| **Стратегия и методи на обучение** | Активно обучение, обучение, основано на игрови дизайн, решаване на проблеми |
| **Форми на обучение** | Фронтална работа<br>Индивидуална работа / Работа по двойки |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)<br>Първоначално се дава на учениците:<br>● Три фона с всички точки, които трябва да свържат<br>● Крейда спрайт<br>Кредата трябва да нарисува квадрат, правоъгълник и да свърже точки във формата на буква „Т“, но не знае как да се движи и как да се завърти. Напишете код и покажете на тебешира как се прави!<br>[Стъпка 1] |




Учениците започват с този фон. Те пишат код за рисуване на квадрат. Започвайки от точка "A", те преместват X стъпки към точка "B", завъртат 90 градуса вляво, преместват X стъпки до точка "C", завъртат 90 градуса вляво, преместват X стъпки до точка " D ", завъртете 90 градуса вляво, преместете X стъпки до точката„ A "(и завъртете 90 градуса вляво).




Използвайки *turn 90 degrees* е най-лесният начин, тъй като винаги можем да използваме завъртане на 90 градуса (зависи само дали искаме да завием наляво или надясно). Използвайки Using *point in direction 0, 90, 180, -90* е друга опция, но е малко по-сложна, защото

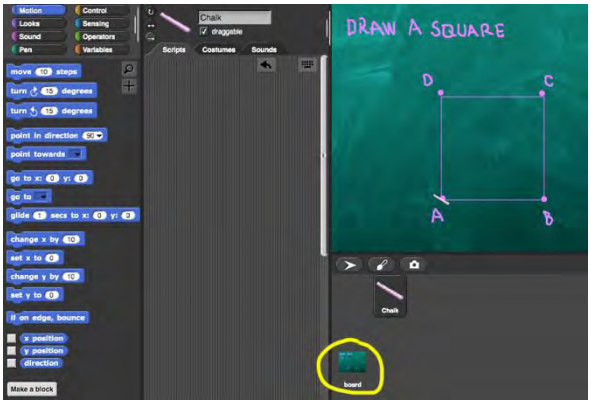
| | |
|---|---|
| | трябва да отделим 4 възможности и не можем да използваме цикъл *repeat*.<br>*Блокът Wait 1 secs* блок се добавя само за да видите чертежа / всички стъпки. Без този блок целият код се случва за секунда. Учениците трябва да го пробват без този блок, за да разберат значението му.<br>Питаме ученика как би съкратил кода, ако е възможно. Има ли част от повторенията? Отговорът е да. Вместо да пишем същия код 4 пъти, при програмирането използваме цикъл *repeat*.<br>  <br>Ако искаме да видим какво нарисуваме, трябва да поставим блок *pen down* преди цикъла *repeat*.<br>  <br>Ако искаме тебеширът да не се върти при завъртане, щракваме върху блока don't rotate in direction.<br>  <br>[Стъпка 2]<br>За активиране на кода учениците използват блока за събития, напр. *when S key is pressed*. Те могат да изпозлват също *set pen color*, и както вече знаят от предходната тема да изпозлват следните блокове *pen up* (в случай, че е останал настрана от предишното проиграване), *clear* (изчиства рисунката от предишното възпроизвеждане) и *go to x, y* (тебеширът винаги започва от тези координати).<br>Понякога се случва да спрем програмата по време на пиесата и след това спрайтът да се завърти в „странна посока". Това е проблем |

при стартиране на играта отново, ако спрайтът се завърти погрешно, той ще отиде например надолу, а не вдясно на първата стъпка. За да избегнем този проблем, добавяме блок *point in direction 90*.

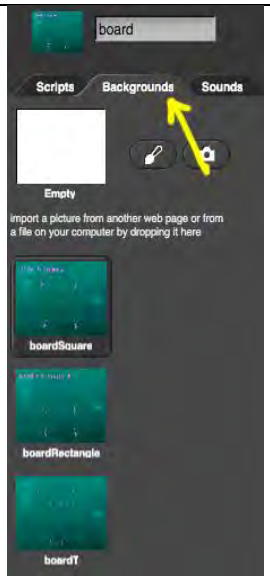

[Стъпка 3]

След изчертаването на квадрат искаме да нарисуваме правоъгълник. Това означава, че трябва да променим фона. Ще направим това с две стъпки:

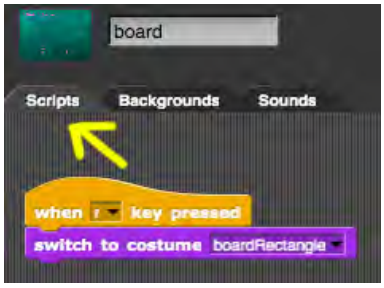
а) Щракваме върху фона (име на дъска, от дясната страна на екрана).



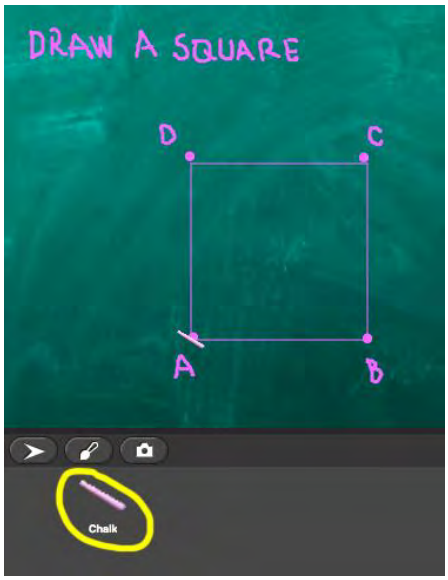

Щраквайки върху *Backgrounds* можем да видим трите фона (*boardSquare, boardRectangle, boardT*), които предварително са създадени за този урок.




За да напишат код, учениците трябва да щракнат върху *Scripts*. За да програмират променящия се фон, те избират блок от Събития - *when R key pressed* и след това *switch to costume boardRectangle*.

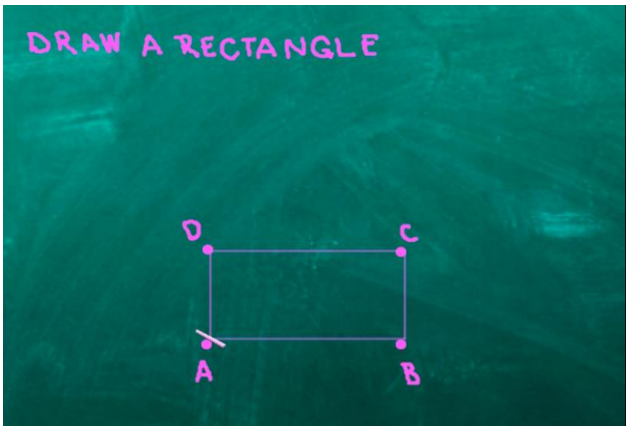



b) Щракваме обратно върху тебешира.

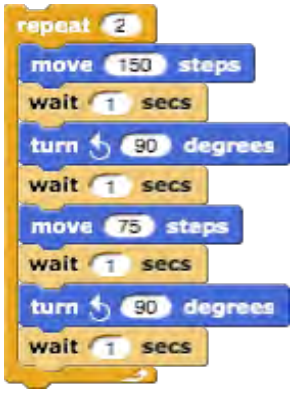


Под кода от [Стъпка 2] учениците добавят блок, където ще казват на играч какво да направи, за да смени фона, което е, „Натиснете клавиша „R“.“


[Стъпка 4]




След натискане на клавиша “R”, фонът се променя на този. Подобно на преди, те трябва да свържат точки и да нарисуват правоъгълник. Учениците могат да копират предишните блокове код и да ги коригират, така че програмата ще нарисува правоъгълник.




Те сменят цикъла *repeat*. Сега този цикъл ще се повтори 2 пъти.

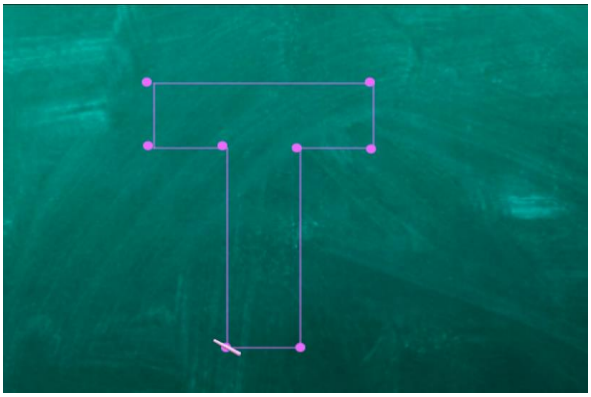
[Стъпка 5]

След изчертаване на правоъгълник учениците ще свържат точки във форма на буква „T“. Това означава, че трябва да сменят фона, така че в тази стъпка те действително повтарят [Стъпка 3], просто сменят буквата („T“) и костюма (boardT):

a) Те щракват върху фона (наименован board, от дясната страна на екрана), където пишат код за промяна на фона. Те ще направят това с *when T key pressed* и след това *switch to costume boardT*.


b) Те щракват обратно върху тебешира и под кода от [Стъпка 4] добавят блок, където ще казват на играч какво да направи, за да смени фона, което е, натиснете клавиша „Т“.


[Стъпка 6]



След натискане на клавиша “Т”, фонът се променя на този. Подобно на преди, те трябва да свържат точки и да нарисуват буква „Т“. Учениците могат да копират предишните блокове от кода и да ги коригират.

Учениците ще трябва да сменят началните координати, които не са същите като преди. Те вече знаят как да определят правилните координати от предишната дейност.

След това те пишат код за изчертаване на буква „Т“. Те трябва да открият броя на стъпките. Едно от възможните решения е:


[Стъпка 7]

Тъй като сме сменили фона, не можем да се върнем на първия фон, за да нарисуваме квадрат. Така че учениците ще трябва да добавят един последен код. Те повтарят [Стъпка 3/5].

а) Те щракват върху фона (наименована board, от дясната страна на екрана), където пишат код за промяна на фона. Те ще направят това с *when S key pressed* и след това *switch to costume boardSquare*.

b) Те щракват обратно върху тебешира и под кода от [Стъпка 6] добавят блок, където ще казват на играч какво да направи, за да смени фона, което е, натиснете клавиша „S“.


[Финален код]

| | |
|---|---|
| | [Допълнителни задачи]<br><br>Учениците могат да добавят допълнителни задачи според техните желания или могат да следват задачите по-долу:<br>● Добавете нов фон и нарисувайте няколко точки.<br>● Напишете код, който свързва точките. Можете да нарисувате фон или да използвате даден. |
| **Инструменти и ресурси за учителя** | ● Цялата дейност в Snap!: https://snap.berkeley.edu/project?user=mateja&project=Drawing%20with%20a%20chalk<br>● Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.<br>● Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Инструменти и ресурси за учениците** | ● Полуготови ресурсни материали в in Snap!: https://snap.berkeley.edu/project?user=mateja&project=Drawing%20with%20a%20chalk%20-%20Part<br>● Инструкции за учениците (C4G8_InstructionsForStudent_BG.docx) |

## Учебен сценарий 9 - Прибиране на боклука и почистване на парка

| | |
|---|---|
| **Учебен сценарий Име** | Прибиране на боклука и почистване на парка |
| **Предишен опит в програмирането** | • Задаване на начални координати<br>• Задаване размер на спрайта<br>• Добавяне на текст към спрайта<br>• Придвижване на обект със стрелки, използвайки събития<br>• Използване на условия за проверка дали даден обект докосва друг обект |
| **Очаквани резултати** | Общи очаквани резултати:<br>• Променливи<br>• Показване и скриване на спрайтове<br>• Дублирани спрайтове<br>• Дублиран блок от код<br>• Условия<br>Специфични очаквани резултати за развитие на алгоритмично мислене:<br>• Ученикът използва променлива за преброяване на събраните отпадъци<br>• Ученикът използва скрит спрайт, когато е докоснат спрайт и показва спрайт в началото<br>• Ученикът знае как да дублира спрайт (от една бутилка до например 4 бутилки)<br>• Ученикът знае как да дублира блок код (от спрайт на бутилка на хартиен спрайт)<br>• Ученикът знае как да използва условни условия за проверка дали е показан спрайт и дали всички боклуци са взети |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Паркът е пълен с боклук и момичето решава да го почисти. Когато събере всички боклуци, тя ги хвърля в кошчето.<br>**Задачи:** Учениците започват с задаване на начални координати за момичето. Играта приключва, когато момичето събере всички боклуци и ги сложи в кошчето. За целта учениците ще трябва да използват променливи за преброяване на точки (1 събрано кошче = 1 точка). Когато момичето докосне кошчето, го вдига, кошчето |

| | |
|---|---|
| | се скрива и броят точки се увеличава за 1. Когато вземе цялото кошче, тя отива до кошчето. Ако тя не вземе целия боклук и отиде по-рано в кошчето, кошчето казва да се върне, когато вземе цялото кошче.<br>**Цел: Учениците ще научат как да използват променливи и как да дублират блок код или дори цял спрайт.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно обучение, обучение, основано на дизайн на игри, решаване на проблеми |
| **Форми на обучение** | Фронтално обучение<br>Индивидуална работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)<br>Първоначално се дава на учениците:<br>● Фон<br>● Спрайт за момичета (с кода за движение), спрайт за бутилки, спрайт за хартия и спрайт за боклук<br>Момичето иска да се разходи и да се наслади на деня си в парка. Когато идва там, тя вижда, че паркът е пълен с боклук. Тя решава да вземе всички боклуци. Когато го направи, най-накрая може да легне и да се наслади на слънчевия ден в чист парк.<br>  |

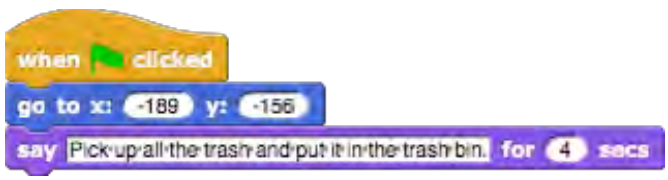
[Стъпка 1]

Даден е фонът, а също и спрайт-момиче с код за движение с клавиши и условен блок с условие за докосване на кафявата линия.





Учениците трябва да зададат началните координати за момичето с блок x, y. Координатите се избират сами, важно е само те да са по пътя. Учениците вече знаят как да задават координатите от предишни дейности. Те също така добавят някои инструкции. Например:



[Стъпка 2]

За преброяване на броя на боклука, който момичето е взело, ще използваме променливи.

Какво е променлива? Променливата е като кутия, в която съхраняваме някаква информация.

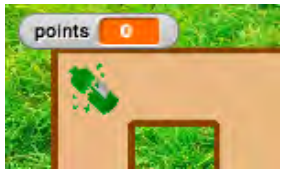
В нашия случай можем да видим нашата променлива като поле, наречено точки. Когато момичето вземе боклук, кошчето се съхранява в променлива *points*. Тази променлива отчита колко боклук е избрало момичето.

Как да направим променлива?



Избираме оранжев блок *Variables*, след това щракнете върху бутона *Make a variable*, запишете име на променлива (*Variable name) и щракнете върху* OK. Появяваше блок *points* appears.


Ако квадратчето е отметнато, променливата с нейната стойност ще се вижда на екрана:




В началото на играта стойността на променливата трябва да бъде 0, тъй като няма прибран боклук. Под кода от [Стъпка 1] ученикът добавя блок *set __ to 0. Чрез щракване върху падащото меню те избират подходяща променлива, която е points*.

[Стъпка 3]

Учениците пишат код за бутилка. Идеята е, че спрайтът изчезва (което означава скрий), когато докосне момичето.

Така кодът ще започне, когато спрайтът докосне момичето. Тогава трябва да помислим в кой случай тя взема боклука. Ако казахме, че кошчето се скрива, когато е взето, можем да го вземем само ако все още е там = е показано. Ако спрайтът (бутилката) все още е там, ние го вдигаме „и го поставяме в полето с променливи“. Преди имахме 0 елемента в променливата *points*, сега имаме 1. Виждаме, че като вземем боклука, променяме броя на променливите (*points*) с 1, което е, увеличете с 1. Когато боклукът бъде взет, ние го скриваме.


Сега можем да тестваме дали нашият код е правилен.

Щракаме върху зелено знаме и взимаме бутилката. Бутилката трябва да изчезне и броят точки трябва да е 1. След това искаме да играем отново и отново щракваме върху зеления флаг. Какво става? Къде е бутилката сега?

Бутилката е скрита, ние я скрихме преди. Така че в началото на играта трябва да програмираме, така че бутилката да се показва. Правим това, като изберем блок Show (покажи).

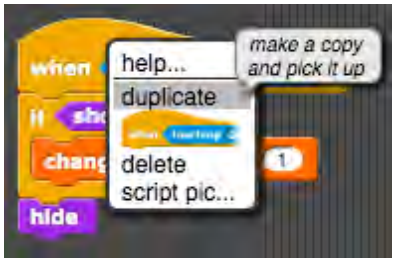

[Стъпка 4]

Сега учениците искат да имат повече бутилки в играта си, за да могат лесно да дублират своя спрайт. Те щракват с десния бутон върху спрайта и избират duplicate (дублирай).


Сега те просто щракват с мишка върху новата бутилка и я плъзгат някъде вътре в лабиринта. Те могат да повторят тази стъпка и да дублират бутилката отново.

[Стъпка 5]

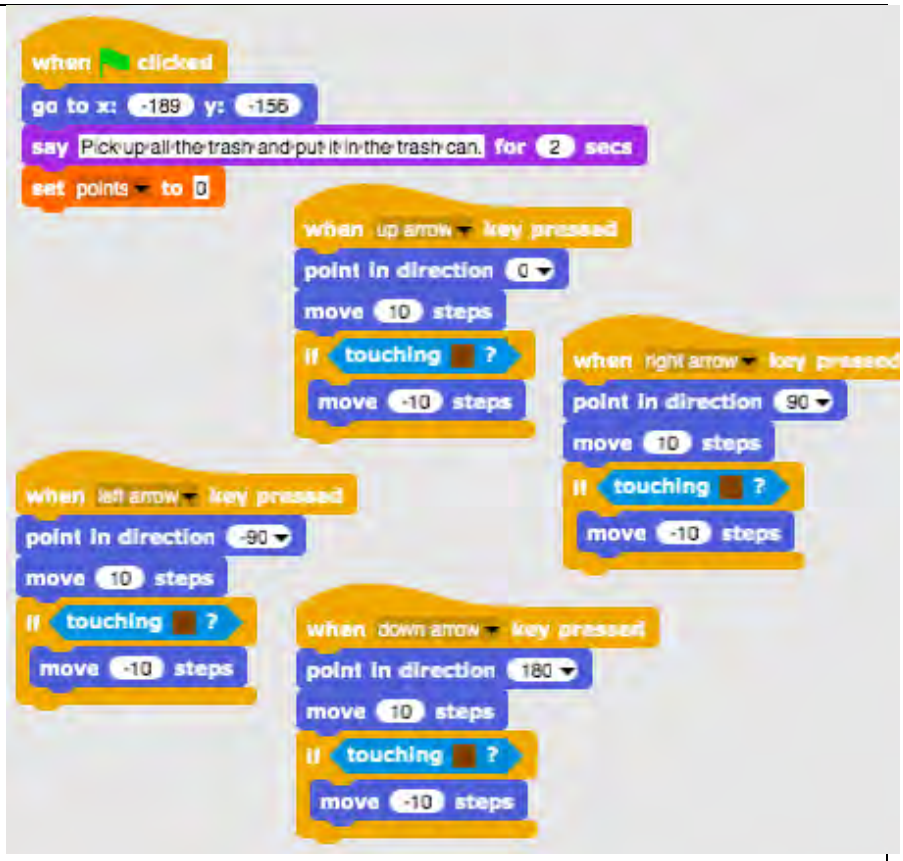
Сега учениците искат да имат същия код за хартиения спрайт. Те могат да дублират кода на бутилката, като щракнат с десния бутон върху блока с код:




И го пуснете в хартиения спрайт, като щракнете с мишката върху спрайта за хартия.


Те повтарят тази стъпка, за да дублират блока *when green flag clicked – show*.

| | |
|---|---|
| | Те могат също да повторят [Стъпка 4] и да дублират целия хартиен спрайт, за да има повече боклук от хартия в лабиринта.<br>[Стъпка 6]<br>Последното нещо, което учениците трябва да направят, е да напишат код за кошчето. Кофата за спрайт вече е дадена, те могат да я преместят навсякъде вътре в лабиринта.<br>Също така този код ще се активира, когато момичето го докосне.<br>Кошчето ще трябва да провери дали всички боклуци са взети. Благодарение на променливите точки това ще бъде лесно да се направи. Да приемем, че имаме 8 спрайта в играта, така че учениците трябва да проверят дали броят на точките е равен на 8. Ако е, това означава, че всички боклуци са взети, в противен случай не е така. Те ще използват инструкция if, за да програмират това и ще добавят текст, за да кажат на играча дали е взел всички боклуци или не.<br>  <br><br>[Финален код]<br>Момиче |




Бутилка/ Хартия


Кошче

| | |
|---|---|
| | [Допълнителни задачи]<br>Учениците могат да добавят допълнителни задачи според техните желания или могат да следват задачите по-долу:<br>● Добавете друг вид отпадъци (например биоотпадъци).<br>● В кошчето пише напр. „Взехте X бутилки, Y хартии и Z дини“.<br>● Ако играч вземе цялото кошче, кошчето казва: „Поздравления! Взехте всички боклуци! “<br>● Ако даден играч не вземе цялото кошче, кошчето му казва кой боклук не е бил взет, напр. „Не сте взели всички бутилки. Не сте взели всички дини. и „Върнете се, когато вземете всички боклуци“. |
| **Инструменти и ресурси за учителя** | ● Цялата дейност в Snap!: https://snap.berkeley.edu/project?user=mateja&project=Picking%20up%20trash%20and%20cleaning%20the%20park<br>● Допълнителни задачи (възможни решения): https://snap.berkeley.edu/project?user=mateja&project=Picking%20up%20trash%20and%20cleaning%20the%20park%20%2B%20Add.%20Task<br>● Lajovic, S. (2011). *Scratch. Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.<br>● Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Инструменти и ресурси за учениците** | ● Полуготови файлове с ресурсни материали в Snap!: https://snap.berkeley.edu/project?user=mateja&project=Picking%20up%20trash%20and%20cleaning%20the%20park%20-%20Part<br>● Инструкции з аученициte (C4G9_InstructionsForStudent_BG.docx) |

## Учебен сценарий 10 - Хранене на котките

| | |
|---|---|
| **Учебен сценарий Име** | Хранене на котките |
| **Предишен опит в програмирането** | • Условни блокове (if, if-else blocks)<br>• Извеждане на текст (block say) |
| **Очаквани резултати** | Основни очаквани резултати:<br>• Задаване и учеличаване стойност на променлива,<br>• присвояване стойност на променлива вътре / извън цикъла,<br>• цикъл for (repeat n times),<br>• случайни числа,<br>• слепване на стрингове,<br>• логически и аритметични оператори,<br>• вход<br>Специфични цели на обучение ориентирани към алгоритмичното мислене:<br>• Ученикът разпознава ситуацията за използване на повторение n цикъл пъти,<br>• Ученикът прави разлика между присвояване на стойността във всяка итерация на цикъла и веднъж преди цикъла.<br>• Ученикът използва блок за въвеждане, за да получи номера от играч,<br>• Ученикът знае как да използва аритметични оператори, за да генерира правилния отговор,<br>• Ученикът използва if - else, за да провери правилността на въведеното от играча и дава подходящ отговор,<br>• Ученикът знае как да използва променлива, за да брои правилни отговори. |
| **Цел, задачи и кратко описание на дейностите** | Кратко описание: Програмирайте игра, в която играчът ще трябва да извърши десет изчисления за умножение и да преброи верните отговори.<br>Задача: Програмирайте дейността, при която пазачът на приюта Марта многократно ще пита играча за броя на котките, които може да храни в определена стая. Броят зависи от броя и размера на |

| | |
|---|---|
| | купите. За всяка стая тези два номера трябва да бъдат разпределени произволно. Трябва да имаме и брояч, който да отчита правилните отговори. Първо пазителят на приюта трябва да обясни заданието за играча и след това играта започва. Играта приключва, когато тя поиска броя на котките 10 пъти. Всеки път тя трябва да даде отговор дали входният номер е правилен или не. След активност тя трябва да обобщи колко успешен е бил играчът, тя казва колко пъти играчът е отговорил правилно и колко пъти е сгрешил.<br>**Учениците ще бъдат запознати с концепцията за многократно присвояване на случайни стойности в цикъл и как тя е различна от тази, когато го правим извън цикъл. Те също така ще научат как да получат, тестват и преброят правилните входове на играчите.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно обучение, съвместно обучение, решаване на проблеми |
| **Форми на обучение** | Фронтално обучение<br>Индивидуална работа / работа по двойки / групова работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)<br>Пазителят на приюта се опитва да нахрани котките си в десет различни стаи. Във всяка стая има произволен брой купи (2 до 10), които имат различни размери (1 до 5), но във всяка стая всички купи са с еднакъв размер. Размерът на купата показва колко котки могат да ядат от нея, например ако размерът на купата е 3, това означава, че 3 котки могат да ядат от нея. Помогнете да намерите броя на котките, които тя може да храни във всяка стая. |

[Стъпка 1]

Първо инструктираме учениците да проектират интересен фон за играта. Ако искаме да спестим време, можем да им го предоставим.

[Стъпка 2]

Трябва да изберем нов костюм за спрайта на костенурките по подразбиране, който ще представлява пазителя на приют за котки.

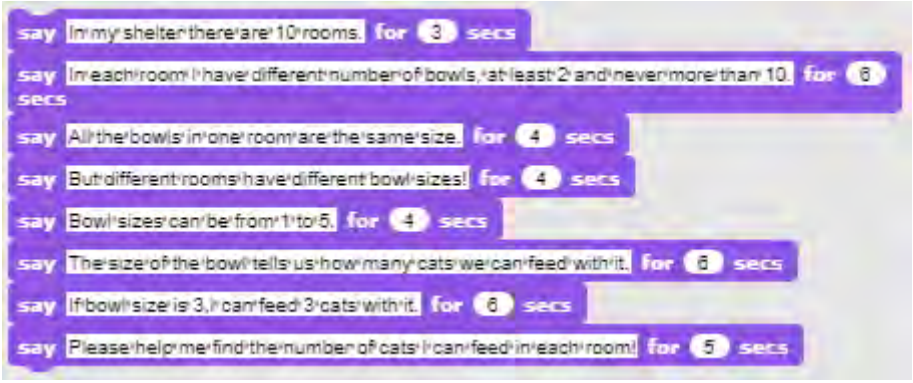
[Стъпка 3]

За да съхраним необходимите стойности, са ни необходими три променливи: 1) за съхраняване на броя на верните отговори, 2) за присвояване на произволна стойност за броя на купите във всяка стая (2-10) и 3) за присвояване на произволна стойност за купата капацитет (1-5). Началната стойност на точния брояч на отговори ще трябва да бъде 0, а другите два не трябва да бъдат инициализирани преди цикъла, защото ние ще им присвоим нови произволни стойности във всяка итерация на цикъла. Ние също искаме да броим стаите, но не се нуждаем от специална променлива, за да го броим.

Ще използваме същата променлива като в цикъл for. Неговият номер ще бъде инициализиран до стойност 1 и след това ще бъде увеличен с 1 за всяка итерация, докато се достигне стойност 10. Това възпроизвежда броенето на стаята.

[Стъпка 4]

След това трябва да програмираме инструкциите за плейъра. Правим това с *Looks / say [string]* и *wait [n] seconds*.



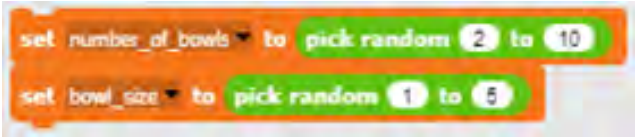

[Стъпка 5]

Обсъждаме с учениците какви са действията, които ще се случат във всяка стая и по този начин ще бъдат еднакви. Това са команди, които ще трябва да бъдат поставени вътре в блока на цикъла, за да бъдат изпълнени във всяка итерация на цикъла.

Първо ще трябва да зададем произволно стойност (1-10) за броя на купите и размера на купата в тази стая (1-5). Тогава ще трябва да попитаме играч колко котки можем да храним в тази стая. Нейният отговор ще трябва да бъде тестван за коректност и ние ще трябва да дадем подходящ отговор и да запомним дали е бил верен (брояч на верните отговори). В края на всяка итерация също ще трябва да увеличим номера на стаята с 1.

[Стъпка 6]

За да присвоим произволно стойностите за броя на купите и техния размер, ще използваме стойност Variables / set [options] с Operators/pick random [n] to [m].

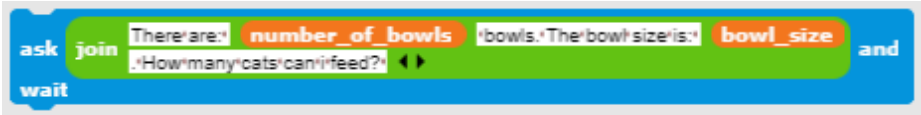


[Стъпка 7]

Искаме да поискаме от играча броя на котките, които можем да храним в блока Sensing / ask [string] и изчакайте, защото в противен случай той ще се покаже за определени секунди и след това ще бъде актуализиран с нов ред текст. По този начин играчите могат лесно да забравят колко купи / размери са в текущата стая. За да направим низ, който ще бъде изграден от комбинация от текст и препратки към променливи, които използваме Operators/join [string1][string2] block. Ще трябва да разширим този блок, така че да отговаря на цялото изречение.

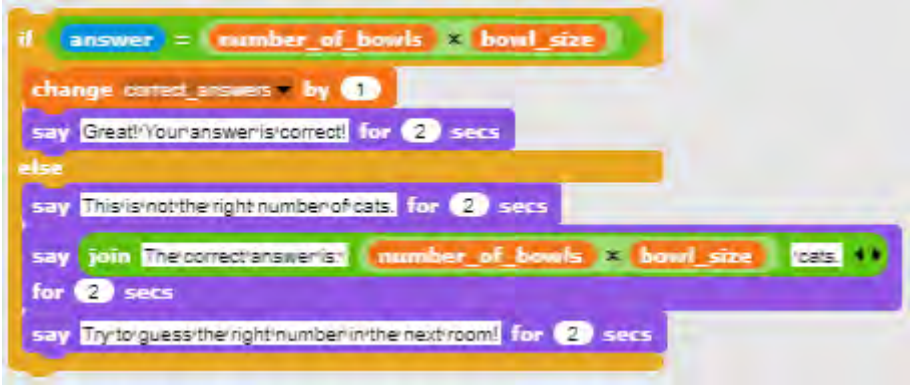

[Стъпка 8]

Трябва да поставим този дълъг низ вътре Sense/Ask [string] и блока wait block за да получим отговор от играча.




[Стъпка 9]

Когато играчът отговори, трябва да проверим верността. Има само две възможни ситуации, играчът може да бъде правилен или грешен, така че ще използваме блока If-Else. Правилният отговор е стойността на умножаване на броя на купите с размера на купата. Трябва да проверим дали отговорът на играча е равен на това число. Ако отговорът е верен, увеличаваме брояча на верните отговори с 1 и даваме отговор. Ако не, ние само даваме отговор. Не е нужно да броим грешни отговори, защото можем да го изчислим от правилния брояч на отговорите.




[Стъпка 11]

Сега трябва да изберем цикъл. Както споменахме по-рано, най-добре е да изберете за цикъл, защото променливата, която се използва за итерация, възпроизвежда преброяването на стаите.

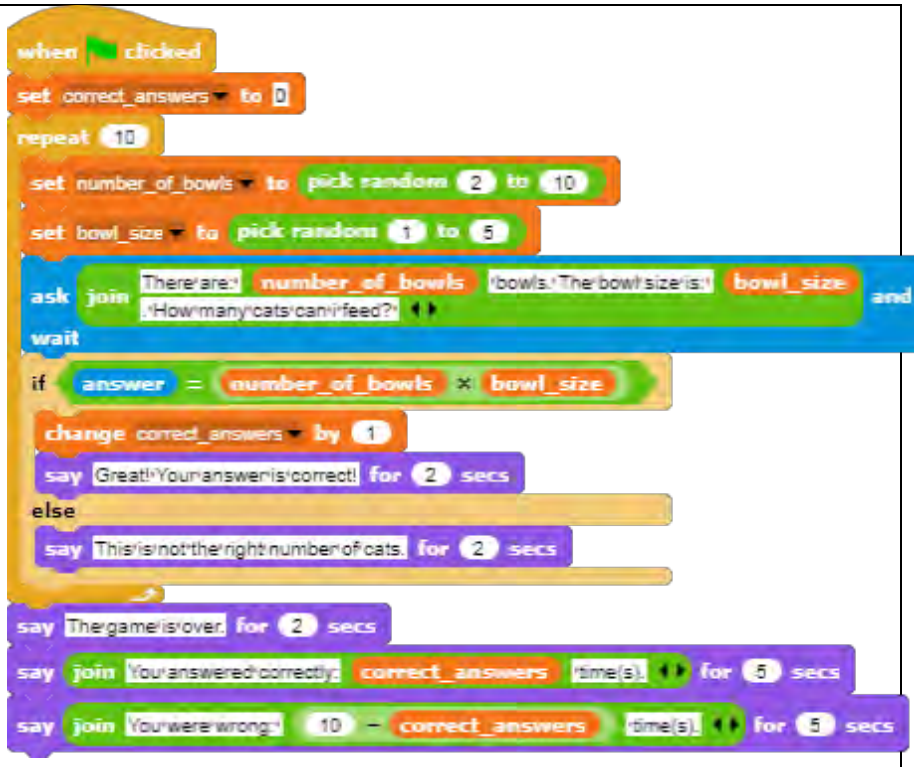
[Стъпка 12]

Когато цикълът спре, играта приключва. Ние предоставяме информация за постиженията на играчите. Броят на верните отговори се съхранява в брояча на верните отговори; броят на грешните отговори може да бъде изчислен.

[Финален код]


[Основна версия на дейността]

За да спестим време, можем да използваме основната версия на сценария. В основна версия са включени всички основни понятия, други функции, описани по-горе, могат да бъдат използвани като по-късни надстройки.

| | |
|---|---|
| |  |
| **Инструменти и ресурси за учителя** | • Цялата дейност в in Snap!: https://snap.berkeley.edu/project?user=zapusek&project=cat_feeding_2<br>• Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.<br>• Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Инструменти и ресурси за учениците** | • Шаблон в Snap!: https://snap.berkeley.edu/project?user=zapusek&project=cat_feeding_template<br>• Инструкции за учениците (C4G10_InstructionsForStudent_BG.docx) |

## Учебен сценарий 11 - Познай броя на котките в приют

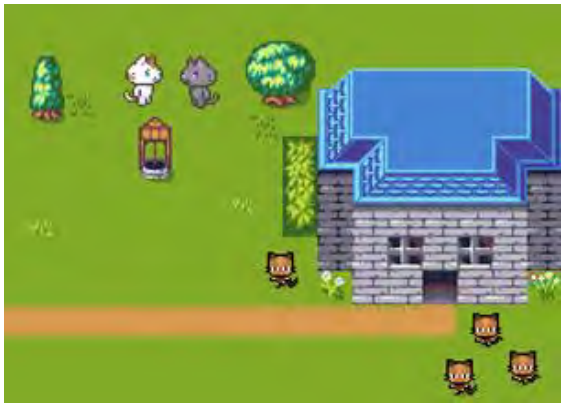
| | |
|---|---|
| **Учебен сценарий Име** | Познай броя на котките в приют |
| **Предишен опит в програмирането** | • Условни блокове (if block)<br>• Извеждане на текст (block say) |
| **Очаквани резултати** | Основни очаквани резултати:<br>• Случайни числа,<br>• Присвояване стойност на променлива,<br>• Вход,<br>• Цикъл repeat until,<br>• Оператори за сравнение,<br>• Брояч<br>Специфични очаквани резултати, ориентирани към алгоритмично мислене:<br>• ученикът присвоява произволна стойност на променливата,<br>• ученикът използва блок за въвеждане, за да получи номера от играч,<br>• ученикът използва повторение до цикъл, за да поиска многократно играч да въведе числото и да извърши тестване на стойност<br>• ученикът извършва тестване на стойност с оператори за изречение и сравнение и дава подходящ отговор,<br>• ученикът задава условието на цикъла за повторение, за да тества дали играта е приключила,<br>• ученикът осъзнава, че не трябва да тества дали играта е приключила, тъй като тя е намерена в състояние,<br>• ученикът прилага брояч за преброяване на предположенията на играчите и използва крайната стойност, за да направи разлика между двата възможни резултата. |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Програма на проста игра, в която в началото произволно число от 1 до 100 ще бъде разпределено на случаен принцип към променлива. Играчът ще се опита да го отгатне, като |

| | |
|---|---|
| | въведе цифри. Тя ще получи отговор, ако входното число ще бъде: повече, по-малко или равно на случайната стойност.<br>**Задача:** Програмирайте приюта за котки Марта да задава на случаен принцип броя на котките, да попита играча за него или името му и след това да му обясни задачата. След това Марта трябва да поздрави играча с неговото / неговото име и след това многократно да поиска номер. Когато играчът въведе своето предположение, тя трябва да отговори: 1) ако входният номер е по-малък от действителния брой, тя казва: „броят на котките е по-голям“, 2) ако входният номер е по-голям от действителния брой, казва тя : „Броят на котките е по-малък“, 3) ако входният номер е правилен, тя казва: „Отлично, познахте правилния номер“. Програмирайте брояч, който ще отчита всеки опит на играч. Когато играчът отгатне правилния номер, трябва да проверите дали броят на опитите е по-малък от 5. В този случай играчът получава котката, в противен случай не.<br>**Цел: Учениците ще бъдат въведени да повтарят до цикъл и как да зададат условието да имплицитно проследява условието, което спира играта. Те също така ще се научат как да използват променливи в различни ситуации: да зададат произволна стойност, като брояч или да получат вход на играчите.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно обучение, съвместно обучение, решаване на проблеми |
| **Форми на обучение** | Фронтално обучение<br>Индивидуална работа / работа по двойки / групова работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)<br>Пазачът на приюта за котки Марта иска да познаете точния брой котки, които тя има в приюта си. Числото може да бъде между 1 и |

100. Играчът въвежда числото, а тя отговаря, ако текущото число е по-малко, повече или равно на правилния брой котки. Ако играч отгатне броя на котките за по-малко от пет опита, тя получава котката, в противен случай тя подканя да играе отново.

[Стъпка 1]

Първата задача е да се направи интересен фон за играта. Учениците могат да го нарисуват сами или да използват безплатни изображения за лиценз от интернет. За да спестим време, можем да подготвим фона предварително.

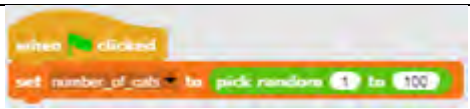


[Стъпка 2]

Трябва да изберем нов костюм за спрайта на костенурките по подразбиране, който ще представлява пазителя на приют за котки.

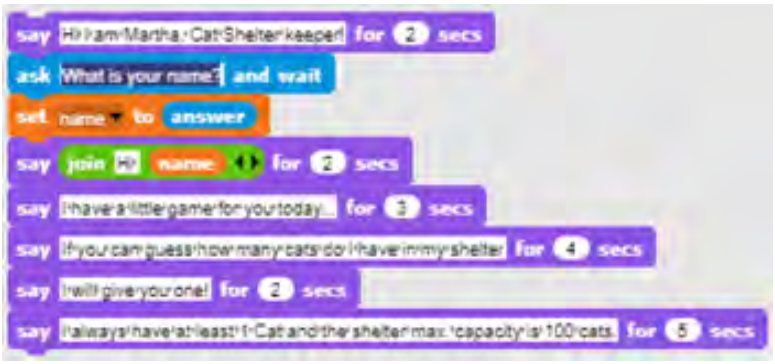
[Стъпка 3]

Обсъждаме с учениците, че тази игра може да бъде интересна, ако я играете повече от веднъж, ако броят на котките е зададен на случаен принцип. За да запазим това произволно число за сравнение на числата за предположения, ние също трябва да го съхраним в променлива. Сега променливите (предполагаме, че все още не познават концепцията за списъци) са единственият начин да запомните определена стойност в Snap. Това трябва да се случи, когато програмата стартира (Event/When green flag is clicked).




[Стъпка 4]

Пазителят на приюта пита играча за името й, за да я поздрави. Това се прави с Sense / ask [низ] и блок за изчакване. Отговорът на играча се съхранява във вградена променлива с име отговор. За да я поздравим, трябва да се присъединим към низа, съхраняван в променливата отговор, с някакъв поздрав. Това се прави с блок Operators / join [string1] [string2]. За да покажем текста, използваме Изглежда / казва [низ] за n секунди блок. Също така използваме тези блокове, за да напишем инструкции за игра. Също така можем да подчертаем, че е важно да бъдете внимателни към продължителността на показване на текста.



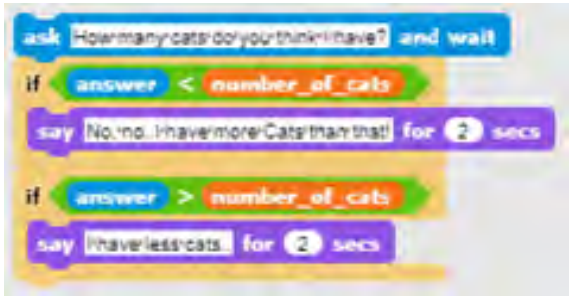

[Стъпка 5]

Обсъждаме с учениците, че не е възможно да се предскаже колко пъти играчите ще трябва да познаят, за да намерят правилния номер. Тя може да извади голям късмет и да го познае при първия си опит, може би ще са й необходими 5 предположения или дори повече, не можем да кажем! Това е причината да трябва да изберем правилния цикъл за дадената задача. Пазителят на приюта трябва многократно да поиска номер и да даде подходящ отговор, докато играчът не познае правилния номер. Единственият цикъл, който можем да реализираме

желаното изпълнение, се повтаря до цикъл [condition]. Условието е сравнително лесно да се види, трябва да го циклираме, докато отговорът на играча, който се съхранява във вградения променлив отговор, е равен на стойността, съхранявана в променливата *cat_number*.


[Стъпка 6]

След това трябва да попитаме учениците кои са командите, които ще влязат в тялото на цикъла. Каква е активността или командите, които ще се повтарят, докато играчът не познае правилния номер? Първо, трябва да помолим играча да въведе число, след което трябва да отговорим въз основа на стойността на това число.




[Стъпка 7]

Последното нещо, което трябва да се обясни или обсъди с учениците, е кога ще приключи този цикъл и какво означава това. Когато отговорите на играчите ще бъдат равни на броя на котките, и двете условия в тялото на цикъла ще бъдат неверни, така че цикълът ще премине в следващата итерация, проверявайки състоянието на цикъла. Този път условието ще бъде вярно, така че цикълът ще се прекрати и командите, които следват цикъла, ще бъдат изпълнени. Перифразирайки, когато цикълът се прекрати,

знаем, че играчът е отгатнал правилно числото. Така че сега можем да отговорим съответно.


[Стъпка 9]

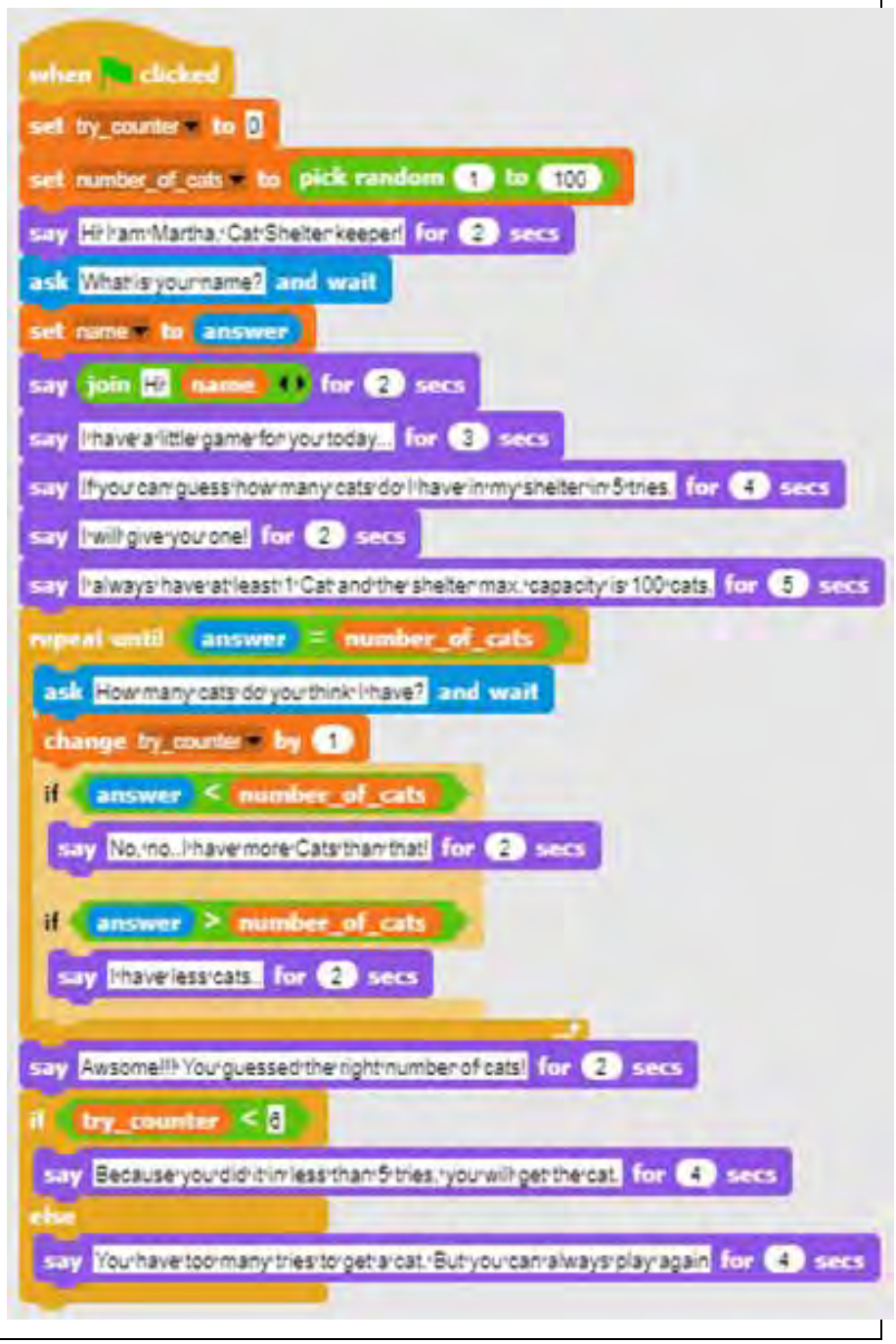
Трябва да създадем нова променлива, която ще има ролята на брояч, и да я инициализираме до стойността 0. Обсъждаме с учениците значението на инициализирането на променлива и разликата между задаването на стойността и увеличаването ѝ. Когато зададем стойността на променлива, предишната стойност се губи. Това не е добре за брояч. Ако увеличим стойността на променливата с някакво число, добавяме това число към каквато и да е променлива на стойността по-рано. Точно това искаме в тази ситуация. Всеки път, когато играч въведе ново число, ние искаме да го увеличим с 1.

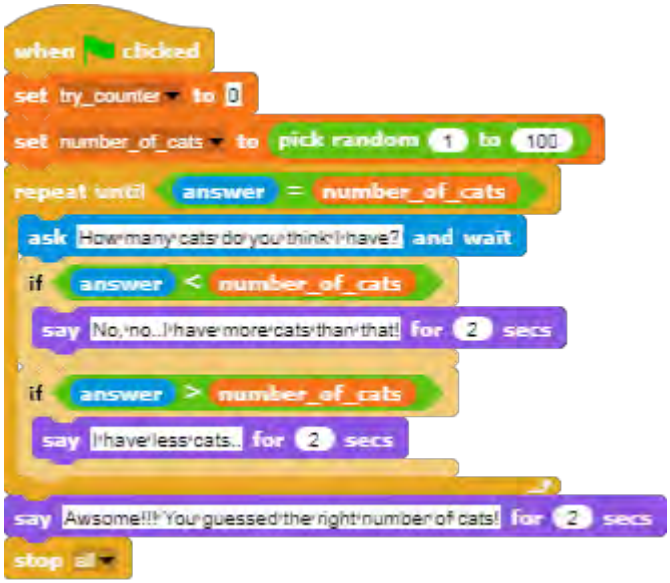
[Стъпка 10]

След верния отговор трябва да проверим стойността на променливата брояч, за да решим дали играчът ще вземе котката

или не. Тъй като Snap има само логически оператори по-малко (<) и няма оператори по-малко или равно, условието за решаване дали играчът получава котката е cat_counter <6. Това също е добър пример за използване на блока на условията If-Else, защото ние разграничават двата случая.

[Краен код]



| | |
|---|---|
| | [Основна версия на дейността]<br>За да спестим време, можем да използваме основната версия на сценария. В основна версия са включени всички основни понятия, други функции, описани по-горе, могат да бъдат използвани като по-късни надстройки.<br>  |
| **Инструменти и ресурси за учителя** | • Цялата дейност в Snap!: https://snap.berkeley.edu/project?user=zapusek&project=cats_in_a_shelter<br>• Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.<br>• Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Инструменти и ресурси за учениците** | • Шаблон в Snap!: https://snap.berkeley.edu/project?user=zapusek&project=cats_in_a_shelter_template<br>• Инструкции (C4G11_InstructionsForStudent_BG.docx |

# СЦЕНАРИИ ЗА НАПРЕДНАЛИ

## Учебен сценарий 12 – Улов на здравословна храна

| | |
|---|---|
| **Учебен сценарий**<br>**Име** | Улов на здравословна храна |
| **Предишен опит в програмирането** | • Добавяне на текст към спрайт<br>• Показване и скриване на спрайт<br>• Използване на посока на завъртане (point in direction)<br>• Използване на случайни числа<br>• Използване н апроменливи за изчисляване на точки<br>• Използване на цикъл repeat<br>• Using forever loop<br>• Using conditionals |
| **Очаквани резултати** | Основни очаквани резултати<br>• Променливи<br>• Условия<br>• Цикли<br>• Посока на завъртане<br>• Случайни числа<br>Специфични очаквани резултати, ориентирани към алгоритмично мислене:<br>• Ученикът използва променлива за предотвратяване започването на играта преди момичето да приключи да говори (по избор)<br>• Ученикът използва if за проверка (с помощта на променлива) дали храната може да започне да се движи<br>• Ученикът използва цикъл (repeat until) за движение на храната, докато точките не са по-малки от 5<br>• Ученикът използва посока на завъртане (point in direction) 180 (надолу) за спрайтове, движещи се надолу<br>• Ученикът използва произволно число за избор на брой стъпки<br>• Ученикът използва случайно число за преместване в произволно положение<br>• Ученикът използва случайно число за преместване в x (произволно), y (фиксирано) положение (moving to x (random), y (fixed) position) (по избор) |

| | |
|---|---|
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Момичето лови храна. Тя трябва да бъде внимателна, само здравословни елементи носят точки!<br>**Задачи:** Учениците трябва да програмират два различни спрайта, момиче, което дава инструкции, казва какво да прави, за да започне играта и брои точки; и храна, която случайно пада от горната част на екрана.<br>Освен това учениците могат да добавят променлива и ако изявление за предотвратяване движението на храната, преди момичето да спре да говори.<br>**Цел: Учениците ще се научат как да се движат произволно за X стъпки и да изберат позиция, както и как да използват променливи и условия за предотвратяване на други събития.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно обучение, обучение, основано на игрови дизайн, решаване на проблеми |
| **Форми на обучение** | Индивидуална работа / Работа по двойки |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)<br>Момичето лови храна. Всяка здравословна храна носи 1 точка, докато всяка нездравословна храна отнема 1 точка. Играта започва с някои инструкции, дадени от момичето. Тогава тя изчезва и храната се появява. Когато играчът събере 5 точки, храната изчезва и момичето се появява отново.<br>  |

[Стъпка 1]

Тази дейност е предназначена като индивидуална работа или работа по двойки. Учителят дава някои подсказки, обяснява някои по-трудни части и помага при нужда.

Първоначално се дава на учениците:

- Фон
- Момиче-спрайт

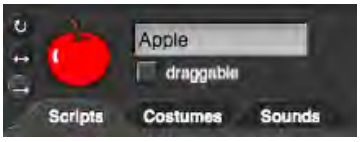
Учениците избират фон и добавят основен спрайт, напр. момиче. Момичето дава някои инструкции в началото и след това се скрива. Както видяхме от предишни дейности, добре е да използваме блок *show*, когато щракнем върху знамето (при повторно възпроизвеждане, ако спрайтът остане скрит). Кодът е например:


Ще се върнем към този спрайт по-късно. Нека сега напишем код за плод.

[Стъпка 2]

Учениците добавят нов спрайт, здравословна храна, напр. ябълка. Първо, те програмират движението на спрайта, което е отгоре надолу, така че те избират следните блокове:

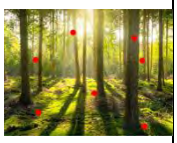

Ако не искат ябълката им да е обърната, те могат да изберат третата опция да не се върти в посока(don't rotate in direction).



За да направите играта по-интересна, броят на стъпките може да бъде избран произволно, така че скоростта не винаги ще бъде еднаква. Например:


Следващата стъпка е да помислите какво се случва, когато ябълката дойде в долната част на екрана?

В този случай учениците могат да използват блок *touching edge (докосва ръб)* в комбинация с оператора *if*. Ако ябълката докосне ръба, тя ще бъде преместена в някакво произволно положение. Блокове за движение ни предлага следващия блок:

Тази команда ще избира произволно x всякакви y координати и ябълката може да се появи навсякъде на екрана (вижте червените точки на снимката)

Ако искаме ябълката да се показва винаги в горната част на екрана, y стойността може да бъде фиксирана и само x стойността ще бъде избрана произволно. Със следния код ябълката винаги ще се показва в горната част на екрана (вижте червените точки на снимката).


[Стъпка 3]

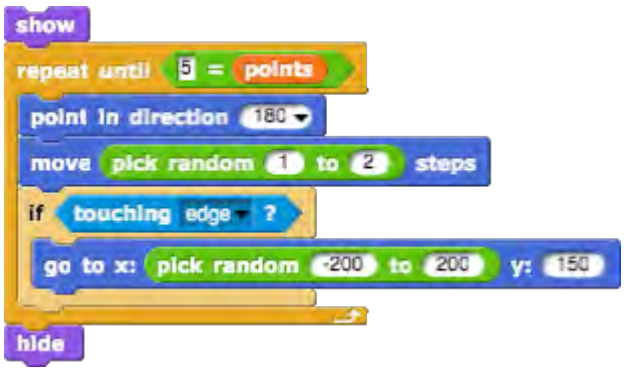
Учениците вече могат да направят променлива, точки, която ще използват за броене. Точките трябва да имат начална стойност 0 (спрайта на момичето). 


[Стъпка 4]

Ако искаме ябълката да се движи многократно, имаме нужда от цикъл. Учениците могат да използват цикъл *repeat until и да поставят условие*. Например те искат играта да завърши, когато достигнат 5 точки. Така че състоянието ще бъде *points* = 5 и цикълът ще се повтаря, докато условието не е вярно. Когато условието е вярно, това е, че играчът достига 5 точки, цикълът ще спре.


[Стъпка 5]

Не искаме ябълката да се показва в началото, но след като момичето даде указанията си. Учениците могат да програмират ябълката да се показва *when key is pressed.* Разбира се, те трябваше да добавят блок *show* преди цикъла repeat и nd *hide* след това. Целият код е:




[Стъпка 6]

Какво се случва, когато ябълката бъде щракната (или въведена с мишката)?

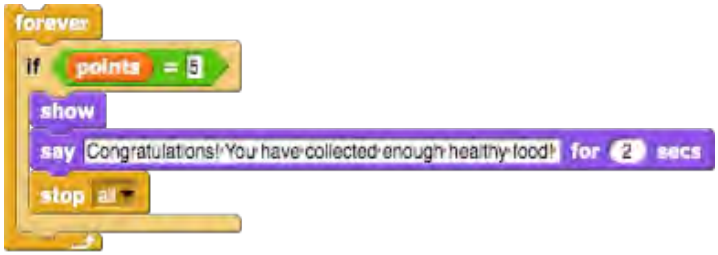
Ябълката трябва да се скрие, да преброи точки, да смени позицията си и да се покаже отново. Точките ще бъдат променени с 1 и за позицията учениците могат да използват същия код като преди.


[Стъпка 7]

Да се върнем при момичето.

Сега момичето трябва да се появи отново и да каже, напр. Поздравления!

Ще се нуждаем от цикъл завинаги, който ще провери дали сме достигнали 5 точки. Ако го направиме, момичето ще се покаже и ще каже нещо. След това ще добавим блокиране на всички. Нека учениците да разберат какво означава това спиране (без спиране момичето ще казва „Поздравления ...“ завинаги).




[Стъпка 8]

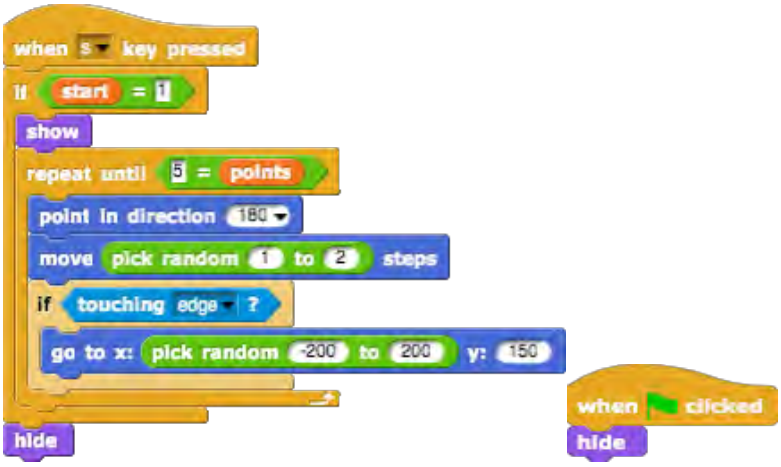
Когато играете отново играта, когато учениците вече ще знаят всички инструкции (от [Стъпка 1]) и със сигурност ще искат да ги пропуснат. Те могат да натиснат буквата „S“ преди, така че играта да започне, но момичето все още ще говори.

За да предотвратим това, можем да създадем друга променлива (с име start), която трябва да бъде зададена на 0 в началото. След това, след инструкциите на момичето, променливата старт ще се промени на 1.


Сега трябва да програмираме ябълката да стартира само ако променливата start е равна на 1. Учениците ще използват блока *if*. С това учениците няма да могат да провеждат игра, преди момичето да спре да говори.

Друго нещо може да се случи, когато играем отново играта. Ако спрем играта, когато имаме например 3 точки, ябълката няма да изчезне. В този случай, когато започнете играта отново, ябълката ще бъде видима, преди момичето да завърши с даване на инструкции. Тъй като не искаме това, ние добавяме код, който apple скрива в началото на играта.

Кодът на ябълката вече е:




[Стъпка 9]

Сега учениците могат да дублират ябълковия спрайт много пъти и да им сменят костюма (ако искат). Кодът ще бъде същият.

Единствената промяна е при нездравословната храна, където те ще загубят 1 точка, като щракнат върху нея.


[Финален код]

Girl/Момиче


Apple/ябълка


[Допълнителни задачи]

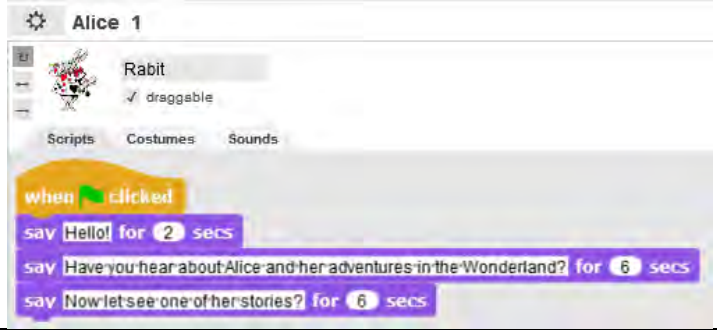
Учениците могат да добавят допълнителни задачи според техните желания или могат да следват задачите по-долу:

- Променете играта, така че спрайтът на купата да улавя храна.

| | |
|---|---|
| | • Добавете нов спрайт (купа). Начертайте го, намерете го онлайн или използвайте приложена снимка / и на купата.<br>• Задайте началната позиция на купата (например в долната част на екрана) и напишете код за движението на купата (наляво и надясно, ако искате и нагоре и надолу). Хранителните спрайтове трябва да изчезнат и да се появят отново на произволно място чрез докосване на купата (а не при щракване с мишката върху храната, както преди).<br>• Променете правилата - оставете играта да приключи, когато играчът набере 20 точки (той печели) или когато вземе 3 нездравословни храни (загуби).<br>• Добавете още хранителни спрайтове, за да направите играта по-интересна.<br>• Сменете костюма на купата, когато играч отбележи напр. 5, 10, 15 точки. |
| **Инструменти и ресурси за учителя** | • Цялата дейност в Snap!: https://snap.berkeley.edu/project?user=mateja&project=Catching%20healthy%20food<br>• Дейност в Snap! С допълнителни задачи: https://snap.berkeley.edu/project?user=mateja&project=Catching%20healthy%20food%20%2B%20Add.%20Task<br>• Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.<br>• Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Инструменти и ресурси за учениците** | • Полуготови шаблони в Snap!: https://snap.berkeley.edu/project?user=mateja&project=C4G12_Catching%20healthy%20food%20-%20Part<br>• Инструкции (C4G12_InstructionsForStudent_BG.docx)<br>• Images: bowl1.png, bowl2.png, bowl3.png, bowl4.png |

## Учебен сценарий 13 – Разказване на история

| | |
|---|---|
| **Учебен сценарий Име** | Разказване на история |
| **Предишен опит в програмирането** | • Показване и скриване на спрайт<br>• Използване на посока на движение (Point in direction)<br>• Използване на условни изрази и условен блок<br>• Използване на блок К*ажи*<br>• Използване на блок Изчакай за ... секунди |
| **Резултати от обучението** | **Общи резултати от обучението:**<br>• Основни очаквани резултати:<br>• Преместване и промяна на размера на спрайт<br>• Промяна на размера<br>• Съобщения<br>• Съставете структура на разказването на истории<br>• Промяна на фона на сцените<br>**Специфични резултати от обучението, ориентирани към алгоритмично мислене:**<br>• Ученикът планира диалози и дейности на спрайтовете в историята<br>• Ученикът изпращане на предавания за диалог между спрайтове<br>• Ученикът използва движещи се и променящи се размери за спрайтове<br>• Ученикът използва показване и скриване на спрайта<br>• Ученически рефактор и разширен код на спрайтовете |
| **Цели, Задачи и кратко описание на дейностите** | **Кратко описание:** Заекът разказва историята за Алиса в страната на чудесата. Той започва разказа на историята с няколко изречения в началото на фона на декор на сцена с надпис Allice in Wonderland. Историята на Алиса започва в гората. Алиса се разхожда и си мисли: "Къде съм?" /За да се реализира отдалечаването на Алиса, постепенно с придвижването ѝ се намалява размера. /Алиса попада на кръстопът и на дърво вижда Чеширския котарак/. Започва разговор между Алиса и Чеширския котарак.<br>Разговорът е представен на картинката.<br>**Задачи:** Учениците трябва да експериментират с кратък пример на историята за срещата между Алиса и котарака, базирана на синхронизиране на диалога чрез блок за изчакване. След това разглеждат втора версия на |

| | |
|---|---|
| | историята с използване на съобщения. Въвеждат се команди за предаване на съобщения. учениците допълват кода на героите по текста от картинката. Задачата се усложнява с въвеждане смяна на декор на сцена чрез broadcast и движение на Алиса в гората преди да срещне котаракът.<br><br>**Цел: Учениците ще се научат как да планират разказване на истории, как да използват излъчени съобщения за синхронизиране на дейностите на спрайтовете и сценични промени.** |
| **Продължителност** | 90 минути |
| **Методи на обучение** | Активно учене, обучение, основано на програмиране на игра, решаване на проблеми |
| **Форма на преподаване** | Самостоятелна работа / Работа по двойки |
| **Ход на урока** | (Мотивация-Въведение, Прилагане, Осмисляне и Оценка)<br>Учителят дискутира с учениците историята на Алиса в страната на чудесата и показва картинката със срещата на Алиса и чеширския котарак. Обяснява, че с помощта на кодирането може да се пресъздаде историята на Алиса. Поставя задача да се стартира проекта и да се разгледат кодовете на sprites https://snap.berkeley.edu/project?user=ddureva&project=Alice_1<br>Дискутира се: Кой започва да говори първи? Кога се включва Алиса и кога Котаракът? Защо при диалога на героите няма синхрон? Отговорът е в некоректното изчисляване на времената в които всеки от героите „говори“ и „неизчакването даден герой да приключи репликите си“.<br>  |

Коментират се кодовете и се попълва таблицата:

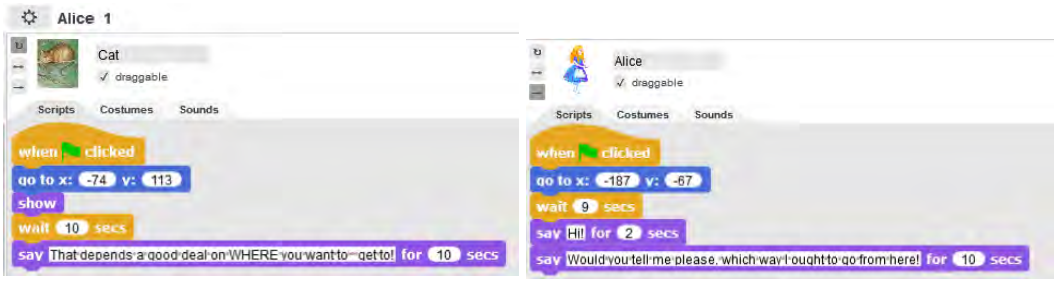
| Спрайт | Десйтвие | Време за стартиране от началото | Време за край | Продъл-жител-ност |
|---|---|---|---|---|
| Заек | Кажи: ЗдравейтеЧували ли сте за Алиса инейните приключения в Страната на чудесата? Нека да видим една от нените истории. | 0 | 14 | 14 |
| Алиса | Казва: Би ли ми казал по кой път да тръгна? | 9 | 21 | 12 |
| Котаракът | Казва: А ти къде искаш да отудеш? | 10 | 20 | 10 |

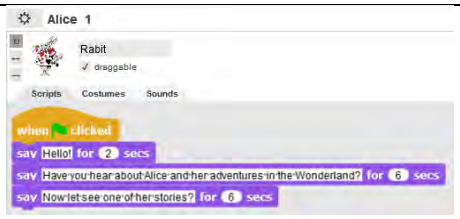
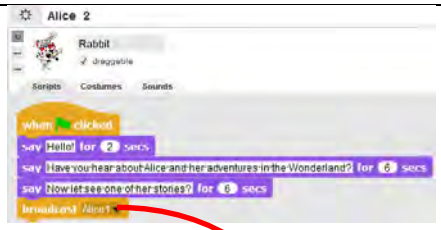
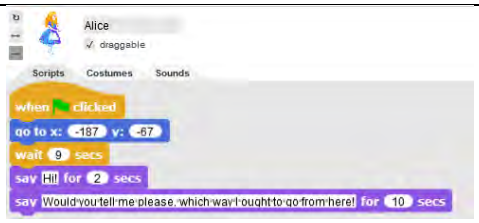
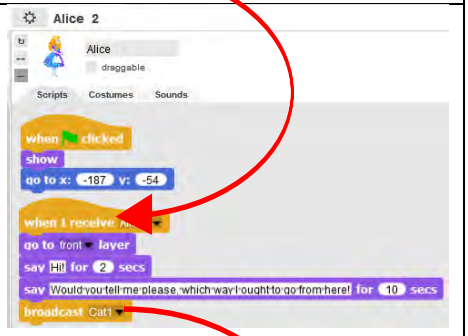
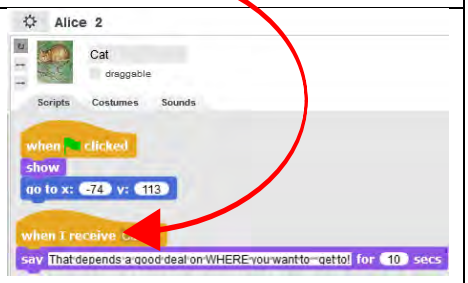
Изводът е, че синхронизирането чрез блок **wait for …second** може да доведе до грешки в поведението на героите при разказване на истории.

1. Учителят поставя задача да се стартира и разгледа кода на проект https://snap.berkeley.edu/project?user=ddureva&project=Alice_2

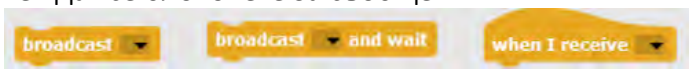
   Кои са непознатите до момента команди?

   Сравняват се кодовете от Alice_1 и Alice_2.



| Alice_1 | Alice_2 |
| --- | --- |
|  |  |
|  |  |
|  |  |

2. Въвеждат се блоковете за съобщения:




Коментира се, че съобщенията зададени с Broadcast са насочени към всички герои, но могат да се получават само от някои от героите. Блокът broadcast… and wait изисква всички герои получили съобщението да изпълнят действията си и тогава продължават действията на героя изпратил съобщението. Учителят демонстрира как се задава име на съобщение в broadcast и как се използва при събитието When I receive …

1. 

2. 

3. Въвеждане на име. ОК

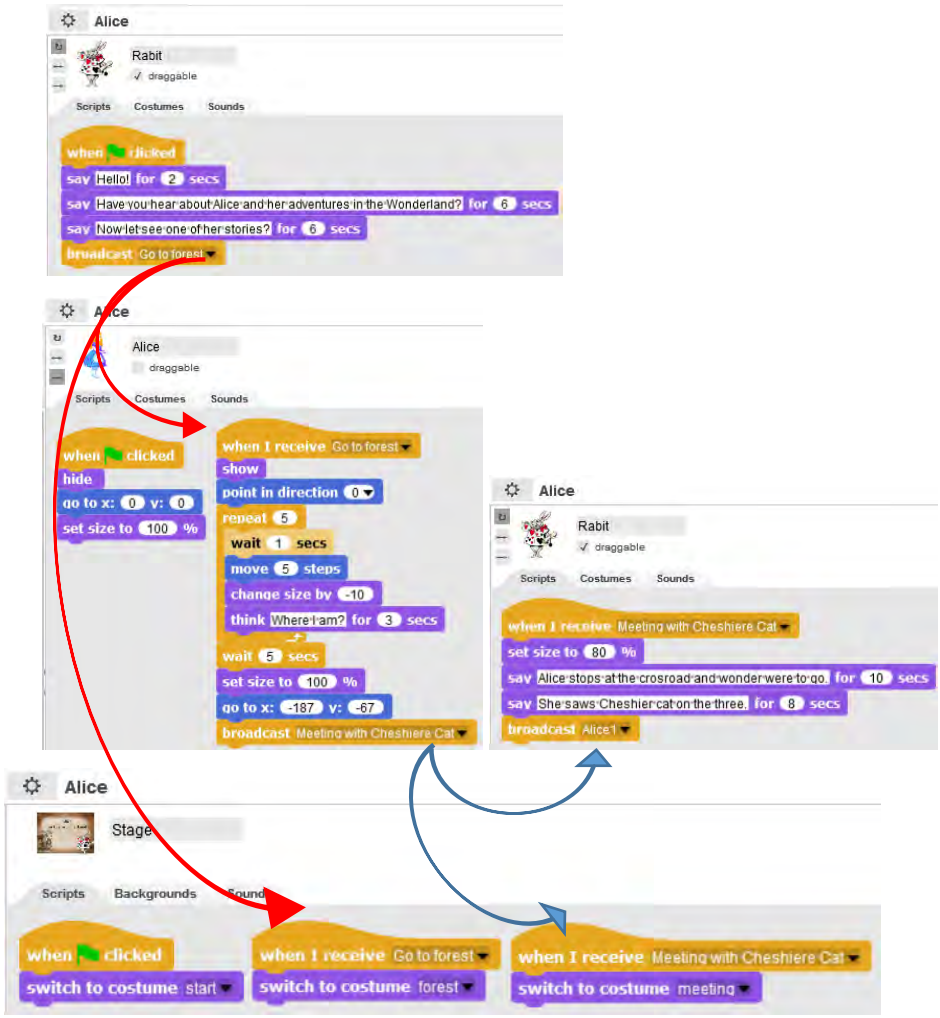
Използване в събитие: 1. 

, 2. От списъка се избира кое съобщение трябва да получи the sprait.

| | |
|---|---|
| | 3. Обсъжда се групово как да се довърши историята от картинката. Как да се наименоват съобщенията: напр. Съобщението от Cat към Alice да бъде Alice2, А от Alice to Cat – Cat1.<br>4. Учениците довършват историята по двойки.<br>5. Учителят коментира, че при разказване на истории често се налага смяна на декорите/ костюмите на сцена (stage costumes). Да направим по-пълна историята на Alice, като започнем разказа на Заека на фон за начало, прехвърлим действието в гората, където Алиса върви и се чуди „Къде съм?“ и постепенно се намалява размера и. След това се озовава на кръстопътя и вижда Чеширския котарак. Започва разговорът между двамата.<br>6. Учителят демонстрира проектът https://snap.berkeley.edu/project?user=ddureva&project=Alice<br>  |

Коментират се смяната на сцените и действията на героите. Кога се сменя дадена сцена, кога се появява Alice и какви са нейните действия? Кога се появява котаракът и какви са неговите действия?

Разглеждат се сцените в проекта Alice_2. Има 3 сцени, едната вече е използвана. Коя сцена да се използва за начало. Какво трябва да се направи, така че при стартиране на историята Alica I Cat да не се виждат? Как да сменим костюма на сцената? Може да се използва broadcast за смяна на сцената зададен от Заека, след като каже въвеждащите думи в историята. Алиса се появява при смяна на сцената със съобщението Go to forest




Когато Алиса е на пътя в гората тя върви, „мисли“ и за по-голяма реалистичност се намалява нейния размер с -10%. Това се повтаря 5 пъти с

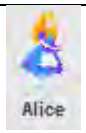
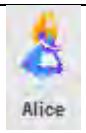
| | |
|---|---|
| | помощта на repeat.<br>Когато достигне кръстопътя се сменя сцената със съобщението “Meeting with Cheshire Cat”. Това съобщение се получава едновременно и от Заека, който намалява размера си на 80% и продължава да разказва историята. На този етап котаракът не се показва,защото присъства като част от декора. Той се появява при съобщението Cat1. Учителят може да обясни, че котаракът е изрязан от декора с помощта на външен графичен редактор. (За съжаление Snap! не предоставя големи възможности на графичния редактор за разлика от Scratch 3.0).<br>След съобщението на Заека Alice 1 историята продължава, така както е направена в проекта Alice 2.<br>7. Учителят коментира, че за да се разкаже една история най-напред трябва да се направи нейния сценарии. За целта може да се използва допълнителна таблица за описание на сценария на историята. (Приложение 1) По преценка на учителя може да се даде готовата таблица или да се даде частично попълнена, а учениците базирайки се на демонстрацията да я попълнят.<br>8. Поставя се задачата учениците да опишат разгледания сценарии и да довършат историята от проекта Alice_2 по двойки. |
| **Ресурси за учителя** | Цялата дейност е в Snap!:<br>https://snap.berkeley.edu/project?user=ddureva&project=Alice |
| **Ресурси за учениците** |  <br>https://snap.berkeley.edu/project?user=ddureva&project=Alice_1<br>https://snap.berkeley.edu/project?user=ddureva&project=Alice_2<br>Инструкции за ученици (C4G13_InstructionsForStudent_BG.docx) |

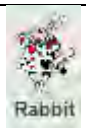
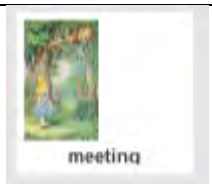
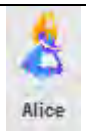
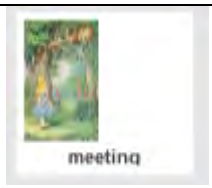
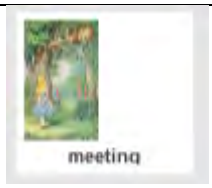
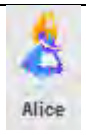
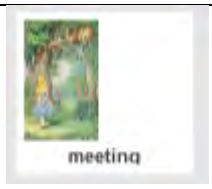
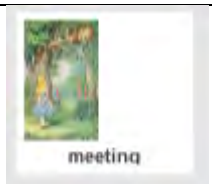
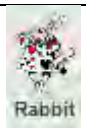
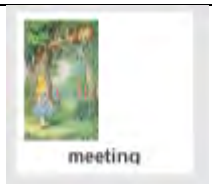
Приложение 1. Сценарии на историята

Сцени

| Име | Дизайн | Десйтвие | Бележки |
|---|---|---|---|
| 1. Start |  | Историята започва със сцената. (When green flag is clicked) | На този фон Заекът прави въведение в историята. |
| 2. Forest |  | Фонът на сцената се появява когато заекът приключи с въведението в историята. (Изпратил е съобщение Go to forest) | На този фон Алиса се появява позиционирана в центъра на сцената. Започва да се движи мислейки „Къде съм?“. Постепенно намалява 5 пъти размера си с 10% и мисли. Когато стига в края на пътя (до един кръстопът) сцената се сменя с декор Meeting. (Алиса дава съобщение Meeting with Cheshier Cat) |
| 3. Meeting |  | Появява се когато получи съобщението от Алиса Meeting with Cheshier Cat | Тук Алиса и котаракът са част от фона. За да се използва спрайта на Алиса, преди съобщението, тя се позиционира, така че да покрие изображението и от декора. Спрайта на котаракът се появява на по-късен етап.<br>Със смяната на сцената Заекът продължава да разказва историята.<br>По-късно се провежда разговорът между алиса и Чеширския котарак. |

Герои (Спрайтове)

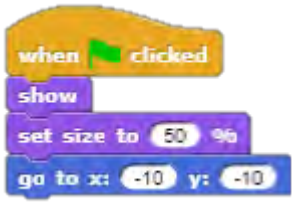
| Спрайт | Действие | Фон на сцена |
|---|---|---|
|  | При старт:<br>Казва: Здравей! за 2 сек.<br>Казва: Чувал ли си някога за Алиса и нейните приключения в Страната на чудесата? за 6 сек.<br>Казва: Сега нека да видим нейната история! за 6 сек.<br>Изпраща съобщение Go to forest /Иди в гората |  |
|  | При старт<br>Скрива се от сцената. Позиционира се в центъра на сцената и се задава размер 100% в готовност за показване в новия фон. |  |
|  | При старт<br>Скрива се от сцената. Позиционира се в x: -74, y:113 (Позициите са предварително определени след наместване на спрайта на котарака върху сцената Meeting.) |  |
|  | Получава съобщение Go to forest:<br>Показва се<br>5 пъти се повтаря: изчакване за 1 сек.; преместване с 5 стъпки; намаляване на размера (смяна с -10); „Мисли“ – „Къде съм?“<br>Подготвя се за следващия декор: Изчаква 5 сек; Възстановява размера си (смяна 100%) и се позиционира в x:-187, y:-67<br>Изпраща съобщение: Meeting with Cheshiere Cat/ Среща с котарака. |  |
|  | Няма действие. Само е показан от предходния декор. |  |

| | | |
|---|---|---|
|  | Получава съобщение: Meeting with Cheshiere Cat.<br>Сменя размера си на 80%<br>Казва: „Алиса спряла н акръстопътя и се чудела на къде да тръгне.“ за 10 сек.<br>Казва: „Тя видяла Чеширския котарак на дървото.“ за 8 сек.<br>Изпраща съобщение Alice1 |  |
|  | Получава съобщение: Alice1.<br>Премества се на преден слой (Това е необходимо, защото след нея се появява котаракът, който пречи да се визуализират репликите на Алиса, ако тя не е в предния слой).<br>Казва: „Hi!“ за 2 сек.<br>Казва: „Would you tell me please, which way I ought to go from here!“ за 10 сек.<br>Изпраща съобщение (broadcast) към Котаракът: Cat1. |  |
|  | Получава съобщението Cat1.<br>Показва се.<br>Казва: „А ти къде искаш да отидеш?“ за 10 сек.<br>Изпраща съобщение „Alice2“ |  |
|  | Получава съобщение: Alice 2.<br>Казва: ……<br>Изпраща съобщение Cat2 |  |
|  | Получава съобщение Cat2.<br>Казва: ……<br>Изпраща съобщение: Rabbit1 |  |
|  | Получава съобщение Rabbit1<br>Казва: „Каква е поуката от тази история?“ за 8 сек.<br>Казва: „За да знаеш по кой път да тръгнеш, трябва да определиш каква е целта ти.“ |  |

## Учебен сценарий 14 – Рисуване

| | |
|---|---|
| **Учебен сценарий Име** | Рисуване |
| **Предишен опит в програмирането** | Добавяне на спрайт<br>● Задаване на посока на завъртнае (point in direction)<br>● Използване на променливи за изчисляване на точки<br>● Използване на цикъл<br>● Използване на разклонения (условия) |
| **Очаквани резултати** | Осовни очаквани резултати:<br>● Променливи<br>● Разклонения и условия<br>● Цикъл<br>● Посока на джижение<br>● оператори<br>Специфични очаквани резултати, ориентирани към алгоритмично мислене:<br>● ученикът използва писалка за рисуване<br>● ученикът използва цикли за рисуване<br>● ученикът променя стойността на променлива при рисуване<br>● ученикът използва точка в посока, за да рисува предмети на сцената<br>● ученикът използва излъчване за управление на спрайт<br>● ученикът използва условни условия за смяна на сцената<br>● ученикът използва оператор> за смяна на сцената |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Климатът се е променил много, въздухът е силно замърсен поради индустрията. Трябва да се засаждат дървета, за да се подобри качеството на въздуха!<br>**Задачи:** За да подобрят качеството на въздуха, учениците трябва да програмират спрайт, да нарисуват два вида различни дървета - бор и дъб и бутони, които символизират тези видове дървета. Когато се натисне бутон, се изчертава определен тип дърво. |

| | |
|---|---|
| | **Цел: Учениците ще се научат да рисуват в Snap !, за да променят цвета и дебелината на писалката и как да използват променливи и условни условия, които причиняват ново събитие.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно обучение, обучение, основано на игрови дизайн, решаване на проблеми |
| **Форми на обучение** | Индивидуална работа / Работа по двойки |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)<br>В началото на играта на сцената се показват индустрия, която причинява климатични промени и променлива, която показва качеството на въздуха. Трябва да се засаждат дървета, за да се подобри качеството на въздуха. Могат да се нарисуват два различни вида дървета, бор и дъб. Когато е изтеглен бор, въздухът се подобрява с 3, а чрез изчертаване на дъб въздухът се подобрява с 2 единици. Когато качеството на въздуха достигне 10 единици, сцената се променя на поляна.<br><br>[Стъпка 1]<br>Учениците трябва да отворят програмата за подобряване на климата, която съдържа шаблон на фонове (промишленост и трева) и спрайтове (молив, бор, дъб и спрайт, наречени ясно).<br>Също така, добавете нов спрайт - pencil (“pencil a” от предложените спрайтове). Тъй като спрайтът е твърде голям, той трябва да бъде намален до 50%. Трябва да се посочи и началната позиция на молива (координатите), напр. X = -10, y = -10. |

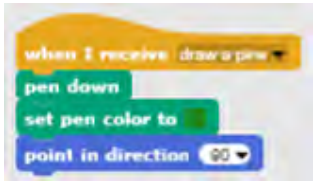




[Стъпка 2]

Спрайтът с молив трябва да получава съобщения „дъб“ и „бор“ и да изчертава подходящи дървета в отговор на съобщението. Първо маркирайте спрайта на молива и добавете кода, който ще позволи рисуването на бор с помощта на молив, когато спрайтът получи съобщението "бор".

Посоката на завъртане трябва да бъде зададена на 90, за да се очертае короната във формата на триъгълник и да се зададе цветът му.




За да нарисувате цялата корона, преместете спрайта на 40 стъпки, завъртете наляво на 120 градуса. 


Това движение трябва да се повтори три пъти. 

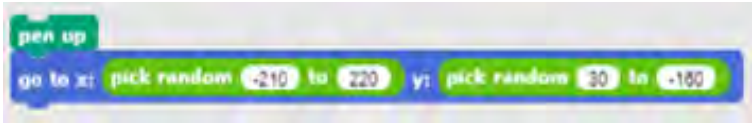

След короната, стеблото също трябва да бъде изчертано. За да бъде стеблото в правилното положение, преместете го с 22 стъпки.

След това задайте кафяв цвят на молива. 

Завъртете на 90 градуса надясно и след това преместете 10 стъпки. 

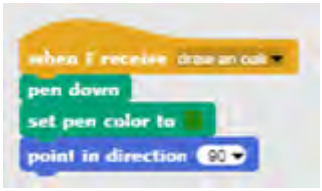

Това движение трябва да се повтори 3 пъти.

В крайна сметка е необходимо да вдигнете молива нагоре, така че спрайтът да не остави следа по време на следващото движение. Също така моливът трябва да се премести в произволно положение.




[Стъпка 3]

По същия начин е необходимо да добавите кода към молив спрайт за рисуване на дъбове. Дъбът трябва да бъде нарисуван, когато спрайтът получи съобщението „дъб“. Посоката на въртене в посока трябва да бъде настроена на 90, за да се запази короната в кръг, писалката трябва да е надолу и да се зададе цвят.




За да нарисувате короната на дъба, преместете спрайта на 1 стъпка и завъртете 3 градуса наляво след всяка стъпка.


Това движение трябва да се повтори 120 пъти. 


След като нарисуват екороната, трябва да изчертаете ствола на дървото. Спрайтът молив трябва да се премести в центъра на

| | |
|---|---|
| | нарисувания кръг с -3 стъпки и цветът на писалката да се промени на кафяв.<br>  <br><br>За да нарисува ствола на дъба, спрайтът трябва да се обърне надясно на 90 градуса и да се премести с 10 стъпки.<br>  <br><br>Тази част се повтаря три пъти.<br>  <br><br>Когато рисуването е готово, е необходимо да вдигнете молива нагоре, така че да не очертава линията, когато спрайта на молива се движи.<br>  <br><br>След изрисуването на дъба, моливът трябва да се премести в произволна позиция.<br>  <br><br>[Стъпка 4]<br>Ученици трябва да добавят кода, който кара всички изчертани дървета да бъдат изтрити, когато играчът щракне върху спрайта за изчистване Clear. Когато щракнете върху спрайта Clear с мишката, той разпространява (изпраща) съобщение за изчистване на всички дървета. Когато спрайтът Pencil получи съобщение, той изтрива всички изчертани дървета.<br>  <br> |

[Стъпка 5]

Създайте нова променлива "clean air", за да покажете текущото качество на въздуха. Задайте началната стойност 0 и покажете променливата на сцената.


Всеки път, когато се изчертава бор, въздухът се подобрява с 2 единици, така че добавете блок към спрайта бор (pine), който ще промени стойността на променливата "clear air) с 2 всеки път, когато щракнете върху бор.


Всеки път, когато се изчертае дъб, въздухът се подобрява с 3 единици, така че добавете блок към спрайта дъб (oak), който ще промени стойността на променливата "clear air" с 3 всеки път, когато щракнете върху дъб.


[Стъпка 6]

Когато променливата "clear air" достигне 10, сцената трябва да се промени на трева. Затова от изтеглените материали добавете нов фон „grass“ за сцената (фонът е от изтеглените материали,).


Добавете блок „When“ от групата „Control“ към спрайта молив.


След това добавете оператор за сравнение >.


Задайте на спрайта да разпространи съобщение „grass“когато променливата „clean air“ стане по-голяма от 10.


Добавете кода на сцената, за да смените костюма на „grass“, когато се получи съобщението „grass“.


[ДОПЪЛНИТЕЛНА ЗАДАЧА]

Можете да надстроите играта, като добавите животни, които се появяват, когато въздухът вече не е замърсен.

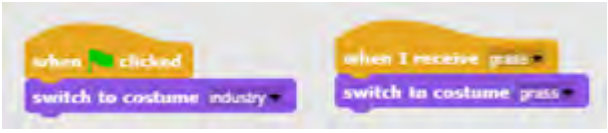
[Краен код]

Pine


Oak


X


Pencil

| | |
|---|---|
| |  <br>Stage<br>  |
| **Инструменти и ресурси за учителя** | Snap! проект “Drawing”: https://snap.berkeley.edu/project?user=tadeja&project=Improve%20the%20climate (9.1.2020) |
| **Инструменти и ресурси за учениците** | • Programming language Snap!: https://snap.berkeley.edu/ (9.1.2020)<br>• Инструкции за ученици (C4G14_InstructionsForStudent_BG.docx) |

## Учебен сценарий 15 – Хвани мишката

| | |
|---|---|
| **Учебен сценарий Име** | Хвани мишката |
| **Предишен опит в програмирането** | ● Ученикът добавя фон.<br>● Ученикът добавя нов спрайт.<br>● Ученикът добавя нов звук.<br>● Ученикът кара спрайт да каже нещо.<br>● Ученикът променя костюма на спрайта, за да направи анимация.<br>● Ученикът е реализира движението на обекта с клавишите със стрелки, използвайки събития и взема предвид ограниченията<br>● Ученикът прави разлика между две различни състояния и знае как да ги изрази с логически изрази.<br>● Ученикът използва условни условия. |
| **Очаквани резултати** | Основни очаквани резултати:<br>● Цикъл forever;<br>● Случайни числа;<br>● Брояч;<br>● Таймер.<br>Специфични очаквани резултати, ориентирани към алгоритмичното мислене:<br>● ученикът използва цикъл forever за преместване на спрайтовете;<br>● ученикът използва случайни числа, за да определи позицията на спрайта, премества спрайта за произволни стъпки и завърта спрайта за случайни градуси;<br>● ученикът прилага брояч за отчитане на улова на мишки и използва крайната стойност, за да обобщи доколко е успешен играчът;<br>● ученикът използва таймер, за да определи края на играта. |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Програмирайте игра, в която играчът (котката) ще трябва да хване мишката.<br>**Задача:** Програмирайте дейността, при която котката ще хване мишката. Котката ще бъде преместена от играча с клавиши със стрелки и мишката ще се движи произволно. Когато котката докосне мишката, мишката ще се скрие и ще се появи на произволно място. Трябва също да имаме брояч, който да отчита колко пъти котката е хванала мишката. Трябва също да имаме от таймер, за да завършим играта. След дейността |

| | |
|---|---|
| | момичето трябва да обобщи колко успешен играч е, тя казва колко пъти играчът е хванал мишката.<br>**Цел: Ученикът ще се запознае с концепцията за множество променливи случайни стойности. Те ще се научат как да използват *Operators/pick random[x]to[y]* block.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно обучение, съвместно обучение, решаване на проблеми, обучение, основано на игрови дизайн |
| **Форми на обучение** | Фронтално обучение<br>Работа по двойки / групова работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)<br>Мотивация-Въведение<br>Мотивираме учениците, като показваме играта. Обсъждаме с тях как биха започнали да програмират тази игра. Заедно със учениците определяме последователността на стъпките, например:<br>1. изберете фон и добавете спрайтове;<br>2. програмирайте котката да се движи с клавишите със стрелки;<br>3. програмирайте мишката да се движи произволно;<br>4. програмирайте мишката да се скрие (и да се появи на произволно място), когато докосне котката;<br>5. брояч на програми;<br>6. добавете таймер и определете края на играта;<br>7. добавете момичето и я програмирайте, за да обобщите колко успешен играч е бил;<br>8. програмирайте момичето да скача, когато докосне мишката;<br>9. добавете звук на котката / мишката;<br>10. и т.н.<br>Учениците могат да помогнат със стъпките или да създадат свои собствени правила на играта (но трябва да следват смели стъпки). |


Въвеждаме оператор за случайно присвояване на стойност.


Учениците програмират следните задачи по двойки / групи с подкрепата на учителя.

Изпълнение

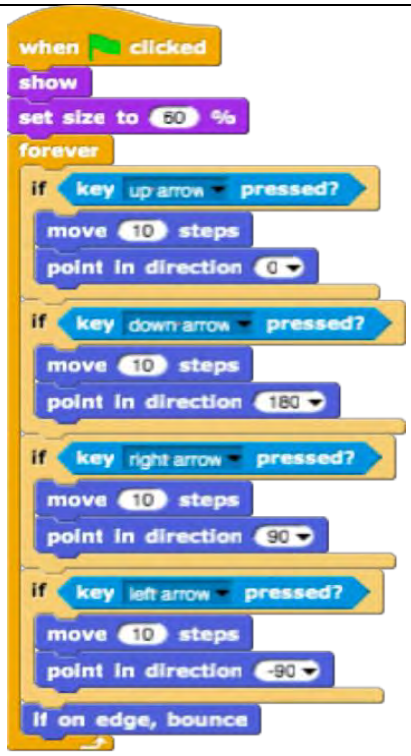
[Стъпка 1]

Първата стъпка е да се определи предисторията на играта. Учениците търсят безплатни изображения онлайн. След това те добавят нови спрайтове - котката и мишката.



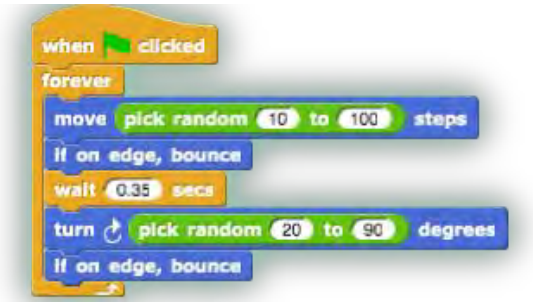
[Стъпка 2]

Учениците програмират котката да се движи с клавишите със стрелки. Тук те трябва да определят какво се случва, ако котката е на ръба.



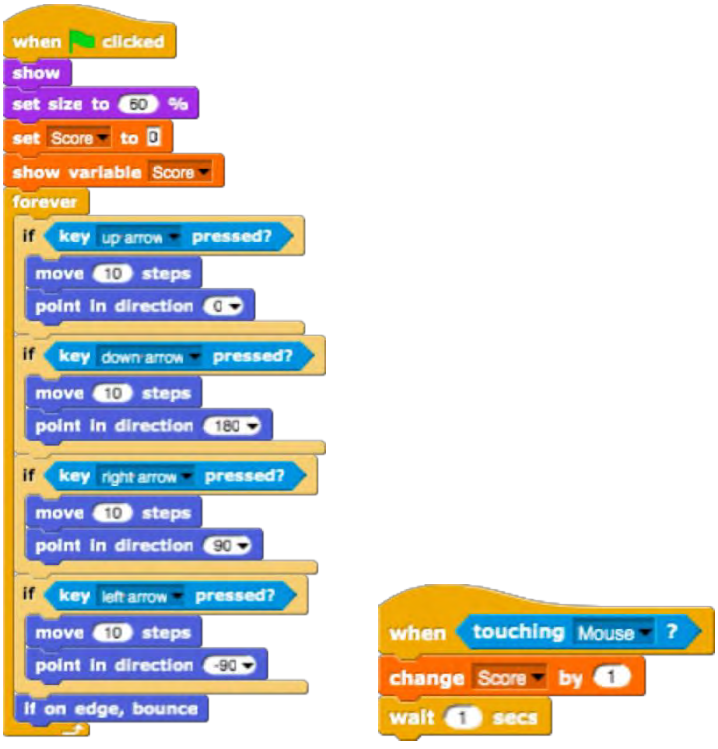

[Стъпка 3]

Учениците трябва да програмират мишката да се движи произволно. В този случай идеята е, че мишката в цикъл forever прави произволен брой стъпки и се обръща за произволна степен. Учениците правят това с блоковете *Motion/move[x]steps* и *Motion/turn[x]degrees*, в които тв вмъкват оператора *pick random[x]to[y]* operator.

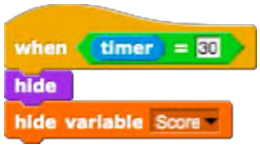



[Стъпка 4]

Следващата стъпка е да програмирате мишката да се скрие, когато докосне котката. Идеята е, че мишката се скрива и се появява на произволно място, когато докосне котката. В този случай играта не завършва при първия улов на мишката. Учениците могат да добавят

| | |
|---|---|
| | собствено правило тук. Във всеки случай те трябва да използват оператора *pick random[x]to[y]*.<br>  <br><br>[Стъпка 5]<br>В случай, че искаме да знаем колко пъти мишката е била уловена, трябва да добавим брояч. Учениците правят нова променлива - score (резултат) и я добавят към кода на котката. Резултатът в началото винаги трябва да е 0. Учениците правят това с *Variables/set[variable]to[x]* b. Ако искаме резултатът да бъде показан, учениците трябва да добавят блока *show variable[variable]*. След това учениците добавят нов контролен блок (*Control/when*) за да проверят дали котката докосва мишката. Ако котката докосне мишката, резултатът се увеличава с 1 (*Variables/change[score]by[x]*).<br>  <br><br>[Стъпка 6] |

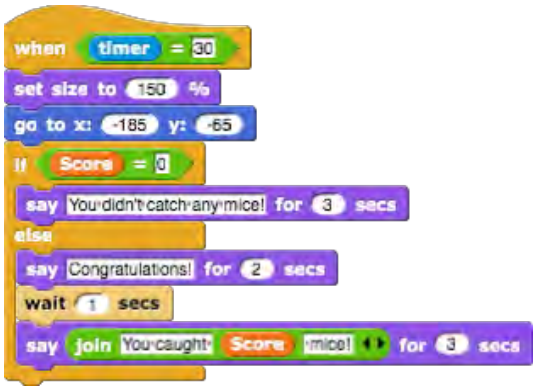
Учениците определят кога играта приключва. Те правят това с добавяне на таймера. След известно време (напр. 30 секунди) мишката и котката изчезват, променливата Резултат се скрива и играта приключва.




Учениците трябва да добавят тези блокове към скрипта за котка и мишка.

[Стъпка 7]

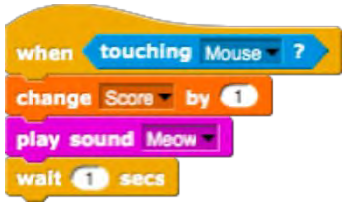
Учениците трябва да добавят код за момичето, за да обобщят крайния резултат. Ако играчът не хване никакви мишки, момичето казва: „Не сте хванали никакви мишки!“. В противен случай тя казва: „Поздравления! Хванахте x мишки! "




[Допълнителни задачи]

Учениците могат да добавят всякакви елементи към играта си. Например момичето, което скача всеки път, когато докосне мишка.

Учениците могат да добавят звук. Например, те добавят звук на котката. Звукът се възпроизвежда, когато мишката бъде уловена.



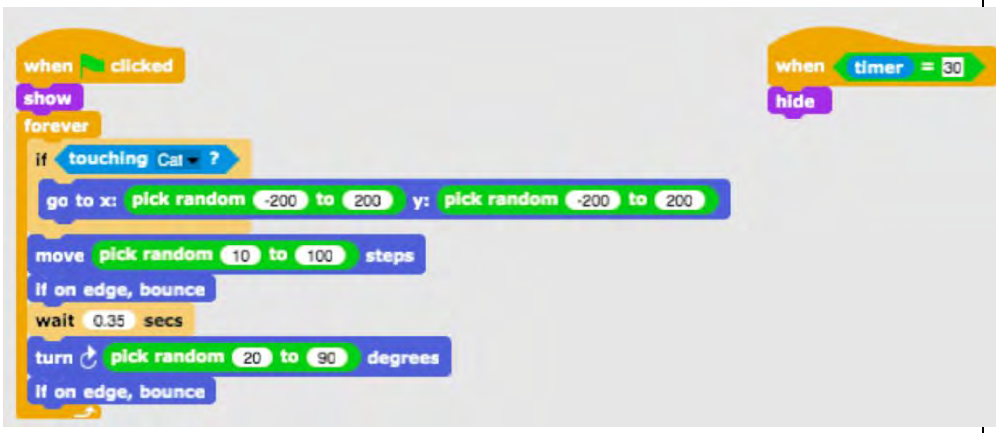

**Рефлексия и оценка**

Учениците коригират кода:

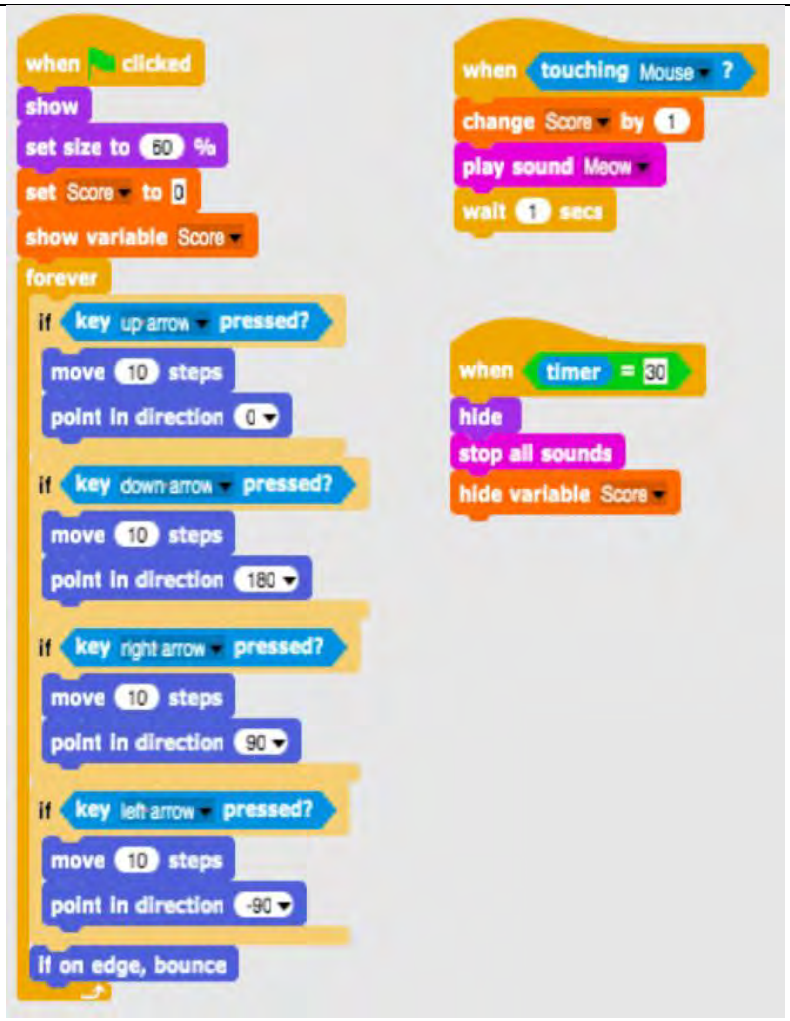
- мишката се движи 20 до 60 стъпки завинаги;
- мишката отива на място x = 100, когато докосне котката;
- мишката се завърта завинаги на 90 градуса;
- и т.н.

**[Финален код]**

*The mouse*




*The cat*




*The girl*


*The background*

| | |
|---|---|
| **Инструменти и ресурси за учителя** | • Цялата дейност в Snap!: https://snap.berkeley.edu/project?user=tadeja&project=Catch%20the%20mouse<br>• Безплатни изображения: https://pixabay.com/<br>• Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.<br>• Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Инструменти и ресурси за учениците** | • Шаблон в Snap!: https://snap.berkeley.edu/project?user=tadeja&project=Catch%20the%20mouse_0<br>• Безплатни изображения: https://pixabay.com/<br>• Инструкции (C4G15_InstructionsForStudent_BG.docx) |

## Учебен сценарий 16 – Купуване на храна за пикник

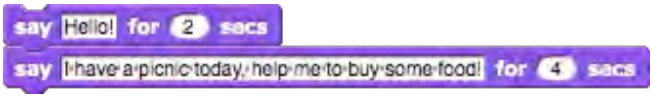
| | |
|---|---|
| **Учебен сценарий Име** | Купуване на храна за пикник |
| **Предишен опит в програмирането** | • Добавяне на текст към спрайт<br>• Показване и скриване на спрайтове<br>• Използване на оператори<br>• Използване на променливи<br>• Използване на слепване на стрингове<br>• Използване на условия и разклонения |
| **Очаквани резултати** | Основни очаквани резултати:<br>• Променливи<br>• Условни блокове<br>• Оператори<br>Специфични очаквани резултати, ориентирани към алгоритмично мислене:<br>• Ученикът използва променливи за определяне на цената за различни спрайтове - храни<br>• Ученикът променя стойността на променливите, тъй като бюджетът се променя, когато играчът купи храна<br>• Ученикът използва if оператор за проверка на наличността на пари<br>• Ученикът използва оператори за присъединяване на текст - стойност на променливите - текст<br>• Ученикът използва оператори за сравняване на цени и бюджет<br>• Ученикът използва оператори (изваждане) за промяна на стойността на променливите |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Момичето отива на пикник и се нуждае от помощ при закупуване на храна. Тя има 15 EUR и не може да похарчи повече. Когато тя купи нещо, стойността на бюджета се променя. Ако бюджетът ѝ е твърде нисък, тя не може да купи избраната храна.<br>**Задачи:** Учениците трябва да програмират три различни спрайта: момиче, храна (която могат да дублират с леки промени) и бутон за завършване. Момиче дава инструкции, казва колко пари |

| | |
|---|---|
| | има играчът и накрая (с щракване върху бутона за финал) тя казва колко здравословни и нездравословни продукти е купил играчът. Храната казва цената си, когато курсорът на мишката я задържи. Ако играчът има достатъчно пари, той може да закупи продукт и стойността на бюджета се променя. В противен случай храната не може да бъде закупена.<br>**Цел: Учениците ще се научат как да работят с променливи: задаване на различни начални стойности, използване на условни условия за сравняване на стойността на променливите, промяна на стойността на променливите, използване на променливи за отчитане на (не) здравословна храна. Освен това те ще повтарят добавянето на текст, обединяването на текстове и ако изявление.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно обучение, обучение, основано на дизайн на игри, решаване на проблеми |
| **Форми на обучение** | Индивидуална работа / Работа по двойки |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)<br>Момичето е в бакалия и купува храна за пикник. Тя има 15 EUR. Тя може да види цената на храната, когато се задържи курсорът на мишката и да я купи, като се щракне върху избраната храна. Тя може да купува храна само докато има достатъчно пари. Купете, като щракнете върху бутона за завършване, тя казва колко здравословни и нездравословни продукти е купил играчът. |


[Стъпка 1]

Тази дейност е предназначена като индивидуална работа или работа по двойки. Учителят дава някои улики, обяснява някои по-трудни части и помага при нужда.

Учениците избират фон и добавят основен спрайт, напр. момиче. Момичето дава някои инструкции в началото, напр .:




[Стъпка 2]

В тази игра ще ни трябват няколко променливи:

- *budget*, за определяне на размера на наличните пари,
- *healthy_food*, за преброяване колко здравословни елементи е купил играчът,
- *unhealthy_food*, за преброяване колко нездравословни елементи е купил играчът,
- a variable for each food, e.g. *watermelon_price*, за определяне на цената на всяка храна.

В началото променливата *budget* има стойност например 15 (EUR). Другите две променливи са зададени на 0. Този код може да бъде добавен преди кода на момичето от [Стъпка 1].

[Стъпка 3]

Учениците добавят спрайт (храна) и избират костюма му.

Кодът на храната (динята) се нуждае от три контролни събития:

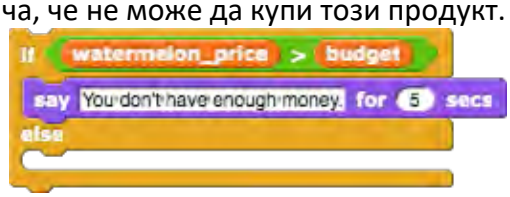
a) *When the green flag clicked*: да покаже и определи цената на храната. Нека цената на променливата бъде разумно определена (разбира се, не 0, а по-голяма от 1).


b) *When mouse-entered:* за да кажете на играча колко струва продуктът. Учениците могат да използват блок *Looks – thinking* със слепения текст – стойността на променливата и тескт за пояснение , например:


c) *When clicked*: тук трябва да направят малка рефлексия (да помислят).

1) В кой случай играчът може да закупи продукта и в кой не?
2) Какво се случва с бюджета, ако той купи храната?
3) Как да броим закупените продукти?
4) Какво се случва с храната на рафта?

1) Играчът може да закупи продукта, ако има достатъчно пари. Така че учениците трябва да сравняват две променливи: *budget* and *watermelon_price*. Ако динята струва повече, отколкото има, той не може да я купи. Учениците могат да добавят малко текст, за да кажат на играча, че не може да купи този продукт.




2) Ако играчът има 15 EUR и купи диня за 4 EUR, той вече има 15 - 4 = 11 EUR. Така че бюджетната стойност е сега: Стойността на предишнич бюджет *budget value* – *watermelon_price*.

Учениците могат да добавят малко текст и тук.


3) Прebroяването на броя на закупените продукти ще се осъществи с промяна на променливата *healthy_food* va с 1.


4) Когато върху храната се щракне, тя се скрива.


Едно възможно репение е:


[Стъпка 4]

За да има повече храна по рафтовете, учениците могат да дублират спрайта на динята. Да приемем, че втората храна ще бъде торта. Тогава кодът от [Стъпка 3] се нуждае от някои промени. Учениците трябва да:

- Сменят костюм
- Да създадат нова променлива: *cake_price*
- Да присвоят някаква стойност на промвнливата cake_price
- Да променят в кода всеки блок, съдържащ променливата *watermelon_price* с *cake_price*
- Променят отговора при закупуване на тортата
- Да заменят *change healthy_food by 1* с *change unhealthy_food by 1*

E.g., За тортата кодът към *when clicked* е:


[Стъпка 5]

Когато играчът приключи покупката си, той щраква върху бутона Finish. За да кажем на програмата, че играчът е щракнал върху бутона (приключил с купуването на храна), разпространяваме съобщение.


[Стъпка 6]

В края се връщаме към спрайта на момичето.

Когато играчът завърши пазаруването си, искаме момичето да му каже колко здравословни и нездравословни продукти е купил.

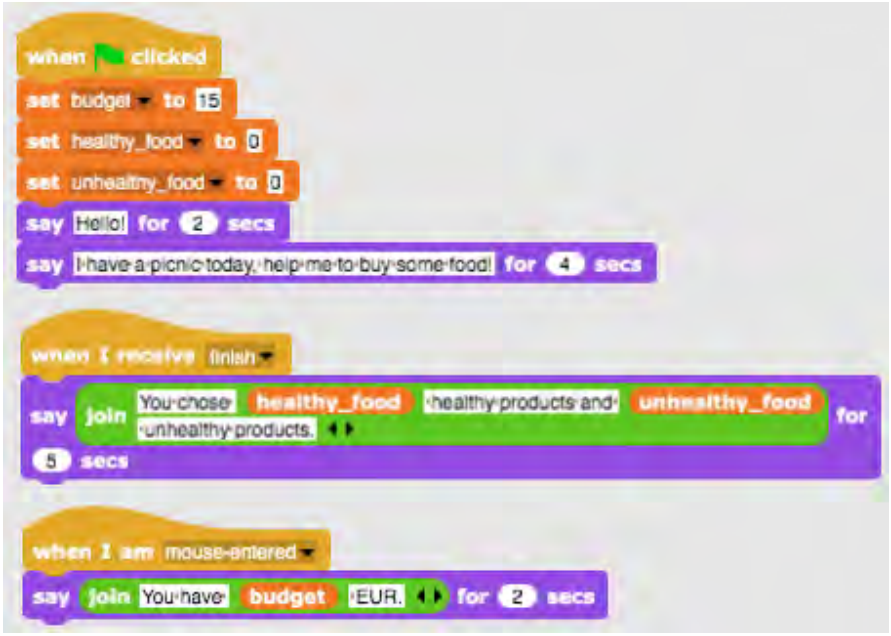
Когато играчът щракне върху бутона за финал, се изпраща съобщение за финал. Когато момичето получи финала на съобщението, тя казва, напр. „Избрахте X здравословни продукти и Y нездравословни продукти“.


[Стъпка 7]

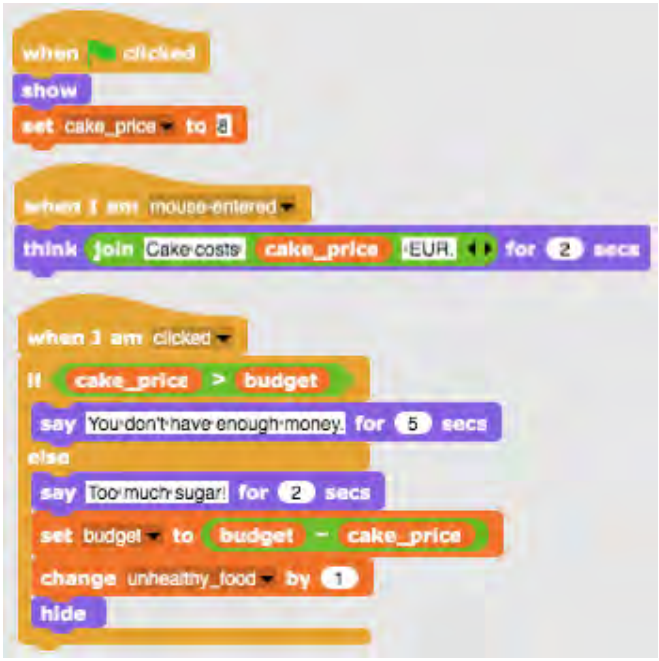
По всяко време по време на играта, играчът може да провери бюджета си, като постави мишката върху момичето. Например тя може да каже / мисли нещо като:

[Финален код]

Girl




Food




Finish Button

| | |
|---|---|
| | [Допълнителни задачи]<br>Учениците могат да добавят допълнителни задачи според техните желания или могат да следват задачите по-долу:<br>● Променете играта, за да можете да купувате всяка храна 3 пъти.<br>● Дайте повече пари на играча в началото.<br>● В края момичето разказва също колко продукти сте закупили. Напр. “Купихте 2 пъти диня, 1x грозде, 2 пъти пържени картофи”. |
| **Инструменти и ресурси за учителя** | ● Цялата дейност в Snap!:<br>https://snap.berkeley.edu/project?user=mateja&project=Buying%20food%20for%20a%20picnic<br>● Допълнителните задачи (възможни решения):<br>https://snap.berkeley.edu/project?user=mateja&project=Buying%20food%20for%20a%20picnic%20%2B%20Add.%20Task<br>● Lajovic, S. (2011). *Scratch. Nauči se programirati in postani računalniški maček.* Ljubljana: Pasadena.<br>● Vorderman, C. (2017). *Računalniško programiranje za otroke.* Ljubljana: MK. |
| **Инструменти и ресурси за учениците** | Инструкции за учениците<br>(C4G16_InstructionsForStudent_BG.docx) |

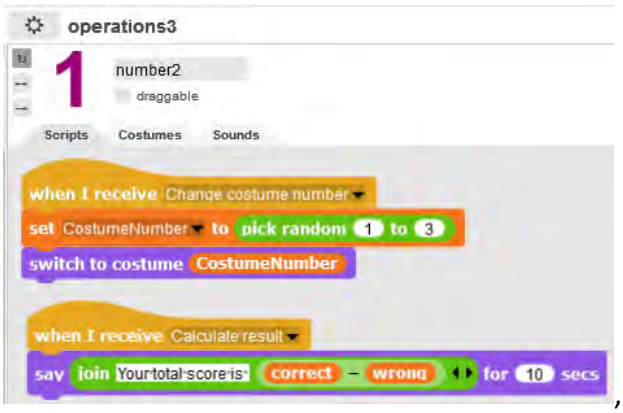
## Учебен сценарий 17 – Операции

| | |
|---|---|
| **Учебен сценарий** | Операции |
| **Предишен опит с програмирането** | • Използване на променливи за преброяване на точки в игра и за избор на костюм на сцената и спрайта;<br>• Използване на случайно число за избор на костюм на сцената и на спрайта.<br>• Използване на цикъл за повторение<br>• Използване на условни блокове<br>• Използване на операции за сравнение<br>• Използване на блок за диалогов прозорец (попитайте ... и изчакайте)<br>• Използване на съобщения за синхронизиране на действията |
| **Резултати от обучението** | **Общи резултати от обучението:**<br>• Променливи<br>• Условни блокове<br>• Цикъл<br>• Сензорни блокове<br>• Синхронизиране на действията чрез съобщения<br>**Специфични резултати от обучението, ориентирани към алгоритмично мислене:**<br>• Ученикът използва променливи за преброяване на точки и за костюми на сцената и на запазването на спрайт.<br>• Ученикът използва променливи за преброяване на точки<br>• Ученикът инициализира променливи за преброяване на точки<br>• Ученикът използва условни и логически операции<br>• Ученикът използва съобщения за промяна на спрайта и изчисляване на крайния резултат. |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:**<br>Нека с игра да проверим дали играчът е усвоил аритметичните операции в Snap!. Правилата са следните: Десет пъти по-случаен начин се избира аритметична операция с първи операнд 6 и по случаен начин се избира втори операнд – число от 1 до 3. Играчът трябва да въведе верен отговор. Броят се верните и грешните отговори. В края на играта се съобщава какъв е верният резултат.<br>**Задача:** Учениците трябва да определят декора на сцената и костюмите на спрайта; да планират необходимите променливи; да определят какви |

| | |
|---|---|
| | блокове са им необходими. На финала трябва да създадат кодовете към сцената и спрайта.<br>Допълнителните задачи могат да са: в зависимост от получения резултат, спрайтът да казва „Справи се чудесно!“, „Все още не познаваш добре аритметичните операции в Snap!“ и т. н.<br>**Цел: Учениците ще подобрят своите вече придобити знания за променливи, случайни числа, цикли, съобщения.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно учене (дискусии, експеримент с вече подготвена игра), обучение, основано на програмиране на игра, решаване на проблеми, |
| **Форми на обучение** | Самостоятелна работа / Работа по двоики/ Презентация пред класа/групата |
| **Ход на урока** | (Мотивация-Въведение, Прилагане, Осмисляне и Оценка)<br>Учителят поставя проблема за необходимостта от игра, която да се установи дали са усвоени добре аритметичните операции в Snap! и демонстрира проекта. https://snap.berkeley.edu/project?user=ddureva&project=operations3.<br>  <br>1. Обсъжда се как да се формулира условието на задачата. Прави се формулировка на задачата.<br>*Десет пъти по-случаен начин се избира аритметична операция с първи операнд 6 и по случаен начин се избира втори операнд – число от 1 до 3. Играчът трябва да въведе верен отговор. Броят се* |

*верните и грешните отговори. В края на играта се съобщава какъв е верният резултат.*

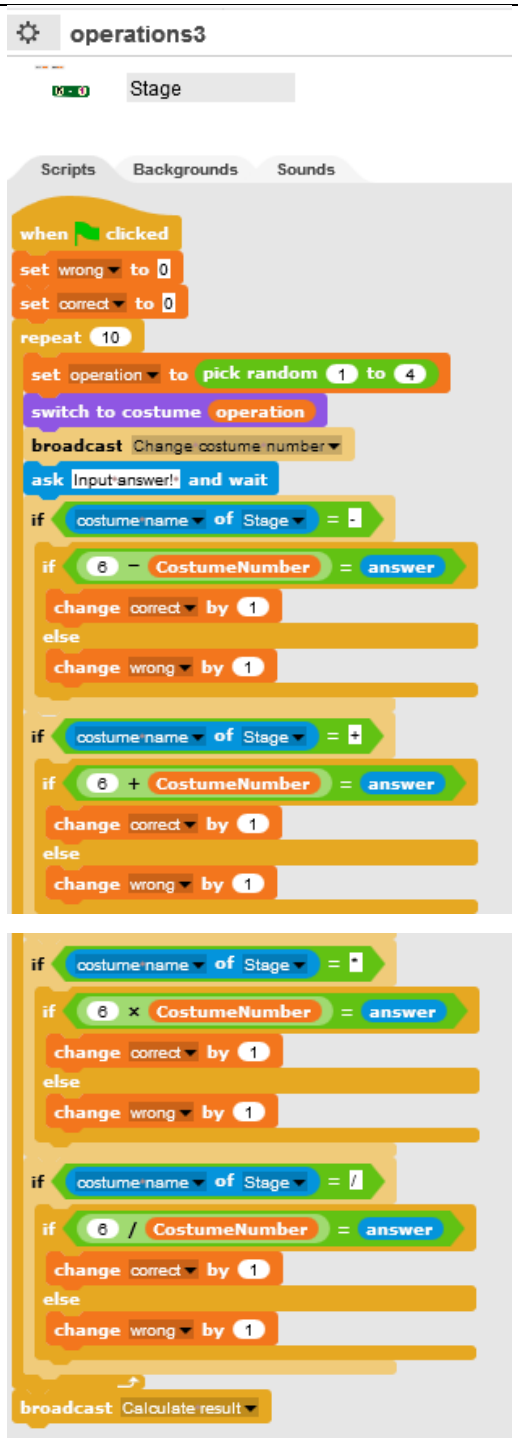
2. Коментират се променливите и начина на тяхното дефиниране, инициализиране и промяна на стойността им.
3. Припомнят се командите за random number, аритметичните и логическите операции, broadcast event
4. Обсъжда се дали основният код да е към сцената или към спрайта. В примера основният код е към сцената, а в кода на спрайта има скриптове за смяна на костюм и изчисляване на крайния резултат.




В кода на сцената се прави инициализация на променливите за верни и грешни отговори.

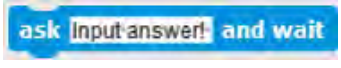
За избор на операция се използват командите:


Избора на костюм за спрайта се осъществява с broadcast to Number Sprite. избраният номер на костюма се съхранява в променливата CostumeNumber, която е дефинирана за всички обекти в проекта и затова може да се използва в кода на сцената.

| | Код към сцена | Декори на сцена |
|---|---|---|
| |  |  |





След като са избрани по случаен начин декора на сцената и костюма на спрайта се поставя въпрос към играча за въвеждане на верния отговор от операцията с командата. 

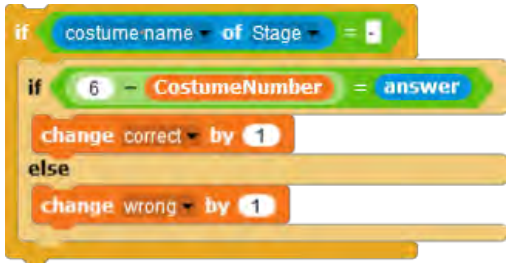
Въведеният отговор се сравнява с резултата от избраните операции.

Използва се командата

**if (условие)**

**else**

Ако е избрана операция „-„ , то се прави проверка дали 6 минус „номера на костюма на спрайта“ съвпада с отговора. Ако съвпада се увеличава променливата correct, иначе се увеличава променливата за броене на грешни отговори wrong.

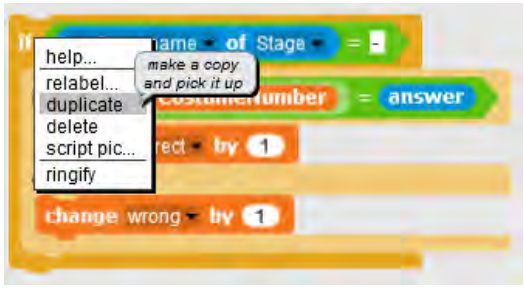



За останалите команди скриптът е подобен, разликата е в избраната операция.

За да се спести подреждането на кода за останалите операции може да се покаже как се копира част от кода и се променя аритметична операция в  :

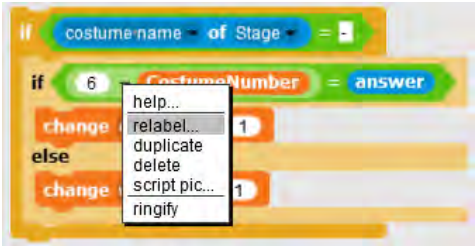
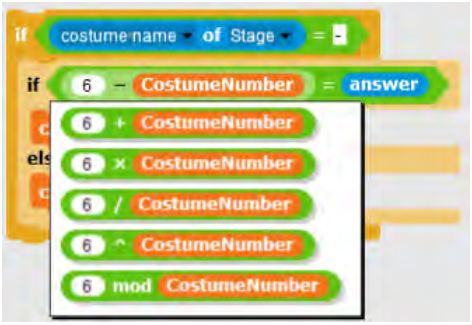
Копиране на код:

1. Щракни с десния бутон на мишката върху кода
2. От контекстното меню избери Choose duplicate (дублирай)




3. Постави с мишката дублирания скрипт на съответното място.

Учителят може да постави задача учениците сами да открият как да копират част от кода.

| | |
|---|---|
| | Промяна на операцията.<br>1. Щракни с десния бутон на мишката върху дадена операция. Ще с епоявя контекстно меню.<br>  <br><br>2. Появява се списък с операции.<br>  <br><br>3. Избери операция<br>Забележка: Ако възрастта и знанията на учениците за аритметичните операции позволяват задачата може да се разшири с операциите степенуване (^) и деление по модул (mod)<br>5. Учениците работят в екипи като създават собствени костюми за сцената и спрайта. При ограничения във времето може да се използва “half backed” project, който съдържа готовите сцена и спрайт. |
| **Инструменти и ресурси за учителя** | Цялата дейност е в Snap!:<br>https://snap.berkeley.edu/project?user=ddureva&project=operations3<br>Цялата дейност е в Scratch:<br>• Дурева Д., М. Касева, Г. Тупаров, Компютърно моделиране, 4. клас, Просвета, 2018, София (Dureva, D., M. Kaseva, G. Tuparov, Kompyutarno modelirane, 4. klas, Prosveta, 2019, Sofia) |
| **Ресурси/материали за ученика** | https://snap.berkeley.edu/project?user=ddureva&project=operations_half<br>Инструкции за учениците<br>(C4G17_InstructionsForStudent_BG.docx) |

## Учебен сценарий 18 – Рециклиране

| | |
|---|---|
| **Учебен сценарий** | Рециклиране |
| **Предишен опит с програмирането** | • Показване и скриване на спрайт<br>• Използване на променливи за преброяване на точки<br>• Използване на цикъл завинаги<br>• Използване на условни блокове<br>• Използване на операции за сравнение<br>• Използване на чувствителност на цветовете |
| **Резултати от обучението** | **Общи резултати от обучението:**<br>• Променливи<br>• Условни блокове<br>• Цикъл<br>• Придвижване до точка в посока<br>• Сензорни блокове<br>• Рефакторинг на код<br>**Специфични резултати от обучението, ориентирани към алгоритмично мислене:**<br>• Ученикът използва **Изчакай до (Wait until)** и логически операции за приключване на играта<br>• Ученикът използва **Изчакай до (Wait until)** и блок за промяна на сцената<br>• Ученикът използва променливи за преброяване на точки<br>• Ученикът използва условни и логически операции<br>• Ученикът сравнява кодовете на подобни спрайтове.<br>• Ученикът прави рефакторинг на кода<br>• Ученикът използва позициониране на спрайтовете (в допълнителна задача използвайте произволно позициониране) |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:**<br>Някой е разхвърлял боклук пред училището. Играчът е помолен да помогне за разделно събиране на боклука като го сортира за рециклиране на хартия и стъкло. Когато се поставя боклук в правилния контейнер, то боклукът се скрива. Ако боклукът се поставя в неправилен контейнер се появява съответното съобщение – „Това не е контейнер за хартия.“ или „Това не е контейнер за стъкло“ и боклукът се връща на първоначалната си |

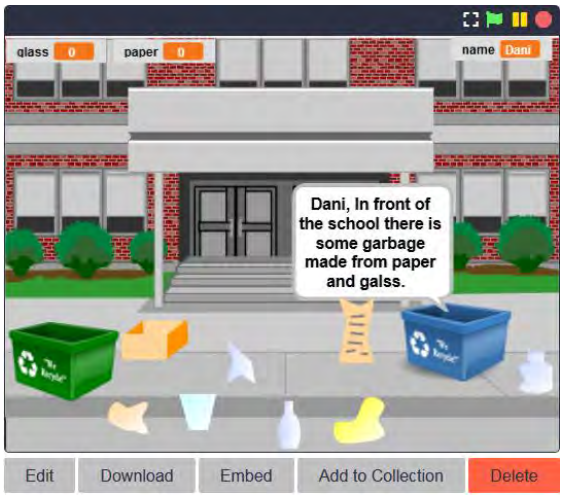
| | |
|---|---|
| | позиция. Играта приключва когато всичкия боклук е поставен в правилните контейнери.<br>**Задача:** Учениците трябва да изследват кодовете на сцената и спрайтовете, да сравняват кодовете от типа хартия за отпадъци и отпадъци от стъкло, да добавят нови спрайтове и код и да променят кода в сцената по отношение на новите добавени спрайтове. Допълнителни задачи могат да бъдат към:<br>• промяна на положението на отпадъчните спрайтове с произволен избор на координати на спрайтовете;<br>• намалете броя на етапите и извлечете робота като отделен спрайт. (Роботът е част от фона на сцената).<br>**Цел: Учениците ще подобрят своите вече придобити знания и ще разширят сценария на играта с нови обекти, код и променящ се код по отношение на нови спрайтове. Те ще бъдат обучени да правят рефакторинг на код.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Активно обучение (дискусии, експеримент с вече подготвена игра), обучение, основано на играл дизайн, решаване на проблеми. |
| **Форми на обучение** | Самостоятелна работа / Работа по двойки/ Презентация пред класа/групата |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Осмисляне и Оценка)<br>Учителят поставя проблема за разделно събиране на боклук и коментира какви са цветовете на кофите за различните видове боклук – син за хартия, зелен за пластмаса.<br>1. Поставя за задача учениците да изиграят играта и опишат с думи: Колко сцени наблюдават и колко спрайта (герои)? Как започва играта? Кой спрайт пита за името на играча? Колко променливи се използват и как са наименовани? Какво се |

случва когато хартия се постави в контейнер за стъкло и какво когато се постави в контейнер за хартия?


2. Актуализация на изучавани команди

Припомнят се командите за осъществяване на диалог с потребителя.

Прави се коментар за смяна на сцените – Сцена 1 с Робота, Сцена 2 с училището и разхвърляния боклук и Сцена 3 с Робота и надписа Bravo!. Обсъждат се възможните команди за смяна на сцена.




Дискутира се, че проверката за правилното поставяне на боклука в даден контейнер трябва да се осъществи с условен блок (conditional block) и блокове с условия за допир от групата Sensing.

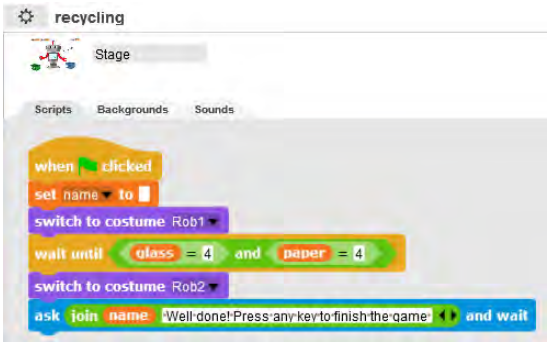
Прави се словесно описание: Ако даден боклук-хартия допира кофата за хартия, то боклука се скрива (поставен е в правилната кофа) и точките за прибрана хартия се увеличават с 1. Ако даден боклук-хартия допира кофата за стъкло, то „казва“ – „Това не е контейнер за хартия“. Прави се аналогия и за боклук-стъкло.


3. Разглеждане на кодовете на сцените и героите.

След обсъждането на възможностите за решение на задачата се разглеждат кодовете към сцената и героите.

За сцената се коментира кода с акцент върху:

- задаването на начална стойност на променливата name и използването и в диалог с потребителя;
- смяна на декорите (костюмите) на сцената и условието за приключване на играта.




При разглеждането на кодовете на героите е добре да се покажат на един слайд или да се дадат в печатен вариант по два кода на боклук-хартия и боклук-стъкло. Прави се сравнение между общите и различните елементи в кодовете.


4. Поставяне на задача за допълване с два нови спрайта – боклук-хартия и боклук-стъкло, задаване на код към тях и промяна в кодовете на сцената и контейнерите за боклук.

Коментира се как да създадат двата нови спрайта. Варианти – дублиране на съществуващи и редактиране в Snap!, създаване на нови в графичен редактор или търсене на свободно разпространявани изображения в интернет и импортирането им в играта.

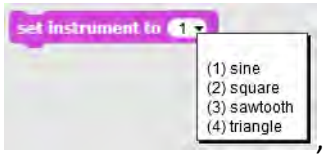
Необходимо е да се направи коментар и за промените в кода на сцената по отношение на завършването на играта.

| | |
|---|---|
| | Да се коментира възможно ли е задаването на началните стойности на променливите да бъде не в кода на двата контейнера, а в кода на сцената и съответно да се направи корекция.<br>По преценка на учителя може да се усложни условието на задачата:<br>• при всяко стартиране на играта боклукът да се разпръсва на произволно място. Тук е добре да се обърне внимание, че трябва да се ограничат координатите в които би могъл да се разпръсне боклука, така че да бъде на реалистично място. Например ограничен от координатите на червения правоъгълник.<br>  <br>• Да се въведе нов спрайт Робот и да се намали броят на декорите на сцената. Да се напише съответния код към Робота, така че той да води диалога с играча, а не спрайта - син контейнер. |
| **Инструменти и ресурси за учителя** | Цялата дейност е в Snap!:<br>https://snap.berkeley.edu/project?user=ddureva&project=recycling<br>Цялата дейност е в Scratch:<br>• Дурева Д., М. Касева, Г. Тупаров, Компютърно моделиране, 4. клас, Просвета, 2018, София (Dureva, D., M. Kaseva, G. Tuparov, Kompyutarno modelirane, 4. klas, Prosveta, 2019, Sofia) |
| **Ресурси/материали за учениците** | https://snap.berkeley.edu/project?user=ddureva&project=recycling<br>Инструкции за учениците (C4G18_InstructionsForStudent_BG.docx) |

## Учебен сценарий 19.1 - Свири на пиано

| | |
|---|---|
| **Учебен сценарий** | Свири на пиано |
| **Предишен опит в програмирането** | Използване на променливи за преброяване на точки;<br>Използване на събитие Когато съм под натиск<br>Използване на цикъл за повторение (repeat loop)<br>Използване на условни блокове<br>Използване на съобщения за промяна на костюмите на сцената и за управление на дейностите на спрайта |
| **Резултати от обучението** | **Общи резултати от обучението**:<br>• Променливи<br>• Условни блокове<br>• Цикъл<br>• Съобщения;<br>• Звуци;<br>• Програмиране на музика;<br>**Специфични резултати от обучението, ориентирани към алгоритмично мислене:**<br>• Ученикът използва променливи за преброяване на точки.<br>• Ученикът инициализира променливи за преброяване на точки<br>• Ученикът използва условни блокове, за да оцени постигнатите точки<br>• Ученикът използва съобщения събитие за смяна на костюмите на сцената и за действията на спрайтове.<br>• Ученикът използва блокове от Група Звук, за да композира мелодии;<br>• Ученикът идентифицира необходимостта от цикъл за повторение, за да намали броя на блоковете в скриптовете.<br>• Ученикът разширява функционалността на играта. |

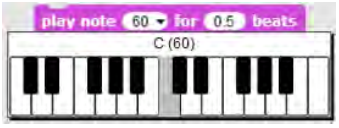
| | |
|---|---|
| **Цели, Задачи и кратко описание на дейностите** | **Кратко описание:**<br>Да се пренесем в чудния свят на Кралица Мери. Тя кани играча в двореца си да послуша музика. В балната зала нейният приятел динозавърчето Дино свири на пиано. В играта Дино свири няколко музикални тона, а играчът трябва да познае кой тон свири Дино. Ако познае получава една точка за правилен отговор, ако не познае една точка за неправилен отговор. След разпознаването на тоновете се поставя по-сложна задача: Дино свири на пианото мелодия, а играчът трябва да познае мелодията от коя песен е. За правилно позната мелодия получава 5 точки.<br>**Задача:** Учениците използват “half backed”(полуготов) файл с декорите на сцените и спрайтовете. Трябва да планират необходимите променливи, да определят какви блокове са им необходими. Да се запознаят с блоковете от групата Sound и начина на изсвирване на ноти. Да създадат скриптове, с които да се изсвирят няколко мелодии.<br>**Цел: Учениците ще научат за кодиране и възпроизвеждане на мелодии и ще подобрят своите вече придобити знания за променливи, цикли, условни блокове и операции, излъчвани събития и други събития.** |
| **Продължителност** | 90 минути |
| **Стратегия и методи на обучение** | Активно обучение (дискусии, експеримент с вече подготвена игра), обучение, основано на играл дизайн, решаване на проблеми. |
| **Форми на обучение** | Самостоятелна работа / Работа по двойки/ Презентация пред класа/групата |

| | |
|---|---|
| **Ход на урока** | (Мотивация-Въведение, Прилагане, Осмисляне и Оценка)<br>1. Учителят поставя задача за създаване на играта. Обсъжда се с какви средства може да се реши задачата. Стига се до извода, че към момента не познават възможностите за писане на код за изпълнение на музика.<br>2. Учителят демонстрира част от играта с композиране на мелодия.<br>https://snap.berkeley.edu/project?user=ddureva&project=Play_a_Piano_1<br>  <br>1. Показва кода и обяснява използването на командите от групата Sound.<br>В Snap! може да се използват звукове от вградената библиотека, файлове от компютъра или да се свирят музикални тонове на различни инструменти.<br>За избор на инструмент се използва командата:<br>  , Забележка. В Scratch инструментите са много повече.<br>Учениците тестват звука на различните инструменти.<br>2. Обяснява начина на задаване на нотите: |

Използва се командата: 

. В нея първото число задава нотата, а второто число описва за колко такта се изпълнява нотата.

Когато се щракне върху стрелката до първото число се появява клавиатура на пиано и от нея може да се избере нота. Пианото обхваща две октави



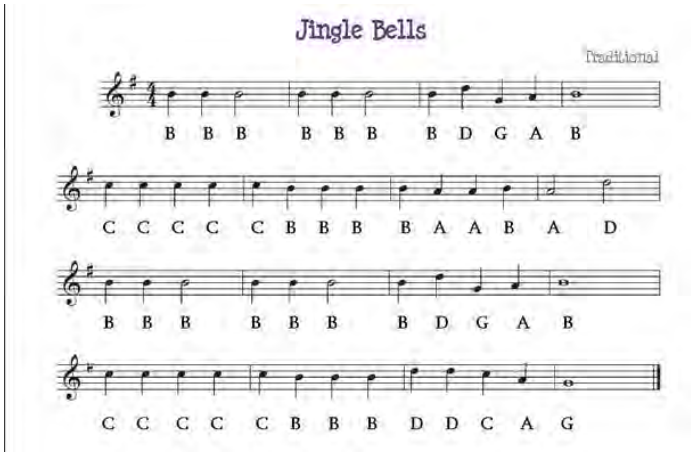

| C | C# | D | Eb | E | F | F# | G | G# | A | Bb | B | C |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 |

Продължителността на всяка нота се задава с числа 1 – цяла нота, 0.5 - половинка, 0.25 - четвъртинка. (За ученици, които не са изучавали реални числа може да се дадат чрез операцията деление – ½, ¼, 1/8 и т.н)

, 


По преценка на учителя, учениците може да експериментират с командите и да установят сами зависимостите.

3. Прави се коментар на скрипта за мелодията Jingle Bells като се използва и нотния запис на мелодията.



| | |
|---|---|
| | Поставя се задачата да се намали броя на редовете в кода, които се повтарят. Коментира се коя команда трябва да се използва. (repeat loop)<br>4. Учениците се разделят на екипи, които трябва да създадат играта поставена в началото на урока. Всеки екип обсъжда сценария на играта и описва плана на играта в листа за описание (Приложени SNAP_Program_Design_and_Planning Worksheet.docx) В описанието може да се добавят таблици за детайлно описание на действията в сцените и на спрайтовете. Може да се добави условие – динозавърът да танцува, докато свири. (В предварително подготвения файл динозавърът има няколко костюма)<br>5. Учителят може да покаже част от сценарии от файла https://snap.berkeley.edu/project?user=ddureva&project=PlayAPiano |
| **Инструменти и ресурси за учителя** | Цялата дейност е в Snap!:<br>https://snap.berkeley.edu/project?user=ddureva&project=Play_a_Piano_1<br>https://snap.berkeley.edu/project?user=ddureva&project=PlayAPiano |
| **Ресурси/материали за ученика** | https://snap.berkeley.edu/project?user=ddureva&project=Play_a_Piano_Half_backed<br>Инструкции за учениците<br>(C4G19.1_InstructionsForStudent_BG.docx) |

## Учебен сценарий 19.2 – Свири на пиано

| | |
|---|---|
| **Учебен сценарий**<br>**Име** | Свири на пиано |
| **Предишен опит в програмирането** | • Използване на цикъл repeat<br>• Използване на променливи<br>• Използване на условни блокове |
| **Резултати от обучението** | Общи цели на обучението:<br>• Условия<br>• Цикли<br>Специфични цели на обучението, ориентирани към алгоритмичното мислене:<br>• Ученикът използва цикъл с повторение за възпроизвеждане на музика<br>• Ученикът използва код, за да накара спрайтовете да реагират на въвеждането<br>• Ученикът добавя звуци към спрайт<br>• Ученикът използва код, за да промени костюма на спрайт |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Ученикът трябва да изсвири песен на пиано по дадени ноти.<br>**Задача:** Трябва да програмирате клавишите за пиано - всеки клавиш трябва да свири определен тон. На сцената трябва да се покажат два различни бутона, единият за показване на нотите, а другият за възпроизвеждане на мелодията.<br>**Цел: Учениците ще се научат как да свирят музика и да сменят костюма, като кликнат върху спрайт.** |
| **Продължителнст** | 45 минути |
| **Стратегия и методи на обучение** | Активно учене; обучение, основано на програмиране на игра; решаване на проблеми |
| **Форми на обучение** | Самостоятелна работа / Работа по двоики |
| **Ход на урока** | (Мотивация-Въведение, Прилагане, Осмисляне и Оценка) |

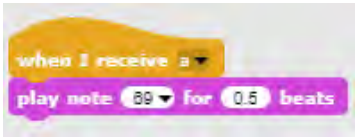
| | |
|---|---|
| | В началото на сцената се показва пиано. До пианото ще има два бутона. Щракването върху първия бутон трябва да показва нотите и думите на песента, а щракването върху втория бутон трябва да възпроизвежда мелодията, която трябва да се изсвири. Освен това до пианото ще бъде бутонът „X“, който ще рестартира проекта.<br>[Стъпка 1]<br>Отворете програмата *Play a piano.* Програмата съдържа всички фонове и спрайтове, необходими за тази задача.<br>Даден е фонът, а също и спрайтът за клавиш C и един черен клавиш.<br>Учениците трябва да дублират ключа C, да го преместят в правилната позиция и да го преименуват. Клавишите трябва да са в следния ред: C, D, E, F, G, A, B. Клавиатурата трябва да изглежда както на следващата картинка и да възпроизвежда тонове, написани под клавишите:<br>  <br><br>Дублирайте спрайт “black_key” 4 пъти, за да получите 5 черни клавиша, и ги именувайте от черен ключ 2 до черен ключ 5. Поставете нови черни клавиши между клавишите D и E, F и G, G и A и A и B.<br><br>Ако черният клавиш е скрит зад белите клавиши, използвайте следния код<br>  <br><br>Направете същото за клавиша B и ги поставете в края на клавиатурата. |

На първо място, трябва да сте сигурни, че бутонът “draggable” е отменен за всички спрайтове, за да не могат да бъдат премествани по време на игра.

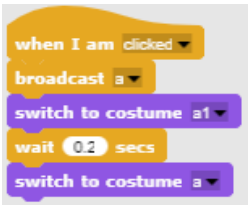

[Стъпка 2]

След това активирайте възпроизвеждането на тонове чрез натискане на спрайтове. За бутона "C" добавете блок "When I am clicked" и го оставете да излъчва съобщението "c", както следва:


Въпреки това, за да издадете звук, когато натиснете клавиш, добавете блок “When I receive c” и добавете бележка 60 за 0.5 удара.




За да подчертаете кой клавиш е натиснат, трябва временно да промените костюма на този спрайт. За да направите това, в блока "When I am clicked" сменете костюма на c1 за 0,2 секунди, след което се върнете към костюм c.




[Стъпка 3]

Повторете това кодиране за другите бели клавиши.

Задайте за другите клавиши следните бележки: D – 62, E – 64, F – 65, G – 67, A – 69, B – 71

[Стъпка 4]

Сега, за да свирите на пианото с клавиатурата, добавете блок "When c key pressed", за да натиснете клавиша с Sprite, и копирайте останалата част от кода от блока "When I am clicked".

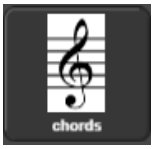

Забележете, ако клавиша "c" на клавиатурата е задържан, звукът ще се повтаря, докато клавишът е натиснат. Това се случва, защото съобщението "c" се излъчва многократно. За да спрете излъчването на съобщение, в края на кода добавете блок "изчакайте до""wait until" от палетата за контрол.


За да завършите излъчването на съобщение, използвайте оператора "not" и добавете блок "key c pressed".


Направете същото за останалите бели клавиши.

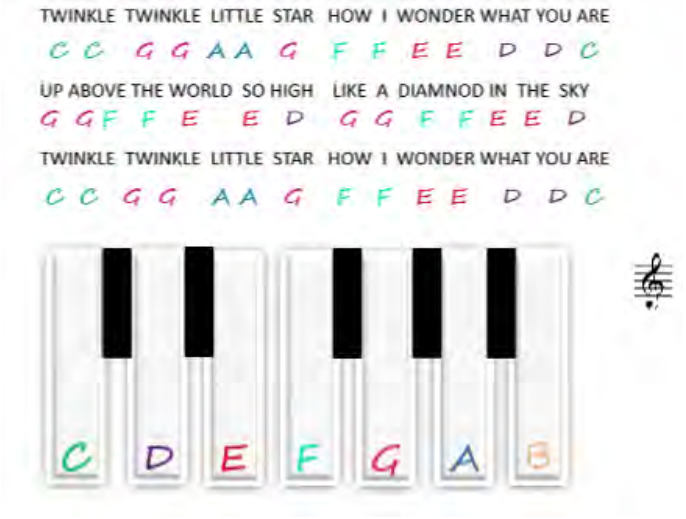
[Стъпка 5]

Време е да програмирате петолинието, ключът сол и нотите, които ще се изпълняват:




За да покажете нотите, активирайте излъчването на съобщението "chords", когато натиснете бутона.


Импортирайте нов костюм "chords" за тази сцена.




И накрая, добавете код, който позволява на сцената да промени костюма на "chords", когато получи съобщение "chords".


[Стъпка 6]

Намерете спрайт с костюм нота. Сега е време да програмирате клавиша за нота за възпроизвеждане на песента, която трябва да се повтори.


Кодът е написан за първите два стиха от песента, а вие трябва да напишете кода за останалите стихове. Това е същата песен, както е показана в нотите.


when I am clicked
repeat 2
play note 80 for 0.5 beats
repeat 2
play note 87 for 0.5 beats
repeat 2
play note 89 for 0.5 beats
play note 87 for 0.5 beats
wait 0.1 secs
repeat 2
play note 85 for 0.5 beats
repeat 2
play note 84 for 0.5 beats
repeat 2
play note 82 for 0.5 beats
play note 80 for 0.5 beats
wait 0.1 secs
repeat 2
play note 87 for 0.5 beats
play note 87 for 0.5 beats
play note 85 for 0.5 beats
play note 85 for 0.5 beats
play note 84 for 0.5 beats
play note 84 for 0.5 beats
play note 82 for 0.5 beats
wait 0.1 secs
repeat 2
play note 80 for 0.5 beats
repeat 2
play note 87 for 0.5 beats
repeat 2
play note 89 for 0.5 beats
play note 87 for 0.5 beats
wait 0.1 secs
repeat 2
play note 85 for 0.5 beats
repeat 2
play note 84 for 0.5 beats
repeat 2
play note 82 for 0.5 beats
play note 80 for 0.5 beats

[Стъпка 7]

След това програмирайте спрайта "X" по начин, който ще нулира проекта (без забележките), както следва:

- задайте размера му на 50%.
- активирайте излъчването на "blank" съобщение при натискане на бутона.


В края добавете към сцената блок “When I receive”, за да промените костюма към „blank“, след като получите съобщението „blank“.

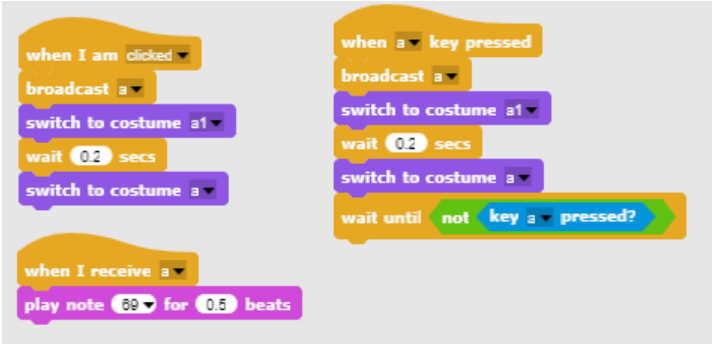

[Допълнителни задачи]

Учениците могат да добавят допълнителни задачи според техните предпочитания или могат да следват задачите по-долу:

- Дублирайте спрайт нотата (и променете позицията й на фона) и напишете програма за друга песен.
- Добавете фон с акорди за новата песен.

[Финален код]

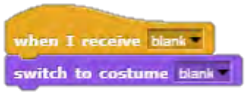
А клавиш




Клавиш цигулка


when I am clicked
repeat 2
play note 60 for 0.5 beats
repeat 2
play note 67 for 0.5 beats
repeat 2
play note 69 for 0.5 beats
play note 67 for 0.5 beats
wait 0.1 secs
repeat 2
play note 65 for 0.5 beats
repeat 2
play note 64 for 0.5 beats
repeat 2
play note 62 for 0.5 beats
play note 60 for 0.5 beats
wait 0.1 secs
repeat 2
play note 67 for 0.5 beats
play note 67 for 0.5 beats
play note 65 for 0.5 beats
play note 65 for 0.5 beats
play note 64 for 0.5 beats
play note 64 for 0.5 beats
play note 62 for 0.5 beats
wait 0.1 secs
repeat 2
play note 60 for 0.5 beats
repeat 2
play note 67 for 0.5 beats
repeat 2
play note 69 for 0.5 beats
play note 67 for 0.5 beats
wait 0.1 secs
repeat 2
play note 65 for 0.5 beats
repeat 2
play note 64 for 0.5 beats
repeat 2
play note 62 for 0.5 beats
play note 60 for 0.5 beats

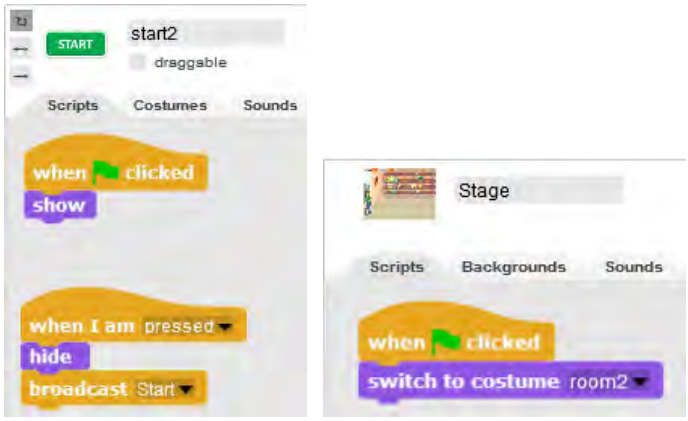
Нота

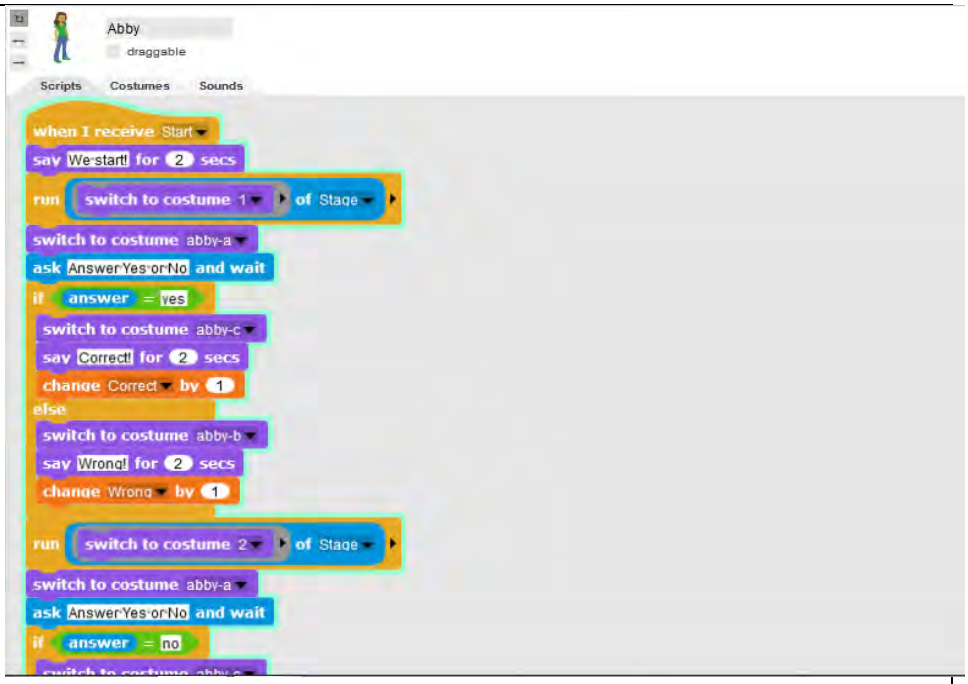
| | |
|---|---|
| | X<br>  <br><br>  <br><br>Сцената<br>  <br><br>  <br> |
| **Инструменти и ресурси за учителя** | Цялата дейност в Snap!:<br>https://snap.berkeley.edu/project?user=ifrankovic&project=Play%20a%20Piano |
| **Ресурси/материали за ученика** | Полуготов файл в Snap!:<br>https://snap.berkeley.edu/project?user=ifrankovic&project=Play%20Piano (27.1.2020)<br>Изображения:<br>• Sprite изображения:<br>  ◦ a.png, a1.png<br>  ◦ b.png, b1.png<br>  ◦ violin_key.png<br>• Backgrounds: notes.png<br>• Инструкции за учениците (C4G19.2_InstructionsForStudent_BG.docx) |

## Учебен сценарий 20 - Тест

| | |
|---|---|
| **Учебен сценарий Име** | Тест |
| **Предишен опит в програмирането** | • Показване и скриване на спрайтове<br>• Употреба на променливи за преброяване на събраните точки<br>• Използване на цикъл forever<br>• Използване на условни блокове<br>• Използване на оператори за сравнение<br>• Използване на сензорни блокове<br>• Промяна на сцена |
| **Резултати от обучението** | Общи цели на обучението:<br>• Променливи<br>• Условия<br>• Цикъл<br>• Сензорни блокове<br>Специфични цели на обучението, ориентирани към алгоритмичното мислене:<br>• Учениците използват условия за да преценят отговора - правилно или грешно<br>• Учениците използват блокове за смяна на костюма на сцената<br>• Учениците използват променливи за броене на точки<br>• Учениците използват логически операции<br>• Учениците използват външен графичен редактор за подготовка на сложни фонове на сцените. |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** Помогнете на вашия учител да тества вашите знания за Snap!, като създадете игра, базирана на Quest(Въпросник), за да тествате командите в Snap!<br>**Задача:** Трябва да проучите примерна игра, затова изберете от полуготовите игри, намерете или създайте свой собствен спрайт (Sprite), който да задава въпроси, изберете от полуготовите игри или проектирайте фона на началния екран и сценичните фонове с подходящи въпроси, променете и разширете кодовете в теста по отношение на въпросите. |

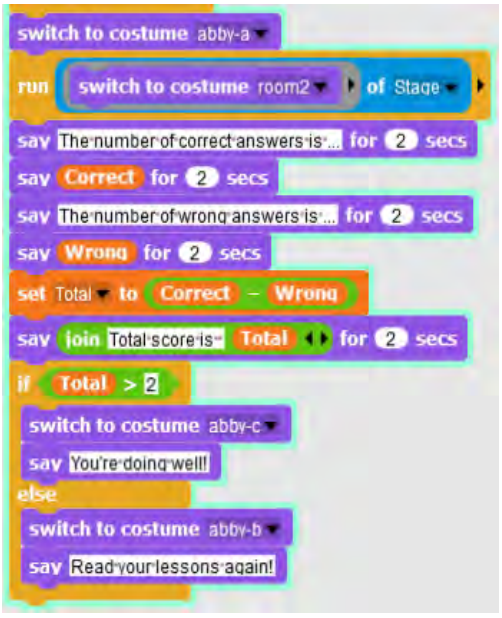
| | |
|---|---|
| | **Цел: Учениците ще подобрят знанията си и ще разширят сценария на играта с нов фон, код и променлив код по отношение на новите сцени.** |
| **Продължителнст** | 90 минути |
| **Стратегия и методи на обучение** | Активно учене (дискусии, работа с полуготови игри); обучение, основано на програмиране на игра; решаване на проблеми |
| **Форми на обучение** | Самостоятелна работа / Работа по двоики / Директен контакт с цял клас |
| **Ход на урока** | (Мотивация-Въведение, Прилагане, Осмисляне и Оценка)<br>1. Трябва да създадете игра-тест, за да тествате знанията си по програмиране.<br>2. Възлага се на учениците да играят играта и да опишат с думи: Колко сценични декорации наблюдават и колко спрайтове (герои)? Как започва играта? Колко променливи се използват, как се именуват, за какво се използват? Какво се случва, когато отговорът е верен/грешен? Как са представени въпросите в теста?/ индивидуална работа или работа по двойки по преценка на учителя/<br>3. Трябва да използвате алгоритъма за задаване и отговор на въпроси:<br>1. преминаване към сценичен костюм (съдържа въпрос);<br>2. задаване на костюм на Аби за задаване на въпрос;<br>3. Аби казва - Отговорете с Да или Не;<br>4. Играчът въвежда отговор - Да или Не;<br>5. Ако отговорът е верен, Аби казва "Правилен отговор" и броят на верните отговори се увеличава; В противен случай Аби казва „Грешен отговор“ и броят на грешните отговори се увеличава.<br>4. Какво се случва след като отговорите на всички въпроси?<br>1. Смяна на костюм / фон на сцената; |

| | |
|---|---|
| | 2. Аби посочва броя на правилните и грешни отговори и дава оценка.<br>5. Изследване на кодовете в играта/Актуализиране на стари знания /индивидуална и фронтална дейност/<br>Коментират се командите за встъпване в диалог с потребителя, за промяна на сценичния декор и костюма на персонажа, условни команди. Разглеждат се кодовете на всеки знак. Създаването на променлива е коментирано.<br>  <br><br>  <br> |




Коментират се ситуациите, когато верният отговор е „да“ и когато верният отговор е „не“.

Кодът за оценка е описан подробно, като е използвана променливата Total.

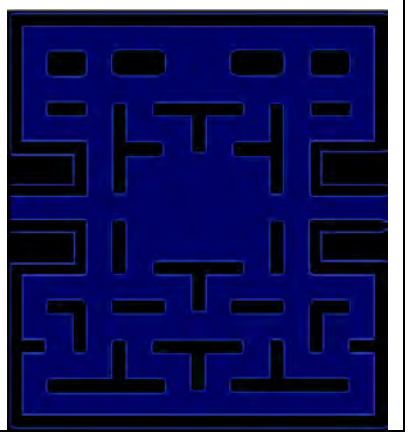



Обсъжда се начинът на проектиране на сцената за отделни въпроси.

| | |
|---|---|
| | Тъй като в Snap! не е възможно да се пише текст в костюми и декори, е необходимо да се използва външен графичен редактор. Друга възможност е да използвате MS Powerpoint, за да създадете въпроса и да експортирате съответното текстово поле в графичен формат.<br>Добавяне на костюм в Snap! Може би преработен.<br>1. Разделяне на групата на отбори от 2 или 3 ученика.<br>2. Публикуване на темата за тестовите въпроси. Например - Използване на променливи; Цикли; Движение, Сензори, аритметични и логически операции.<br>3. Проектиране на сцени с въпроси по тема от съответния екип. Ако е необходимо, учителят съветва учениците по съдържанието на въпросите. Въпросите се обсъждат и всеки екип създава сцена за поне два въпроса.<br>4. Създаване на кода. На учениците се предоставя полуготов файл с костюми на сцената и спрайтове. Те също могат да създадат свой собствен файл, ако желаят. Работата се извършва по аналогия с теста на модела. |
| **Инструменти и ресурси за учителя** | Цялата дейност в Snap!:<br>https://snap.berkeley.edu/project?user=ddureva&project=test2<br>Цялата дейност в Scratch:<br>● Дурева Д., М. Касева, Г. Тупаров, Компютърно моделиране, 4. клас, Просвета, 2018, София (Dureva, D., M. Kaseva, G. Tuparov, Kompyutarno modelirane, 4. klas, Prosveta, 2019, Sofia) |
| **Ресурси/материали за ученика** | ● Полуготов файл в Snap!:<br>https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_20_test_en_tmp<br>● Инструкции за учениците (C4G20_InstructionsForStudent_BG.docx) |

## Учебен сценарий 21 - Опростена игра PACMAN

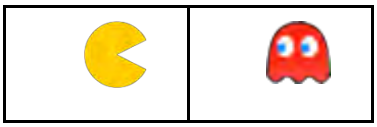
| | |
|---|---|
| **Учебен сценарий Име** | Опростена игра PACMAN |
| **Предишен опит в програмирането** | • използване на условни блокове,<br>• кодиране на множество обекти,<br>• сензорни блокове,<br>• цикли (forever, repeat until),<br>• синхронизиране на действията чрез съобщения,<br>• използване на произволни числа |
| **Резултати от обучението** | Общи цели на обучението:<br>• Движение на обект базирано на събития,<br>• Засичане (разпознаване) на един цвят,<br>• Употреба на булеви стойности в логически изрази,<br>• Определяне, диференциране, динамично проверяване и реагиране на две различни състояния на играта.<br>Специфични цели на обучението, ориентирани към алгоритмичното мислене:<br>• Ученикът осъществява движение на обект с клавишите със стрелки, използвайки събития и отчита ограниченията,<br>• Ученикът използва сензорен цветен блок, за да получи булева стойност за четене на едноцветни сензори,<br>• Ученикът осъзнава, че състоянието на обекта може да бъде изразено с цвета на обекта до който се докосва,<br>• Ученикът прави разлика между две различни състояния и знае как да ги изрази с логически изрази,<br>• Ученикът осъзнава, че позицията на обекта се променя динамично и използва цикъл forever за многократна проверка на текущото състояние,<br>• Ученикът използва if-else, за да даде различни отговори въз основа на текущата позиция на обекта. |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:** В този курс ще програмирате игра, в която главният герой трябва да вземе произволно разположени обекти, в случая звезди и ще бъде преследван от призрак.<br>**Задача:** Движението на главния герой трябва да бъде програмирано, така че той да се движи в лабиринт. След това трябва да програмирате обектът звезда, който ще се клонира при |

| | |
|---|---|
| | стартиране на играта и ще се появява на произволно ново място всеки път, когато героят го вземе. Трябва да съхранявате броят на събраните звезди и да завършите играта, когато играчът събере 20 звезди. За да направите играта по-интересна, можете да програмирате зъл призрак, който на случаен принцип ще се движи в лабиринта. Ако играчът докосне призрак, играта свършва.<br>**Цел: Учениците ще приложат своите знания за движение в лабиринт с помощта на сензорен цветен блок. Те ще научат концепцията за клониране на обект с ограничения за позицията и как да създадат много прост персонаж, който не е играч, с опция за произволно движение.** |
| **Продължителнст** | 90 минути |
| **Стратегия и методи на обучение** | Активно учене; обучение, основано на програмиране на игра; решаване на проблеми |
| **Форми на обучение** | Директно преподаване<br>Самостоятелна работа / Работа по двоики/Работа в групи |
| **Ход на урока** | (Мотивация-Въведение, Прилагане, Осмисляне и Оценка)<br>Играчът трябва да събират произволно разположени звезди, докато е преследван от червен призрак.<br>Ако той и призракът се сблъскат, играта свършва, ако пък успее да съберете 20-те звезди, преди това играчът печели.<br>[Стъпка 1]<br>Инструктираме учениците да проектират лабиринт, в който зоната, където играчът има право да се движи, е от един цвят (например син) и стени, които спират движението на играча, оцветени в някакъв друг цвят (например черен). За да спестим  |

време, можем предварително да подготвим фоновата картина на лабиринта.

[Стъпка 2]

Учениците трябва да нарисуват pacman и червения призрак. За звезда можем просто да нарисуваме кръг в Snap!:



[Стъпка 3]

За да накарате Pacman да се движи, можете да използвате различни варианти. Един от тях е използването на система за събития за откриване на това, кой клавиш е натиснат, наляво, надясно, нагоре или надолу. След всяко от тези събития, трябва да тествате дали той докосва цвета на зоната, по която му е позволено да се движи. В този случай, най-напред той се обръща в тази посока и прави ход. Но ако докосне цвета на стените, трябва да се придвижи назад, защото в противен случай ще се забие в стената.

[Стъпка 4]

Следващата задача е да програмирате звездите. Звездите ще са еднакви и ще има много от тях. В този случай е по-добре, вместо да се направят множество идентични обекти (в нашия случай 20), да се направи един обект и след това да се клонира. В началото на играта първият клонинг ще се появи на случаен принцип в лабиринта, след това, когато го съберете, той ще изчезне и ще бъде създаден нов на различно произволно място. За да създадете първия клонинг в началото на играта, поставете този код на сцената:


А звездата трябва да бъде скрита when the green flag is clicked в началото на играта.

За да намерите подходящо произволно място за звездата, трябва да спазвате определени ограничения. Ако звездата е поставена върху стена, няма да можете да я достигнете, което означава, че не трябва да я поставяте там.

Стратегията за това е следната:

1. Трябва да намерите случайна (x, y) позиция на клонинга (звездата). И двете координати, и x и y са в един и същи интервал [-140, 140]. Така че избирате произволно число от този интервал и за двете.
2. След това проверявате дали този клонинг докосва цвета на стената. Ако е така, то местоположението му не е правилно.
3. Ако местоположението е правилно, клонингът трябва да се покаже и да проверите чрез цикъл forever дали не се получава сблъсък с играча.

4. Ако местоположението не е правилно, създаваме нов клонинг и изтриваме този.
5. За да преброите събраните клонинги, трябва да използвате общ брояч на звезди, които трябва да бъде дефиниран извън клонингите, например на играча.

Това може да стане чрез излъчване на съобщение, че сблъсъкът е настъпил. След това можете да го изтриете.


[Стъпка 5]

Сега ще програмирате призрак. Той трябва да се движи на случаен принцип из лабиринта и трябва да променя посоката си, когато се блъсне в стена. За да направите движението му случайно, трябва да го програмирате да се движи в произволна посока след удара.

В Snap! посоките се изразяват в градуси:

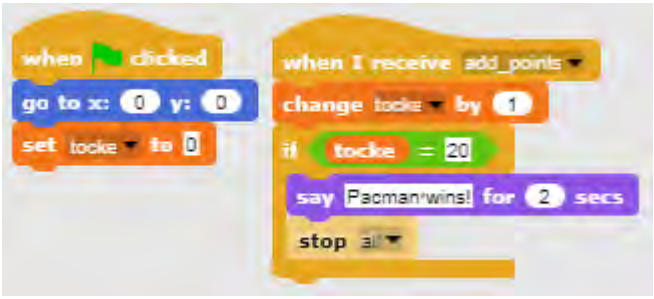
0 градуса - НАГОРЕ

180 градуса - НАДОЛУ

90 градуса - ДЯСНО

270 градуса - НАЛЯВО

С други думи, ако изберете произволно числото от 0 до 3 и го умножите по 90, ще получите произволна посока!

| | |
|---|---|
| | Той трябва да се движи, докато не се сблъска с Pacman. Тогава играта свършва.<br>  <br><br>[Стъпка 6]<br>Сега трябва да програмирате кога ще спечелите играта.<br>Това ще стане, когато съберете 20 звезди. Имате брояч на звездите вътре в Pacman скрипта. В началото го инициализирайте на 0 и след това увеличавайте стойността му с 1 всеки път, когато клонингът изпрати съобщение, че вие (като играч) сте го събрали.<br>Ако броячът стигне до 20, PACMAN печели и играта свършва.<br>  <br> |
| **Инструменти и ресурси за учителя** | • Цялата дейност в Snap!:<br>https://snap.berkeley.edu/project?user=zapusek&project=pacman_clone |

| | |
|---|---|
| | • Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.<br>• Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Ресурси/материали за ученика** | • Примерен шаблон в Snap!: https://snap.berkeley.edu/project?user=zapusek&project=pacman_template<br>• Инструкции за ученика (C4G21_InstructionsForStudent_BG.docx) |

# ПРИЛОЖЕНИЕ 2. КОДОВЕ ЗА СЦЕНАРИИТЕ В ИГРОВАТА СРЕДА

## Основни сценарии

### ENGLISH Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 1 | **Introduction to Snap! Interface**<br>Students add a new sprite, add a costume to the sprite, edits the costume, and deletes one of them. Student creates a new background to the stage, edits it, and deletes unwanted ones. | **starting** |
| 2 | **Discover Snap! : move a spite**<br>Help the students to discover the Snap! interface and code their first sprite so that it moves and speaks. | **dispubeng** |
| 3 | **Moving around the stage**<br>Students learn how to move the sprite in x and y direction on the stage, code an easy program to solve the tasks given, learn how to turn the sprite in a different direction. | **monkey** |
| 4 | **Changing costumes and turning**<br>Students learn how to change the sprite's costume to make an animation by changing between different types of rotation. | **Dancer** |
| 5 | **Sounds of the farm**<br>Students learn how to program a simple game in which player can recognize the sounds of animals by pressing certain keys. | **farm** |
| 6 | **Chameleon's summer vacation**<br>Program a simple game in which an object will change its costume based on the color of the background | **chameleoneng** |
| 7 | **Helping Prince and Princess to find their animals**<br>A girl has to help the Princess find her cat and the Prince find his dog by avoiding that the animals can meet each other during their path. | **finding** |
| 8 | **Drawing with a chalk**<br>Students will be introduced into drawing different shapes with a code. They will learn to use loop repeat for shorten the code and to change a background. | **chalk** |
| 9 | **Picking up trash and cleaning the park**<br>Students will learn how to use variables and how to duplicate a block of code or even a whole sprite. | **cleaning** |
| 10 | **Feeding the cats**<br>Students will be introduced to the concept of multiple variable random value assignment inside a loop and how it is different from when we do it outside a loop. They will also learn about how to get, test and count correct player inputs. | **feeding** |
| 11 | **Guessing the number of cats in a shelter**<br>Students will be introduced to "repeat until" loop and how to set the condition to implicitly track the condition that stops the game. They will also learn how to use variable in a different situations: to set a random value, as a counter or to get the players input. | **shelter** |

## SLOVENIAN Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 1 | **Uvod v Snap!**<br>Učenec doda nov lik, doda obleko, uredi obleko in izbriše obleko/lik. Učenec ustvari novo ozadje, ga uredi, zbriše odvečna ozadja. | **uvod_slo** |
| 2 | **Razišči Snap!: premikaj lik**<br>To uro boš spoznal/a, kako z bloki likom naročimo naj se premikajo po odru in govorijo. | **razisci_slo** |
| 3 | **Premikanje po odru**<br>Učenec se nauči, kako premikati lik v x in y smeri, kako napisati preprost program ter kako obrniti lik v drugo smer | **premikanje_slo** |
| 4 | **Menjava obleke in obrat**<br>Učenec se nauči, kako liku zamenjati obleko in kako narediti animacijo z obračanji lika | **obleka_slo** |
| 5 | **Zvoki na kmetiji**<br>Učenec se nauči programiranja preproste igre, v kateri lahko igralec s pritiskom na določene tipke prepozna zvoke živali | **kmetija_slo** |
| 6 | **Kameleon na počitnicah**<br>Izdelaj preprosto igro, v kateri bo glavni objekt spreminjal svojo obleko glede na barvo ozadja na njegovi trenutni poziciji. | **kameleon_slo** |
| 7 | **Pomagaj princu in princeski najti svoje živali**<br>Dekle mora pomagati princesi poiskati svojo mačko in princu svojega psa. Pri tem se mora izogniti srečanju med živalmi. | **iskanje_zivali_slo** |
| 8 | **Risanje s kredo**<br>Učenec se seznani z risanjem različnih oblik s kodo. Nauči se uporabljati zanko ponovi za krajšanje kode ter kako zamenjati ozadje. | **kreda_slo** |
| 9 | **Pobiranje smeti in čiščenje parka**<br>Učenec se nauči uporabljati spremenljivke in kako podvojiti blok kode ali celotno kodo lika. | **ciscenje_parka_slo** |
| 10 | **Nahrani mucke**<br>Učenec se seznani s konceptom dodeljevanja več spremenljivk z naključno vrednostjo znotraj zanke in kakšna je razlika, če vrednost dodelimo zunaj zanke. Spoznali bodo tudi, kako pridobiti in preveriti igralčev odgovor ter kako šteti pravilne odgovore. | **mucke_slo** |
| 11 | **Mačje zavetišče**<br>Učenec se seznani z zanko "ponavljaj dokler" in kako ustvariti pogoj za implicitno sledeje pogoju, ki konča z igro. Nauči se tudi, kako uporabljati spremenljivko v različnih situacijah: nastaviti naključno vrednost, kot števec ali kot igralčev vnos. | **zavetisce_slo** |

## ITALIAN Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 1 | **Snap!: introduzione**<br>Lo studente impara ad aggiungere un nuovo sprite, a modifica il costume e ad eliminarne uno di essi. Impara a crea un nuovo sfondo per il palco, lo modifica e cancella quelli indesiderati. | **Primipassi** |
| 2 | **Scoprire Snap!**<br>Imparare a programmare un gioco semplice dove l'oggetto cambia il proprio colore secondo il colore dello sfondo | **Snap!** |
| 3 | **Muoversi sullo "stage"**<br>Il discente impara a spostare il suo Sprite in direzione x e y sullo "stage"; a preparare un semplice programma per risolvere i compiti assegnati; a far girare il suo Sprite in diverse direzioni. | **stage** |
| 4 | **Cambiare il costume e creare rotazioni**<br>Lo studente impara a cambiare il costume dello sprite per realizzare un'animazione. | **ballerina** |
| 5 | **I suoni di una fattoria**<br>Gli studenti imparano come programmare un gioco semplice in cui il giocatore può riconoscere i suoni degli animali premendo determinati tasti. | **fattoria** |
| 6 | **Vacanze estive di un Camaleonte**<br>Imparare a programmare un gioco semplice dove l'oggetto cambia il proprio colore secondo il colore dello sfondo | **camaleonte** |
| 7 | **Alla ricerca degli animali del Principe e della Principessa!**<br>La ragazza deve aiutare la principessa a trovare il suo gatto e il principe a trovare il suo cane evitando che gli animali possano incontrarsi durante il loro. | **labirinto** |
| 8 | **Disegnare con un gesso**<br>Gli studenti impareranno a disegnare forme diverse con un codice e ad usare la ripetizione ciclica per abbreviare il codice e cambiare lo sfondo. | **gesso** |
| 9 | **Raccogliere la spazzatura e pulire il parco**<br>Gli studenti impareranno ad usare le variabili e a duplicare un blocco di codice o anche un intero Sprite. | **spazzatura** |
| 10 | **Dare da mangiare ai gatti**<br>Gli studenti impareranno il concetto di assegnazione di più valori casuali, variabili all'interno e al di fuori di un ciclo. Impareranno anche come ottenere, testare e contare gli input corretti immessi dal giocatore. | **gatto** |
| 11 | **Indovinare il numero di gatti in un rifugio**<br>Gli studenti impareranno ad utilizzare il ciclo "ripeti fino a" e ad impostare la condizione per interrompere il gioco. Impareranno anche ad usare la variabile in situazioni diverse. | **rifugio** |

## CROATIAN Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 1 | **Uvod u sučelje alata Snap!**<br>Učenik dodaje nove objekte, dodaje kostime objektima, uređuje kostime te ih briše. Učenik stvara nove pozadine na pozornici, uređuje ju, te neželjene briše. | **start_hr** |
| 2 | **Otkrijte Snap!: kretanje sprite**<br>Učenici otkriva gdje pronaći programske blokove i povezati ih u niz. Učenici će naučiti kako micati objekt i omogućiti da objekt govori. | **disc_cro** |
| 3 | **Kretanje po pozornici**<br>Učenici će vidjeti kako izraditi program u kojem će se objekt kretati u smjeru x, te izraditi program u kojem će se objekt kretati u smjeru y. | **kretanje** |
| 4 | **Mijenjanje kostima i okretanje**<br>Učenik uči kako promijeniti kostim objekta te kako napraviti animaciju. Također uči kako se između njih može mijenjati različite vrste rotacije objekta. | **ples** |
| 5 | **Zvukovi s farme**<br>Učenici će se upoznati kako programirati jednostavnu igru u kojoj igrač može prepoznati zvukove životinja pritiskom na određene tipke. | **farma** |
| 6 | **Kameleonov ljetni odmor**<br>Učenici će biti upoznati s blokom naredbi osjeta boja i kako ga koristiti u logičkim izrazima da bi razlikovali dinamično promjenjiva stanja igre i dali prave odgovore. | **chameleon_cro** |
| 7 | **Pomaganje princu i princezi u pronalasku njihovih životinja**<br>Učenici će se upoznati s crtanjem pokretom tipke. Uz to će naučiti kako koristiti uvjete kojima će spriječiti da se objekt kreće po cijelome ekranu. | **finding_hr** |
| 8 | **Crtanje s kredom**<br>Učenici će se upoznati s crtanjem različitih oblika pomoću kôda. Naučit će koristiti petlju ponavljaj kako bi smanjili veličinu kôda i promijeniti pozadinu. | **kreda** |
| 9 | **Skupljanje otpada i čišćenje parka**<br>Učenici će se upoznati kako koristiti varijable i kako kopirati blok kôda ili čak cijeli objekt. | **park** |
| 10 | **Hranjenje mačaka**<br>Učenici će biti upoznati sa konceptom dodjele slučajne vrijednosti varijabli unutar petlje te će razlikovati kada navedeno napravimo van petlje. Naučiti će kako dobiti, testirati i izbrojati ispravne unose igrača. | **hranjenje** |
| 11 | **Pogađanje broja mačaka u skloništu**<br>Učenici će se upoznati s petljom ponavljaj dok i kako postaviti uvjet koji zaustavlja igru. Također će naučiti kako koristiti varijable u različitim situacijama: za pohranjivanje nasumične vrijednosti, kao brojač ili za pohranjivanje vrijednosti koju upiše igrač. | **pogodi** |

## BULGARIAN Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 1 | **Увод в Snap!** Ученикът добавя нов спрайт, добавя костюм към спрайта, редактира костюма и изтрива едино от двете. Ученикът създава нов фон на сцената, редактира го и го изтрива. | **start_bg** |
| 2 | **Да разучим Snap!: Движещ се спрайт** Помогнете на учениците да разучат интерфейса на Snap! и да съставят код за първия им спрайт, така че той да се движи и говори. | **dissnap_bg** |
| 3 | **Движейки се по сцената** ченикът се научава как да движи спрайта в x и y посока на сцената, създава лесна програма за решаване на задачите и се научава как да завърти спрайт в различна посока. | **monkey_bg** |
| 4 | **Смяна на костюми и завъртане** Ученикът се учи как да промени костюма на спрайт при завъртане, за да направи анимация. | **dancer_bg** |
| 5 | **Звуци от фермата** Учениците научават как да програмират игра, в която играчът може да разпознава звуците на животните чрез натискане на определени клавиши. | **sounds_bg** |
| 6 | **Лятната ваканция на хамелеона** Програмирайте проста игра, в която обект да променя костюма си в зависимост от цвета на фона. | **cham_bg** |
| 7 | **Помогнете на принца и принцесата да намерят своите домашни любимци** Момиче трябва да помогне на принцесата да намери котката си, а на принцът да намери кучето си, като трябва да избегне възможността животните да се срещнат по пътя. | **find_bg** |
| 8 | **Рисуване с тебешир.** Учениците ще бъдат запознати с рисуването на различни фигури(форми) използвайки код. Те ще се научат да използват цикъл с повторение за намаляне обема на кода и за промяна на фона. | **draw_bg** |
| 9 | **Събиране на боклук и почистване на парка** Учениците ще научат как да използват променливи и как да копират блок с код или дори цял спрайт. | **clean_bg** |
| 10 | **Нахранете котките.** Учениците ще бъдат въведени в концепцията за присвояване на случайно число на няколко променливи в цикъл, както и каква е разликата при използването им вътре и извън цикъл. Те също така ще научат как за да зададат, тестват и изброят правилно въведеното от играча. | **cats_bg** |
| 11 | **Познайте броя на котките в приюта.** Учениците ще бъдат запознати с цикъл "repeat until" и как да зададат условието, при което спира играта. Те също така ще се научат как да използват променливи в различни ситуации: за задаване на произволна стойност, като брояч или като стойност въведена от играча. | **shelter_bg** |

## GREEK Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 1 | **Εισαγωγή στο Snap!.** Ο μαθητής προσθέτει ένα νέο στοιχείο, προσθέτει ένα κοστούμι στο στοιχείο, επεξεργάζεται το κοστούμι και διαγράφει ένα από αυτά. Ο μαθητής δημιουργεί ένα νέο φόντο στη σκηνή, το επεξεργάζεται και διαγράφει τα ανεπιθύμητα. | -- |
| 2 | **Ανακαλύψτε το Snap! : μετακινήσε μία εικόνα.** Βοήθησε τους μαθητές να ανακαλύψουν την πλατφόρμα προγραμματισμού Snap! και να κωδικοποιήσουν την πρώτη τους εικόνα ώστε να κινείται και να μιλάει. | **SampleCourse1Greek** |
| 3 | **Ας μετακινηθούμε γύρω από τη σκηνή.** Ο μαθητής μαθαίνει πώς να μετακινεί ένα στοιχείο σε κατεύθυνση x και y στη σκηνή, δημιουργεί ένα εύκολο πρόγραμμα για την επίλυση των καθηκόντων του, μαθαίνει πώς να γυρίζει το στοιχείο σε διαφορετική κατεύθυνση. | -- |
| 4 | **Αλλαγή κοστουμιών και στροφή.** Ο μαθητής μαθαίνει πώς να αλλάζει το κοστούμι του στοιχείου για να κάνει ένα κινούμενο σχέδιο αλλάζοντας μεταξύ διαφορετικών τύπων περιστροφής. | -- |
| 5 | **Ήχοι φάρμας.** Οι μαθητές μαθαίνουν πώς να προγραμματίζουν ένα απλό παιχνίδι στο οποίο ο παίκτης μπορεί να αναγνωρίζει τους ήχους των ζώων πατώντας συγκεκριμένα κουμπιά. | -- |
| 6 | **Καλοκαιρινές διακοπές του Χαμαιλέοντα**<br>Προγραμματίστε ένα απλό παιχνίδι στο οποίο ένα αντικείμενο θα αλλάξει το κοστούμι του με βάση το χρώμα του φόντου | **SampleCourse2Greek** |
| 7 | **Βοηθώντας τον Πρίγκιπα και την Πριγκίπισσα να βρουν τα ζώα τους.** ο κορίτσι πρέπει να βοηθήσει την Πριγκίπισσα να βρει τη γάτα της και τον Πρίγκιπα να βρει τον σκύλο του, αποφεύγοντας τα υπόλοιπα ζώα που μπορεί να συναντήσουν κατά τη διάρκεια της πορείας τους. | -- |
| 8 | **Ζωγραφίζοντας με κιμωλία.** Οι μαθητές θα μάθουν να ζωγραφίζουν διαφορετικά σχήματα με κώδικα. Θα μάθουν να χρησιμοποιούν βρόχους επανάληψης για να συντομεύσουν τον κώδικα και να αλλάξουν φόντο. | -- |
| 9 | **Μαζεύοντας σκουπίδια και καθαρίζοντας το πάρκο.** Οι μαθητές θα μάθουν πώς να χρησιμοποιούν μεταβλητές και πώς να αντιγράφουν ένα κομμάτι κώδικα ή ακόμα και ένα ολόκληρο στοιχείο. | -- |
| 10 | **Ταΐζοντας τις γάτες.** Οι μαθητές θα μάθουν την έννοια της πολλαπλής ανάθεσης τυχαίας τιμής σε μεταβλητές μέσα σε ένα βρόχο και πώς αυτό είναι διαφορετικό αν το κάνουμε εκτός βρόχου. Θα μάθουν επίσης για τον τρόπο λήψης, δοκιμής και μέτρησης των σωστών εισόδων από τον παίκτη. | -- |
| 11 | **Μαντέψτε τον αριθμό των γατιών στο καταφύγιο.** Οι μαθητές θα μάθουν το βρόχο «επανάληψη έως» και πώς να ρυθμίσουν τη συνθήκη ώστε να παρακολουθούν τη συνθήκη που σταματά το παιχνίδι. Θα μάθουν επίσης πώς να χρησιμοποιούν τη μεταβλητή σε διαφορετικές καταστάσεις: για να ορίσουν μια τυχαία τιμή, ως μετρητή ή για να πάρουν είσοδο από τον παίκτη. | -- |

## PORTUGUESE Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 1 | **Introdução à interface Snap!**<br>Os alunos adicionam um novo sprite, adicionam um traje ao sprite, editam o traje e excluem um deles. O aluno cria um novo plano de fundo para o palco, edita-o e exclui os indesejados**.** | **starting_PT** |
| 2 | **Descubra o Snap! : mover um Sprite**<br>Os alunos descobrem a interface do Snap! e codificam o seu primeiro sprite para que ele se mova e fale. | **dispubeng_PT** |
| 3 | **Movendo-se pelo palco**<br>Os alunos aprendem como mover o sprite nas direções x e y do palco, codificam um programa fácil para resolver as tarefas dadas e aprendem a virar o sprite em uma direção diferente. | **monkey_PT** |
| 4 | **Mudando de traje e girando**<br>Os alunos aprendem a mudar o traje do sprite para fazer uma animação e alternam entre os diferentes tipos de rotação. | **dancer_PT** |
| 5 | **Sons da quinta**<br>Os alunos aprendem a programar um jogo simples no qual o jogador pode ouvir os sons dos animais pressionando certas teclas. | **farm_PT** |
| 6 | **Férias de verão do camaleão**<br>Os alunos programam um jogo simples em que um objeto muda de traje de acordo com a cor do fundo | **chameleoneng_PT** |
| 7 | **Ajudando o príncipe e a princesa a encontrar seus animais**<br>Uma menina tem que ajudar a Princesa a encontrar seu gato e o Príncipe a encontrar seu cachorro, evitando que os animais se cruzem durante o caminho. | **finding_PT** |
| 8 | **Desenho com giz**<br>Os alunos aprendem a desenhar formas diferentes com um código. Aprendem a usar ciclos para encurtar o código e alterar um plano de fundo. | **chalk_PT** |
| 9 | **Recolher o lixo e limpar o parque**<br>Os alunos aprendem a usar variáveis, a duplicar um bloco de código e um sprite. | **cleaning_PT** |
| 10 | **Alimentando os gatos**<br>Os alunos aprendem o conceito de atribuição de valores aleatórios a múltiplas variáveis dentro de um loop e como é diferente de quando o fazemos fora de um loop. Eles também aprendem a obter, testar e contar as entradas corretas do jogador. | **feeding_PT** |
| 11 | **Adivinhando o número de gatos em um abrigo**<br>Os alunos aprendem o ciclo "repetir até" e como definir a condição para rastrear a condição que interrompe o jogo. Aprendem ainda a usar variáveis em diferentes situações: definir um valor aleatório, como um contador ou para obter dados dos jogadores. | **shelter_PT** |

## TURKISH Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 1 | **Snap! ara yüzünü tanıyalım**<br>Öğrenciler yeni bir karakter (sprite/kukla) ekler, karaktere bir kostüm ekler, kostümü düzenler ve bunlardan birini siler. Öğrenciler sahne için yeni bir arka plan oluşturur, düzenler ve istenmeyenleri siler. | **başla** |
| 2 | **Snap!'i keşfet - Karakteri hareket ettirme**<br>Öğrencilerin Snap! arayüzünü keşfetmelerine ve ilk kodlarını oluşturarak hareket eden ve konuşan bir karakter oluşturmalarına yardımcı olun. | **keşfet** |
| 3 | **Sahnede hareket etme**<br>Öğrenciler sahnede karakteri x ve y yönünde hareket ettirmeyi öğrenir, verilen görevleri çözmek için basit bir program programlar, karakteri farklı yöne çevirmeyi öğrenir. | **hareket** |
| 4 | **Kostüm değiştirme ve dönüşler**<br>Öğrenci, farklı rotasyon türleri arasında geçiş yaparak bir animasyon yapmak için karakerin kostümünü nasıl değiştireceğini öğrenir. | **dans** |
| 5 | **Çiftlikteki sesler**<br>Öğrenciler, oyuncunun belirli tuşlara basarak hayvanların seslerini tanıyabileceği basit bir oyun programlamayı öğrenirler. | **çiftlik** |
| 6 | **Bukalemenun yaz tatili**<br>Bir nesnenin kostümünü arka planın rengine göre değiştireceği basit bir oyun programlayın | **bukalemun** |
| 7 | **Prens ve Prensese evcil hayvanlarını bulmaları için yardım et**<br>Kız, seyahatleri sırasında hayvanların birbirleriyle karşılaşmasını engelleyerek Prenses'in kedisini bulmasına ve Prens'in köpeğini bulmasına yardım etmelidir. | **prenses** |
| 8 | **Tebeşir ile çizim yapmak**<br>Öğrencilere bir kodla farklı şekiller çizme tanıtılacaktır. Kodu kısaltmak ve arka planı değiştirmek için döngü tekrarını kullanmayı öğrenecekler. | **çizim** |
| 9 | **Çöpleri toplamak ve parkı temizlemek**<br>Öğrenciler değişkenleri nasıl kullanacaklarını ve bir kod bloğunu veya hatta bütün bir hareketli grafiği nasıl kopyalayacaklarını öğrenecekler. | **çöpler** |
| 10 | **Kedileri besleme**<br>Öğrencilere, bir döngü içinde çok değişkenli rastgele değer atama kavramı ve bir döngü dışında yaptığımızdan nasıl farklı olduğu anlatılacaktır. Ayrıca doğru oyuncu girdilerinin nasıl alınacağını, test edileceğini ve sayılacağını da öğrenecekler. | **kediler** |
| 11 | **Barınaktaki Kedi Sayısını Tahmin Etmek**<br>Öğrencilere "----e kadar tekrar et" döngüsü ve oyunu durduran koşulu örtük olarak izlemek için koşulun nasıl ayarlanacağı tanıtılacaktır. Ayrıca değişkeni; rastgele bir değer belirlemek, sayaç olarak veya oyuncuların girdisini almak gibi farklı durumlarda nasıl kullnacaklarını da öğrencekler. | **barınak** |

# ADVANCED LEARNING SCENARIOS

## ENGLISH Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 12 | **Catching healthy food**<br>Students will learn how to randomly move for X steps and choose a position and also how to use variables and conditionals for preventing other event. | **food** |
| 13 | **Storytelling**<br>Students will learn how to plan storytelling, how to use broadcast messages for synchronisation of the activities of sprites and stage changes. | **storytelling** |
| 14 | **Drawing**<br>Student will learn to draw in Snap!, to change the colour and the pen thickness, and how to use variables and conditionals that cause a new event. | **drawing** |
| 15 | **Catch the mouse**<br>Student will be introduced to the concept of multiple variable random value assignment. They will learn how to use the Operators/pick random[x]to[y] block. | **mouse** |
| 16 | **Buying food for a picnic**<br>Students will learn how to work with variables: setting different starting values, using conditionals to compare variables' value, changing variables' value, using variables for counting (un)healthy food. In addition, they will repeat adding text, joining texts and if statement. | **picnic** |
| 17 | **Operations**<br>Students will improve their knowledge on variables, random numbers, loops, broadcast. | **operation** |
| 18 | **Recycling**<br>Students will improve their knowledge and will extend the game scenario with new objects, code and changing code with respect to new sprites. They will be trained in code refactoring. | **recycling** |
| 19.1 | **Play a piano_1**<br>Students will learn about melodies coding and playing and will improve their knowledge about variables, loops, conditional, broadcast and other events. | **music** |
| 19.2 | **Play a piano_2**<br>Students will learn how to play music and change costume by clicking on a sprite. | **piano** |
| 20 | **Test**<br>Students will improve their knowledge and will extend the game scenario with new background, code and changing code with respect to new stages. | **test** |
| 21 | **Simplified PACMAN game**<br>Students will review their knowledge about movement inside a labyrinth with the use of sense color block. They will learn the concept of cloning the object with position restrictions and how to create a very simple non player character with its own random movement. | **pacman** |

## SLOVENIAN Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 12 | **Lovljenje zdrave hrane**<br>Učenec se nauči naključnega premikanja za X korakov in naključne izbire pozicije ter uporabo spremenljivk in pogojnih stavkov za preprečevanje drugih dogodkov. | **lovljenje_hrane_slo** |
| 13 | **Sestavi zgodbo**<br>Učenec se nauči načrtovanja pripovedovanja zgodb, uporabe pošiljanja obvestil za sinhronizacijo dejavnosti likov ter zamenjave ozadij. | **zgodba_slo** |
| 14 | **Onesnažen zrak**<br>Učenec se nauči risanja, spreminjanja barve, debeline svinčnika ter uporabe spremenljivke in pogojev za izvedbo drugega dogodka. | **zrak_slo** |
| 15 | **Ulovi miš**<br>Učenec se seznani s konceptom dodeljevanja naključnih vrednosti in se nauči, kako uporabiti operator naključno število od_[x]_do_[y]. | **mis_slo** |
| 16 | **Kupovanje hrane za piknik** | **piknik_slo** |
| 17 | **Računanje**<br>Učenec izboljša svoje znanje o spremenljivkah, naključnih številkah, zankah ter pošiljanju obvestil. | **racunanje_slo** |
| 18 | **Recikliranje** | **recikliranje_slo** |
| 19.1 | **Zaigraj na klavir 1**<br>Učenec se nauči sestavljanja melodij ter izboljša svoje znanje o spremenljivkah, zankah, pogojih, pošiljanju obvestil ter drugih dogodkov. | **ples_slo** |
| 19.2 | **Zaigraj na klavir 2**<br>Učenec se nauči predvajati glasbo in liku spremeniti obleko s klikom nanj. | **klavir_slo** |
| 20 | **Test**<br>Učenec izboljša svoje znanje in igro razširi z novimi ozadji, doda in spremeni kodo. | **test_slo** |
| 21 | **Enostavni Pacman**<br>Učenec ponovi s pomočjo zaznavanja barve liku omejiti gibanje, spozna koncept kloniranja z omejitvami gibanja in se nauči ustvarjanja lika, ki se samostojno naključno premika po labirintu. | **pacman_slo** |

## ITALIAN Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 12 | **Raccogliere i cibi salutari**<br>Gli studenti impareranno a muoversi casualmente per X passi, a scegliere una posizione, ad usare variabili e condizioni per prevenire altri eventi. | **cibo** |
| 13 | **Alice nel Paese delle Meraviglie**<br>Gli studenti impareranno a pianificare una storia, ad utilizzare i messaggi inviati e ricevuti per la sincronizzazione delle attività degli Sprite e per i cambiamenti di scena. | **alice** |
| 14 | **Disegno**<br>Lo studente imparerà a disegnare con Snap!, a cambiare il colore e lo spessore della penna e ad usare le variabili e le condizioni che determinano nuovi eventi. | **disegno** |
| 15 | **Alla caccia di un topolino!**<br>Gli studenti comprenderanno il concetto di assegnazione di più valori casuali variabili. Impareranno come usare gli Operatori / scegliere il blocco casuale da [x] a [y]. | **topolino** |
| 16 | **Acquistare cibo per un picnic**<br>Gli studenti impareranno a lavorare con le variabili: impostare diversi valori iniziali, usare i condizionali per confrontare il valore delle variabili, cambiare il valore delle variabili, usare le variabili per contare alimenti sani (e non). Inoltre, aggiungeranno il testo, useranno l'unione di testi e la condizione "if". | **picnic_1** |
| 17 | **Operazioni**<br>**Gli studenti miglioreranno le loro conoscenze su variabili, numeri casuali, cicli e broadcast.** | **operazioni** |
| 18 | **Raccolta differenziata**<br>Gli studenti miglioreranno le loro conoscenze per estendere lo scenario di gioco con nuovi oggetti, codice e cambiando codice rispetto ai nuovi Sprite. | **riciclo** |
| 19.1 | **Suonare il piano_1**<br>Gli studenti impareranno a codificare le melodie e miglioreranno le loro conoscenze su variabili, loop, condizionale, trasmissione e altri eventi. | **music_1** |
| 19.2 | **Suonare il piano_2**<br>Gli studenti impareranno a suonare e a cambiare costume facendo clic su uno Sprite | **piano_2** |
| 20 | **Test**<br>Gli studenti miglioreranno le loro conoscenze ed estenderanno lo scenario del gioco con nuovi background, codici e sue modifiche. | **test_1** |
| 21 | **PACMAN**<br>Gli studenti rivedranno le loro conoscenze sul movimento all'interno di un labirinto con l'uso dei blocchi dei sensori del colore. Impareranno il concetto di clonazione dell'oggetto con restrizioni di posizione e a creare un personaggio "non giocatore". | **pacman_1** |

## CROATIAN Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 12 | **Hvatanje zdrave hrane**<br>Učenici će naučiti kako nasumično pomicati za X koraka i odabrati položaj i također kako koristiti varijable i uvjete za sprječavanje drugih događaja. | **hvatanje** |
| 13 | **Pričam ti priču**<br>Učenici će naučiti kako ispričati priču, kako koristiti poruke i kako promijeniti pozadinu pozornice. | **prica** |
| 14 | **Crtanje**<br>Učenici će naučiti kako se crta pomoću Snap!-a, promijeniti boju i debljinu olovke te kako koristiti varijable i uvjete koji uzrokuju novi događaj. | **crtanje** |
| 15 | **Uhvati miša**<br>Učenik će se upoznati s konceptom dodjeljivanja više varijabli slučajnim vrijednostima. Naučit će kako koristiti blok Operatori / slučajni broj od [x] do [y]. | **miš** |
| 16 | **Kupnja hrane za piknik**<br>Učenici će naučiti kako raditi sa varijablama: postavljanje različitih početnih vrijednosti, korištenjem uvjeta za usporedbu vrijednosti varijabli, promjenom vrijednosti varijabli, korištenjem varijabli za brojanje (ne) zdrave hrane. Osim toga, ponovit će dodavanje i spajanje teksta te naredbu ako. | **piknik** |
| 17 | **Operacije**<br>Učenici će poboljšati svoje znanje o varijablama, slučajnim brojevima, petljama, emitiranju. | **operacije** |
| 18 | **Recikliranje**<br>Učenici će poboljšati prethodno stečeno znanje te će proširiti scenarij igre sa novim objektima, kodom i promjenom koda s obzirom na nove objekte. Učenici će moći restrukturirati kod. | **recikliranje** |
| 19.1 | **Sviranje klavira_1**<br>Studenti će upoznati kodiranje i sviranje melodija te će poboljšati svoje prethodno stečeno znanje o varijablama, petlji, uvjetnim, emitiranim i ostalim događajima. | **glazba** |
| 19.2 | **Sviranje klavira_2**<br>Učenici će naučiti svirati glazbu i mijenjati kostim klikom na sprite. | **klavir** |
| 20 | **Test**<br>Učenici će poboljšati svoje znanje i proširiti scenarij igre s novom pozadinom, kodom i promjenom koda u odnosu na nove pozornice. | **test_cro** |

## BULGARIAN Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 12 | **Подбор на здравословна храна**<br>Учениците ще се научат как да се придвижат на случаен принцип с X стъпки и да изберат позиция, както и как да използват променливи и условности за предотвратяване на друго събитие. | **food_bg** |
| 13 | **Разказвач**<br>Учениците ще научат как да планират разказването на истории, как да използват излъчвани съобщения за синхронизиране на дейностите на спрайтове и промени на сцената. | **story_bg** |
| 14 | **Рисуване**<br>Ученикът ще се научи да рисува в Snap !, да променя цвета и дебелината на писалката и да използва променливи и условия, които причиняват ново събитие. | **drawing_bg** |
| 15 | **Хванете мишката**<br>Учениците ще бъде запознати с концепцията за присвояване на случвайни стойности на променливи. Ще се научат как да използват блока Operators/pick random[x]to[y] block. | **mouse_bg** |
| 16 | **Купуване на храна за пикник**<br>Учениците ще се научат как да работят с променливи: задаване на различни начални стойности, използване на условия за сравняване на стойностите на променливите, промяна на стойностите на променливите, използване на променливи за броене на (не) здравословна храна. В допълнение, те ще разучат добавяне на текст, съединяване на текстове и if оператор. | **picnic_bg** |
| 17 | **Операции.** Учениците ще подобрят знанията си за променливи, случайни числа, цикли, Broadcast. | **oper_bg** |
| 18 | **Рециклиране**<br>Учениците ще подобрят знанията си и ще разширят сценария на играта с нови обекти, код и променлив код по отношение на новите спрайтове. Те ще бъдат обучени как да редактират кодове. | **recycling_bg** |
| 19.1 | **Посвири на пиано Версия 1**<br>Учениците ще се запознаят с кодирането и свирането на мелодии и ще подобрят знанията си за променливи, цикли, условия, излъчване и други събития. | **music_bg** |
| 19.2 | **Посвири на пиано Версия 2.** Учениците ще се научат как да свирят музика и да сменят костюма, като кликнат върху спрайт. | **piano_bg** |
| 20 | **Тест.** Учениците ще подобрят знанията си и ще разширят сценария на играта с нов фон, код и променлив код по отношение на новите сцени. | **test_bg** |
| 21 | **Опростена игра PACMAN**<br>Учениците ще приложат своите знания за движение в лабиринт с помощта на сензорен цветен блок. Те ще научат концепцията за клониране на обект с ограничения за позицията и как да създадат много прост персонаж, който не е играч, с опция за произволно движение. | **pacman_bg** |

## GREEK Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 12 | **Πιάνοντας υγιεινά τρόφιμα**<br>Οι μαθητές θα μάθουν πώς να κινούνται για X τυχαία βήματα και να επιλέγουν μια θέση και επίσης πώς να χρησιμοποιούν μεταβλητές και συνθήκες για την πρόληψη άλλου συμβάντος. | **food_gr** |
| 13 | **Διήγηση ιστορίας**<br>Οι μαθητές θα μάθουν πώς να σχεδιάζουν την αφήγηση μιας ιστορίας, πώς να χρησιμοποιούν τα μηνύματα μετάδοσης για συγχρονισμό των δραστηριοτήτων των στοιχείων με τις αλλαγές στη σκηνή. | **story_bg** |
| 14 | **Ζωγραφική**<br>Ο μαθητής θα μάθει να ζωγραφίζει με το Snap!, να αλλάζει το χρώμα και το πάχος του στυλό και πώς να χρησιμοποιεί μεταβλητές και συνθήκες για ένα νέο συμβάν. | **drawing_gr** |
| 15 | **Πιάσε το ποντίκι**<br>Ο μαθητής θα μάθει την έννοια της πολλαπλής ανάθεσης τυχαίας τιμής. Θα μάθει πώς να χρησιμοποιεί τους τελεστές / επιλέγει τυχαία [x] έως [y]. | **mouse_gr** |
| 16 | **Αγοράζοντας φαγητό για πικνίκ**<br>Οι μαθητές θα μάθουν πώς να δουλεύουν με μεταβλητές: ανάθεση διαφορετικής αρχικής τιμής, χρήση συνθήκης για σύγκριση της τιμής των μεταβλητών, αλλαγή της τιμής των μεταβλητών, χρήση μεταβλητών για την καταμέτρηση (μη) υγιεινών τροφίμων. Επιπλέον, θα επαναλάβουν την προσθήκη κειμένου, την ένωση κειμένων και τη δήλωση εάν. | **picnic_gr** |
| 17 | **Λειτουργίες**<br>Οι μαθητές θα βελτιώσουν τις γνώσεις τους σχετικά με μεταβλητές, τυχαίους αριθμούς, βρόχους και μετάδοση. | **oper_gr** |
| 18 | **Ανακύκλωση**<br>Οι μαθητές θα βελτιώσουν τις γνώσεις τους και θα επεκτείνουν το σενάριο του παιχνιδιού με νέα αντικείμενα, κώδικα και ανακατασκευή κώδικα σε σχέση με τα νέα στοιχεία. Θα εκπαιδευτούν στην ανακατασκευή κώδικα. | **recycling_gr** |
| 19.1 | **Παίξτε πιάνο_1**<br>Οι μαθητές θα μάθουν να γράφουν κώδικα και να παίζουν μελωδίες και θα βελτιώσουν τις γνώσεις τους σχετικά με μεταβλητές, βρόχους, συνθήκες, μετάδοση γεγονότων και άλλα γεγονότα. | **music_gr** |
| 19.2 | **Παίξτε πιάνο_2**<br>Οι μαθητές θα μάθουν πώς να παίζουν μουσική και να αλλάζουν κοστούμι κάνοντας κλικ σε ένα στοιχείο. | **piano_gr** |
| 20 | **Δοκιμαστικό μάθημα - Αλλαγή εμφάνισης και στροφή**<br>Μάθετε πώς να αλλάζετε την εμφάνιση του στοιχείου για να κάνετε ένα κινούμενο σχέδιο, αλλάζοντας μεταξύ διαφορετικών τύπων περιστροφών. | **test_gr** |
| 21 | **Απλοποιημένο παιχνίδι PACMAN**<br>Οι μαθητές θα επανεξετάσουν τις γνώσεις τους σχετικά με την κίνηση μέσα σε ένα λαβύρινθο. Θα μάθουν την έννοια της κλωνοποίησης του αντικειμένου με περιορισμό θέσης και πώς να δημιουργήσουν έναν πολύ απλό χαρακτήρα με τη δική του τυχαία κίνηση. | **pacman_gr** |

## PORTUGUESE Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 12 | **Comendo comida saudável**<br>Os alunos aprendem a mover-se aleatoriamente X passos e a escolher uma posição. Aprendem a usar variáveis e instruções condicionais para prevenir um evento. | **food_PT** |
| 13 | **Narrativa**<br>Os alunos aprendem a planejar a narração de histórias e como usar mensagens para sincronização das atividades dos sprites e mudanças de palco. | **storytelling_PT** |
| 14 | **Desenhando**<br>Os alunos aprendem a desenhar no Snap!, a alterar a cor e a espessura da caneta e a usar variáveis e condicionais que geram um novo evento. | **drawing_PT** |
| 15 | **Apanhar o rato**<br>Os alunos aprendem o conceito de atribuição de valores aleatórios de múltiplas variáveis. Aprendem ainda como usar o bloco de operadores de escolha aleatória [x] a [y]. | **mouse_PT** |
| 16 | **Comprando comida para um piquenique**<br>Os alunos aprendem a trabalhar com variáveis: valores iniciais, operadores condicionais para comparar o valor das variáveis, alterar o valor das variáveis, usar variáveis para contar alimentos saudáveis. Além disso, eles vão adicionar e juntar textos. | **picnic_PT** |
| 17 | **Operações**<br>Os alunos vão melhorar os seus conhecimentos sobre variáveis, números aleatórios e ciclos. | **operation_PT** |
| 18 | **Reciclagem**<br>Os alunos vão aumentar um cenário do jogo com novos objetos, código e código em relação a novos sprites. Aprendem ainda sobre refatoração de código. | **recycling_PT** |
| 19.1 | **Tocar piano_1**<br>Os alunos aprendem sobre codificação e reprodução de melodias e melhoram os seus conhecimentos sobre variáveis, ciclos, instruções condicionais e outros eventos. | **music_PT** |
| 19.2 | **Tocar piano_2**<br>Os alunos aprendem a tocar música e a mudar de traje clicando num sprite. | **piano_PT** |
| 20 | **Teste**<br>Os alunos irão melhorar os seus conhecimentos alargando o cenário do jogo com um novo fundo, código e código mutável em relação a novos palcos. | **test_PT** |
| 21 | **Jogo PACMAN simplificado**<br>Os alunos irão rever os seus conhecimentos sobre o movimento dentro de um labirinto com o uso do bloco de reconhecimento de cores. Aprendem o conceito de clonar o objeto com restrições de posição e como criar um personagem não-jogador muito simples com o seu próprio movimento aleatório. | **pacman_PT** |

## TURKISH Scenarios

| N. | TITLE | CODE |
|---|---|---|
| 12 | **Sağlıklı yiyeceği yakalamak**<br>Öğrenciler, belirli bir adım atmak için rastgele hareket etmeyi, bir konum seçmeyi ve ayrıca diğer olayları önlemek için değişkenleri ve koşulluları nasıl kullanacaklarını öğreneceklerdir. | **yemek** |
| 13 | **Hikaye Anlatma**<br>Öğrenciler hikaye anlatımını nasıl planlayacaklarını, karakterin aktiviteleri ile sahne değişikliklerinin senkronizasyonu için yayın mesajlarının nasıl kullanılacağını öğrenecekler | **hikaye** |
| 14 | **İklimi İyileştirmek-Çizim Yapma**<br>Öğrenci, Snap!'te çizim yapmayı, rengi ve kalem kalınlığını değiştirmeyi ve yeni bir olaya neden olan değişkenler ile koşullu ifadeleri nasıl kullanacağını öğrenecek. | **iklim** |
| 15 | **Fareyi yakalamak**<br>Öğrencilere çok değişkenli rastgele değer atama kavramı tanıtılacaktır. Operatörleri nasıl kullanacaklarını ve rastgele [x] ila [y] bloğu seçmeyi öğrenecekler. | **fare** |
| 16 | **Piknik için yemek almak**<br>Öğrenciler değişkenlerle nasıl çalışılacağını öğrenecekler: farklı başlangıç değerleri belirleme, değişkenlerin değerini karşılaştırmak için koşullu ifadeler kullanma, değişkenlerin değerini değiştirme, sağlıklı yiyecekleri saymak (olmayan) için değişkenler kullanma. Ek olarak, metin eklemeyi, metinleri birleştirmeyi ve Eğer (if) ifadesini tekrarlayacaklar. | **piknik** |
| 17 | **İşlemler.** Öğrenciler değişkenler, rastgele sayılar, döngüler, yayın hakkındaki bilgilerini geliştireceklerdir. | **işlem** |
| 18 | **Geri Dönüşüm.** Öğrenciler bilgilerini geliştirecek ve oyun senaryosunu yeni karakterlere göre yeni nesneler, kodlar ve değişen kodlarla genişletecekler. Kod yeniden düzenleme konusunda eğitilecekler | **dönüşüm** |
| 19.1 | **Piyano Çalmak**<br>Öğrenciler melodileri kodlama ve çalma hakkında bilgi edinecek ve değişkenler, döngüler, koşullu, yayın ve diğer olaylar hakkındaki bilgilerini geliştireceklerdir. | **Piyano1** |
| 19.2 | **Piyano Çalmak**<br>Öğrenciler bir karaktere tıklayarak müzik çalmayı ve kostümü değiştirmeyi öğrenecekler. | **piyano2** |
| 20 | **Test- Kostümleri değiştirmek ve Dönüşler**<br>Farklı döndürme türleri arasında geçiş yaparak bir animasyon yapmak için karakterin kostümünü nasıl değiştireceğinizi öğrenin | **test** |
| 21 | **Basitleştirilmiş PACMAN oyunu**<br>Öğrenciler duyu renk bloğu kullanarak bir labirent içindeki hareketin nasıl kodlanması gerektiği hakkındaki bilgilerini gözden geçireceklerdir. Nesneyi konum kısıtlamaları ile rastgele hareket edecek şekilde basitçe kodlayacakar ve karakterleri klonlamayı öğrenecekler. | **pacman** |

# ЛИТЕРАТУРА

A project of the K-12 Initiative at Stanford University. (2009, 05 06). *Taking Design Thinking to School: Approaches to Integrating the Design Process in K-12 Teaching and Learning.* Retrieved 09 10, 2020, from Stanford Graduate School of Education: https://web.stanford.edu/dept/SUSE/taking-design/presentations/Taking-design-to-school.pdf

Alserri, S., Zin, N., & Wook, T. (2018). Alserri, S. A., Zin, N. A. M., & Wook, T. S. M. T.Gender-based Engagement Model for Serious Games. , , 8(4). . *International Journal on Advanced Science, Engineering and Information Technology, 8*(4), 1350-1357.

Baptista, R., & Vaz de Carvalho, C. (2010). Role Play Gaming and Learning. *Learning Technology, 12*(1).

Combéfis, S., Beresnevičius, G., & Dagiene, V. (2016). Learning Programming through Games and Contests: Overview, Characterisation and Discussion. *Olympiads in Informatics, 10*.

Cooper, S., Dann, W., & Pausch, R. (2000). Alice: A 3-D tool for introductory programming concepts. *Journal of Computing Sciences in Colleges, 15*(5), 107-116.

Doorley, S., Holcomb, S., Klebahn, P., Segovia, K., & Utley, J. (2018). *Design Thinking Bootleg.* Retrieved from https://static1.squarespace.com/static/57c6b79629687fde090a0fdd/t/5b19b2f2aa4a99e99b26b6bb/1528410876119/dschool_bootleg_deck_2018_final_sm+%282%29.pdf

Eurostat. (2020, 09 25). *ICT specialists in employment.* Retrieved 11 12, 2020, from eurostat Statistics Explined: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=ICT_specialists_in_employment#Number_of_ICT_specialists

Feldgen, M., & Clua, O. (2004). Games as a motivation for freshman students to learn programming. *34th ASEE/IEEE Frontiers in Education Conference*, (pp. 1079–1084).

Frankovic, I., Hoic-Bozic , N., & Nacinovic-Prskalo, L. (2018). Serious Games for Learning Programming Concepts. *8th Conference edition The Future of Education.* Florence, Italy.

Garris , R., Ahlers, R., & Driskell, J. E. (2002). Games, motivation and learning. Simulation & Gaming. *An Interdisciplinary Journal of Theory, Practice and Research, 33*(4), 43-56.

Gee, J. P. (2003). *What Video Games Have to Teach Us About Learning and Literacy.* New York: Palgrave/Macmillan.

Gee, J. P. (2007). Are video games good for learning? *Curriculum & Leadership Journal, 5*(1).

Geist, E. (2016). Robots, Programming and Coding, Oh My! *Childhood Education, 92*(4), 298-304. doi:10.1080/00094056.2016.1208008

Georgieva, R., & Tuparova, D. (2019). Educatiomnal Computer Game-Based Environments for Teaching Programming and Developmnt of an Algorithmic Thinking - Comparative AnalysisITHMIC THINKING – COMPARATIVE ANALYSIS. *Mathematics and Education in Mathematics, Forty-eighth Spring Conferenceof the Union of Bulgarian Mathematicians* (pp. 150-156). Union of Bulgarian Mathematicians. Retrieved from http://www.math.bas.bg/smb/2019_PK/tom/pdf/150-156.pdf

Georgieva, R., & Tuparova, D. (2020). Design Thinking - helps in school eucation. *Anniversary International Scientific Conference "Synergeticsand ReflectioninMathematics Education"*, (pp. 341-347). Pamporovo.

Grivokostopoulou, F., Perikos, I., & Hatzilygeroudis, I. (2016). An Educational Game for Teaching Search Algorithms. *Proceedings of the 8th International Conference on Computer Supported Education (CSEDU 2016)* (pp. 129–136). Setubal. Portugal: SCITEPRESS - Science and Technology Publications, Lda.

Gross, B. (2003). The impact of videogames in education. 8(7). *First Monday, 8*(7).

Hijon-Neira, R., Velazquez-Iturbide, A., Pizarro-Romero, C., & Carrico, C. (2015). Serious games for motivating into programming. *Frontiers in Education Confere.*

IDEO Design thinking. (2008, 09 7). *Definitions of design thinking*. Retrieved 11 20, 2020, from https://designthinking.ideo.com/blog/definitions-of-design-thinking

Innovation Starter. (2015, 06 04). *What is design thinking?* Retrieved 11 20, 2020, from http://innovationstarterbox.bg/resources/kakvo-e-dizain-mislene/

Jemmali , C. (2016). May's Journey: A serious game to teach middle and high school girls programming. Worcester Polytechnic Institute. Retrieved from https://digitalcommons.wpi.edu/etd-theses/455

Johnson, C., & all, a. (2016). Game Development for Computer Science Education. *the 2016 ITiCSE Working Group Reports.*

Johnston, R. T., & de Felix, W. (1993). Learning from video games. *Computer in the Schools* , 199-233.

Kafai, Y. (1995). Making game artifacts to facilitate rich and meaningful learning. *Annual Meeting of the American Educational Research Association*, (pp. 1–20). San Francisco.

Karakehayova, M. (2019). Opportunities for Using Design Thinking in School Education. *Education and Development, 3*.

Kazimoglu, C., Kiernan, M., Bacon, L., & Mackinnon, L. (2012). A Serious Game for Developing Computational Thinking and Learning Introductory Computer Programming. *Procedia - Social and Behavioral Sciences, 47*, 1991-1999.

Kelleher , C., Pausch, R., & Kiesler, S. (2007). Storytelling Alice motivates middle school girls to learn computer programming. *Proc. CHI 2007.*, (pp. 1455-1464).

Luka, I. (2014). Design thinking in pedagogy. 2014, 5,. *The Journal of Education, Culture, and Society, 5*, 63-74.

Malliarakis, C., Satratzemi, M., & Xinogalos, S. (2014). CMX: Implementing an MMORPG for learning programming, 8th European Conference on Games Based Learning. *8th European Conference on Games Based Learning - ECGBL 2014.* Berlin.

Malone, T. (1981). What makes computer games fun? *Byte, 6*(12), 258-277.

Malone, T. W. (1981b). Toward a theory of intrinsically motivating instruction. *Cognitive Science, 4*.

Mathrani, A., Christian, S., & Ponder-Sutton, A. (2016). PlayIT: Game Based Learning Approach for Teaching Programming Concepts. *Educational Technology & Society, 19*(2), 5-17.

Meerbaum-Salant , O., Armoni, M., & Ben-Ari, M. (2013). Learning computer science concepts with Scratch. *Computer Science Education, 23*(3), 239-264. doi:10.1080/08993408.2013.832022

Michael, D. R., & Chen, S. (2006). *Serious Games: Games That Educate, Train, and Inform.* Boston: Course Technology PTR.

Miljanovic, M., & Bradbury, J. (2016). Robot on!: a serious game for improving programming comprehension. *Proceedings of the 5th International Workshop on Games and Software Engineering.* Austin, Texas, USA.

Miljanovic, M., & Bradbury, J. (2018). A Review of Serious Games for Programming. *4th Joint International Conference on Serious Games.* Darmstadt, Germany.

Naiman, L. (2019, 06 10). *Design Thinking as a Strategy for Innovation*. Retrieved 11 21, 2020, from Creativity at work: https://www.creativityatwork.com/design-thinking-strategy-for-innovation/

Nančovska Šerbec, I., Kaučič, B., & Rugelj, J. (2008). Pair programming as a modern method of teaching computer science. *International journal on emerging technologies in learning, 3*, 45-49.

Ouahbi, I., Kaddari, F., Darhmaoui, H., Elachqar, A., & Lahmine, S. (2015). Learning Basic Programming Concepts by Creating Games with Scratch Programming Environment. , 191, . *Procedia - Social and Behavioral Sciences, 191*, 1479-1482. doi:doi:10.1016/j.sbspro.2015.04.224

Paavola, S., & Hakkarainen, K. (2005). The Knowledge Creation Metaphor – An Emergent Epistemological Approach to Learning.14. *Sci Educ, 14*, 535-546.

Phan, M., Jardina, J., Hoyle, S., & Chaparro, B. (2012). Examining the role of gender in video game usage, preference, and behavior. *Human Factors and Ergonomics Society 51*, (pp. 1496–1500.).

Pivec, M., & Kearney, P. (2007). Games for Learning and Learning from Games. *Informatica, 31*, 419-423.

Pivec, M., Koubek, A., & Dondi, C. (2004). *Guidelines for Game-Based Learning. Lengerich.* Pabst Science Publishers.

Prensky , M. (2001). *Digital Game-Based Learning.* New York: Mc-Graw-Hill.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond , E., Brennan , K., . . . Kafai, Y. (2009). Scratch: Programming for All. *Communication of the ACM, 52*, 60-67.

Rieber , L. P., Smith , L., & Noah, D. (1998). The value of serious play. *Educational Technology, 38*(6), 29-37.

Rugelj, J. (2016). Serious computer games design for active learning in teacher education. (C. Carvalho , P. Escudeiro, & A. Coelho, Eds.) *Serious games, interaction, and simulation : revised selected papers. Lecture notes of the institute for computer sciences, social informatics and telecommunications engineering, 161*, 94-102.

Rugelj, J., & Lapina, M. (2019). Game design based learning of programming. In J. Rugelj, & M. Lapina (Ed.), *Proceedings of the International Scientific Conference Innovative Approaches to the Application of Digital Technologies in Education and Research (SLET-2019), May 20-23, CEUR workshop proceedings. 2494.* Stavropol-Dombay, Russia: Aachen: CEUR-WS.

Shabanah, S., Chen, J., Wechsler, H., Carr, D., & Wegman, E. (2010). Designing Computer Games to Teach Algorithms. *Seventh International Conference on Information Technology: New Generations, ITNG 2010*, (pp. 1119-1126). Las Vegas, NV.

Shute, V. J., & Ke, F. (2012). Games, Learning, and Assessment. In D. Ifenthaler, D. Eseryel, & X. Ge, *Assessment in Game-Based Learning: Foundations, Innovations, and Perspective.* New York, USA: Springer.

Shute, V. J., Rieber, L. P., & Van Eck, R. (2012). Games ... and ... Learning. In R. A. Reiser, & J. V. Dempsey, *Trends and issues in instructional design and technology (3rd ed.).* (pp. 321-332). Boston: Pearson Education.

Sykes, E. (2007). Determining the Effectiveness of the 3D Alice Programming Environment at the Computer Science I Level. *Journal of Educational Computing Research, 36*(2), 223-244. doi:10.2190/J175-Q735-1345-270M

Tuparova, D., Nikolova, E., & Tuparova, E. (2020). Integrated game-based learning in an informatics secondary course: Is there a difference between girls' and boys' achievements? *CSEDU 2020 - Proceedings of the 12th International Conference on Computer Supported Education*, *1*, pp. 702-709. Retrieved 10 12, 2020, from https://www.scitepress.org/Link.aspx?doi=10.5220/0009818407020709

Tuparova, D., Tuparov, G., & Veleva, V. (2019 b). Girls' and boys' viewpoint on educational computer games. *European Conference on Games-based Learning*, (pp. 757–766).

Vahldick, A. (2017). Aperfeiçoamento das Competências de Resolução de Problemas na Aprendizagem Intodutória de Programação de Computadores usando um jogo sério digital. Faculdade de Ciências e Tecnologia da Universidade de Coimbra. Retrieved from http://hdl.handle.net/10316/79528

van Eck, R. (2006). Digital Game-Based Learning: It's Not Just the Digital Natives Who Are Restless. *EDUCAUSE Review, 41*(2).

Vermeulen, L., Van Looy, J., Courtois, C., & De Grove, F. (2011). Girls will be girls: a study into differences in game design preferences across gender and player types. *Under the mask: perspectives on the gamer conference*, (pp. 1-21).

Weintrop, D., & Wilensky, U. (2015). To Block or not to Block, That is the Question: Students' Perceptions of Blocks-based Programming. *Interaction Design and Children*, (pp. 199-208).

Whitton, N. (2009). *Learning with Digital Games: A Practical Guide to Engaging Students in Higher Education.* New York: Routledge.

Whitton, N., & Moseley, A. (2012). *Using Games to Enhance Learning and Teaching: A Beginner's Guide.* New York: Routledge.

Wolz, U., Barnes, T., & Bayliss, J. (2009). Girls do like playing and creating games. *ACM SIGCSE Bulletin, 41*(1), 199–200.

Wolz, U., Maloney, J., & Pulimood, S. (2008). 'Scratch' Your Way to Introductory CS. *SIGCSE Bulletin, 40*, 298-299.

Zapušek, M., & Rugelj, J. (2013). Learning programming with serious games. *EAI Endorsed Trans. on Game-Based Learning, 13*, 1-12.