**O3 – Инструкция за учителя: Съдържание**

**Сборник от работни листове за учителя, базирани на дейности за проектиране на игри**

**Информация за документа**

**Резултат по проекта:** O3 / A1 – Сборник от работни листове за учителя, базирани на дейности за проектиране на игри

**Интелектуален резултат № - O3:** Заглавие – Инструкция за учителя

**Ръководител:** Югозападен университет „Неофит Рилски” (България)

**Участвали партньори:** Университет в Любляна (Словения), Университет в Риека (Хърватия)

Автори:

**Отговорност**

Този проект е финансиран от програма „Еразъм +“ на Европейския съюз.

Информацията и вижданията, изложени в тази публикация, са на автора (авторите) 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.

Coding4Girls, 2018-2020

https://licensebuttons.net/l/by-sa/3.0/88x31.png

## Creative Commons Attribution-ShareAlike 4.0

## International Public License ([CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0/))

**Съдържание**

[Основни учебни сценарии 8](#_Toc61442180)

[Учебен сценарий 1 - Въведение в Snap! интерфейс 8](#_Toc61442181)

[Учебен сценарий 2 - Време е да оживите вашия спрайт 13](#_Toc61442182)

[Учебен сценарий 3 - Придвижване по сцената 17](#_Toc61442183)

[Учебен сценарий 4 - Смяна на костюми и обръщане 23](#_Toc61442184)

[Учебен сценарий 5 - Звуци от фермата 29](#_Toc61442185)

[Учебен сценарий 6 – Лятната ваканция на хамелеона 35](#_Toc61442186)

[Учебен сценарий 7 – Помогнете на принца и принцесата да открият техните животни 45](#_Toc61442187)

[Учебен сценарий 8 – Рисуване с креда (тебишир) 52](#_Toc61442188)

[Учебен сценарий 9 - Прибиране на боклука и почистване на парка 63](#_Toc61442189)

[Учебен сценарий 10 - Хранене на котките 71](#_Toc61442190)

[Учебен сценарий 11 - Познай броя на котките в приют 78](#_Toc61442191)

[Сценарии за напреднали 86](#_Toc61442192)

[Учебен сценарий 12 – Улов на здравословна храна 86](#_Toc61442193)

[Учебен сценарий 13 – Разказване на история 95](#_Toc61442194)

[Учебен сценарий 14 – Рисуване 105](#_Toc61442195)

[Учебен сценарий 15 – Хвани мишката 113](#_Toc61442196)

[Учебен сценарий 16 – Купуване на храна за пикник 122](#_Toc61442197)

[Учебен сценарий 17 – Операции 130](#_Toc61442198)

[Учебен сценарий 18 – Рециклиране 136](#_Toc61442199)

[Учебен сценарий 19.2 – Свири на пиано 147](#_Toc61442200)

[Учебен сценарий 20 - Тест 156](#_Toc61442201)

[Учебен сценарий 21 - Опростена игра PACMAN 161](#_Toc61442202)

[Литература 168](#_Toc61442203)

Увод

Водещи психолози от миналия век определят играта като една от най-важните дейности за развитието на важни житейски умения, независимо от възрастта или етапа на развитие. Детето чрез играта бързо се приспособява към нови обстоятелства и се справя с промяната с лекота. Когато играе, той открива основни понятия от истинската дума и се правят първите основни връзки между тях.

В днешно време игрите се използват по-често в най-ранните етапи от развитието на детето у дома и в детската градина. Обучението в училище все още твърде често се основава на традиционно предаване на знания в рамките на ориентиран към учителя модел с пасивни ученици. От друга страна, теориите за обучение, разработени през миналия век, насърчават нови подходи към преподаването и ученето, които са ориентирани към учениците, базирани на проблеми, насочени към по-високо познавателно равнище на образователните цели (на по-високи таксономични нива), мотивационни и често подкрепяни от ИКТ.

Подходът CODING4GIRLS ще насърчи участието в дейности по програмиране чрез „подход с висок таван с нисък вход“, който в началото има ниски изисквания за знания, като същевременно не ограничава предизвикателствата за решаване на проблеми за по-напредналите учещи. Учащите ще бъдат насърчавани да довършат частично завършени решения чрез добавяне на липсващи градивни елементи на кода или да създадат свои собствени решения. Дейностите се планират последователно, от основни със само едно понятие за програмиране до по-напреднали с множество понятия и процедури за програмиране. Докато подготвяхме учебни дейности в Snap !, ние се фокусирахме върху идентифицираните характеристики на игрите, предпочитани от момичетата, и върху дейностите, свързани с реалните проблеми.

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.

Изготвени са 22 учебни листа, съответстващи на учебните дейности. Учителите могат да използват сценариите и игрите в предложената последователност или могат да ги избират свободно според своите предпочитания и нужди. 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.

Учебните листове са на разположение на английски, както и на националните езици на партньорите по проекта - български, хърватски, гръцки, италиански, португалски, словенски и турски.

**Учебни листове**

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.

|  |  |  |
| --- | --- | --- |
| **Основни обучителни сценарии** | | |
| 1 | **Въведение в Snap! интерфейс**  Запознаване със Snap! среда за визуално блоковопрограмиране | UL |
| 2 | **Време е да оживите вашия спрайт**  Намерете блокове за програмиране, свържете ги, преместете спрайта, накарайте спрайта да каже нещо. | UL |
| 3 | **Придвижване по сцената**  Създаване на смислена последователност от блокове за движение на героите (спрайтовете). | UL |
| 4 | **Смяна на костюми и обръщане** | UL |
| 5 | **Звуци от фермата**  Добавяне, импортиране, запис и възпроизвеждане на звук. | UL |
| 6 | **Лятната ваканция на Хамелеона, опростена версия**  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**  Използване на условни, рисуване | UL |
| 8 | **Рисуване с креда (тебешир)**  Използване на цикли/loop/, обръщане, промяна на фона (сцената) | UL |
| 9 | **Прибиране на боклука и почистване на парка**  Запознаване с променливи, дублиране на спрайтове, части/blocks/ от код | UL |
| 10 | **Хранене на котките**  Използване на променливи (вътре / извън цикъла), цикли, произволни числа, конкатенация на низове, оператори, вход. | UL |
| 11 | **Познай броя на котките в приют**  Използване на случайни числа, въвеждане на променливи, условни блокове, оператори за сравнение, брояч. | UL |
| **ADVANCED LEARNING SCENARIOS** | | |
| 12 | **Улов на здравословна храна**  Използване на променливи, условни, цикъл, точка в посока, произволни | UL |
| 13 | **Разказване на приказки** | SWU |
| 14 | **Рисуване** | UNIRI |
| 15 | **Хвани мишката**  Използване на цикли, условни условия, променливи | UL |
| 16 | **Купуване на храна за пикник**  Използване на променливи, условни условия, оператори | UL |
| 17 | **Операции**  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**  Използване на движение на обект въз основа на събития, сензори на цветовете, логически стойности, проверка и реагиране на две различни състояния на играта | UL |

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:   * Запознаване със Snap! среда за визуално блоково програмиране   Конкретни резултати от обучението:   * ученикът може да добави нов спрайт * ученикът може да добави костюм към спрайт и да го редактира * ученикът е в състояние да центрира спрайта, така че въртенето да работи правилно * 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.  **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  Самостоятелна работа |
| **Форми на обучение** | Работа с класа/групата (фронтални дейности)  Индивидуална работа |
| **Резюме на обучението** | (Мотивация-въведение, прилагане, осмисляне и оценка)  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.  [Стъпка 1]  Покажете на учениците уеб страницата, където могат да намерят Snap! (<https://snap.berkeley.edu/>).  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.    [Стъпка 2]  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.  Задача за учениците: 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:    Задача за учениците: нарисувайте своя фон.  Задача за учениците: претърсете съществуващите фонове и добавете един от тях, така че да имате два.  Задача за учениците: Намерете начин да редактирате фона си. Намерете начин да изтриете един от вашите фонове, така че да остане само един.  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:   * Ученикът знае къде да намери определени блокове за програмиране и как да ги свърже в последователност * Ученикът знае как да премества спрайт * Ученикът знае как да накара спрайта да каже нещо * Специфични резултати от обучението, ориентирани към алгоритмично мислене: * 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  Самостоятелна работа |
| **Форми на обучение** | Работа пред класа/групата  Самостоятелна работа |
| **Резюме на обучението** | (Мотивация-въведение, прилагане, осмисляне и оценка)  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.  [Стъпка 1]  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 блокoвете 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: C:\Users\cerarsp\Downloads\C4G_dog_goes_home script pic(2).png  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.  Задача за ученици: 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?  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.  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.  Рефлексия и оценка:  Колко пъти вашият герой трябваше да повтори кодът за движение и да изпълни последователсността „кажи“, за да изпълни задачата? Еднакво ли е числото за всички в класа? Защо е така? |
| **Инструменти и ресурси за учителя** | **Примерна програма:**  <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) * Ако ученикът не е нарисувал собствен спрайт и фон, той може да използва: * <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:   * Създаване на смислена последователност от блокове   Специфични резултати от обучението, ориентирани към алгоритмично мислене:   * Ученикът позиционира спрайта на сцената * Ученикът променя x и y позицията на спрайта * Ученикът използва цикъл repeat n * 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 х и у 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.  **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.  **Цел: Ученикът прави разлика между движението в позиции х и у на сцената и използва цикъл.** |
| **Продължителност** | 45 минути |
| **Стратегия и методи на обучение** | Учителят демонстрира дейностите пред класа/групата  Самостоятелна работа |
| **Форми на обучение** | Фронатална работа/ групова работа  Самостоятелна работа |
| **Резюме на обучението** | (Мотивация-въведение, прилагане, рефлексия и оценка)  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 (lp 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.  Код за кучето:    Код за маймуната:    Задача: Опитайте се да накарате кучето да тича към топката и обратно.  Задача: 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: |
| **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> * 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> * 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> * 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:   * Създаване на смислена последователност от блокове   Специфични резултати от обучението, ориентирани към алгоритмично мислене:   * Ученикът променя костюма на спрайт, за да направи анимация * 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.  **Задача:** Създайте програма, която променя костюма на спрайта; във всяка програма задайте подходящ тип завъртане за всеки спрайт  **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  Самостоятелна работа |
| **Форми на обучение** | Презентация пред класа/групата  Самостоятелна работа |
| **Резюме на обучението** | (Мотивация-Въведение, Приложение, Рефлексия и Оценка)  Ще научите как да направите анимация на спрайт, така че да изглежда като ходене, танци, ...  [Стъпка 1]  Отворете нов празен проект, щракнете върху иконата, която прилича на бял лист хартия, и изберете **Костюми ...**  Щракнете върху Балерина a (Ballerina) и щракнете върху **Импортиране** (import). Направете същото с Балерина b, Балерина c и Балерина d.  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**:    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 . 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 й 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:    Кръглата стрелка означава, че вашият спрайт може да се обръща във всяка посока (като мишката).  Стрелката <-> означава, че спрайтът ви ще се обръща само наляво или надясно (това е, което бихте използвали кучето да не ходи „на главата си“)  Последната -> стрелка означава, че спрайтът винаги ще изглежда такъв, какъвто е (можете да използвате това за маймуната)  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> * Avery ходене: <https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_Avery_walking> * 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> * 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. * Ученикът може да добави нов спрайт. * Ученикът знае как да накара спрайта да каже нещо. |
| **Очаквани резултати** | Общи резултати от обучението:   * добавяне звук от медийната библиотека на Snap, * импортиране на звук от други носители, * запис на нов звук, * възпроизвеждане на звук при натискане на клавиш.   Специфични резултати от обучението, ориентирани към алгоритмично мислене:   * ученикът добавя звук от медийната библиотека на Snap и го възпроизвежда при натискане на определен клавиш, * ученикът импортира звук от компютър и го възпроизвежда при натискане на определен клавиш, * 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.  **Задачи:** В първата стъпка ученикът трябва да избере фона на сцената. След това трябва да програмира жената фермер да казва инструкциите:  1) Ако искате да чуете кучето, щракнете върху бутона "D" !;  2) Ако искате да чуете кравата, щракнете върху бутона "C" !;  3) Ако искате да чуете овцете, щракнете върху бутона "S" !;  4) Ако искате да чуете прасето, щракнете върху бутона "P" !;  5) Ако искате да чуете коня, щракнете върху бутона "H" !.  След това ученикът трябва да програмира задачата според указанията на жената фермер.  **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  Самостоятелна работа |
| **Резюме на обучението** | (Мотивация-въведение, прилагане, осмисляне и оценка)  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-11 at 12.26.19.png**Мотивация-Въведение**  Мотивираме учениците, като играем играта (те не виждат кода). Целта на урока е да се създаде подобна игра.  [Стъпка 1]  Първата стъпка е да се определи предисторията на играта. Фонът трябва да съдържа различни животни. Имаме три възможности:  1. Учениците сами рисуват фона;  2. Учениците търсят безплатно изображение онлайн;  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-11 at 12.24.23.png3. Осигуряваме предварителни ресурси за учениците (ако искаме да спестим време).  Учениците вече знаят как да добавят фон на сцена, така че го правят индивидуално.  [Стъпка 2]  Втората стъпка е да добавите жената фермер. Имаме същите опции като в първата стъпка:  1. Учениците сами рисуват жената фермер;  2. Учениците търсят безплатно изображение на жената фермер онлайн;  3. Предоставяме образ на жената фермер за ученици (ако искаме да спестим време).  Учениците вече знаят как да добавят нов спрайт, така че го правят индивидуално.  [Стъпка 3]  /Users/tadejanemanic/Downloads/Farm script pic.pngСлед 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.  **Изпълнение**  След това показваме на учениците как да добавят звук в играта. Имаме три възможности:  1. Импортиране на звук от медийна библиотеката на Snap!;  2. Импортиране на звук от нашия компютър чрез плъзгане в Snap!;  3. Запис на нов звук в Snap!  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).  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-12 at 17.44.15.png  След това те избират звука на кучето (Куче 1 или Куче 2).  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-12 at 17.40.11.png  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].  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-13 at 09.10.03.png/Users/tadejanemanic/Downloads/Farm script pic (3).png  [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]*.  /Users/tadejanemanic/Downloads/Farm script pic (7).png /Users/tadejanemanic/Downloads/Farm script pic (6).png  /Users/tadejanemanic/Downloads/Farm script pic (5).png /Users/tadejanemanic/Downloads/Farm script pic (4).png  [Стъпка 6]  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-11 at 14.47.02.png/Users/tadejanemanic/Desktop/Screen Shot 2019-12-11 at 14.34.32.pngСледващата 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]./Users/tadejanemanic/Downloads/Farm script pic (1).png 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**  Студентите 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*  */Users/tadejanemanic/Desktop/Screen Shot 2019-12-13 at 09.44.51.png*  *Фонът*  **/Users/tadejanemanic/Desktop/Screen Shot 2019-12-13 at 09.46.18.png** |
| **Инструменти и ресурси за учителя** | * Цяла дейност в Snap !: <https://snap.berkeley.edu/project?user=tadeja&project=Farm> * Уебсайт с безплатни изображения: <https://pixabay.com/> * Уебсайт на безплатни звуци: <https://www.zapsplat.com/> * Лайович, С. (2011). Драскотина. Научете се програмиране и пощенски компютърни мачки. Любляна: Пасадена. * Vorderman, C. (2017). Računalniško програмиране 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  ● Уебсайт с безплатни изображения: https://pixabay.com/  ● Уебсайт на безплатни звуци: https://www.zapsplat.com/  ● 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:   * движение на обект въз основа на събития; * чувствителност към един или повече цветове; * отчитане на булева стойност в логически изрази; * дефиниране, разграничаване, динамична проверка и реагиране на различни състояния на играта,   Специфични резултати от обучението, ориентирани към алгоритмично мислене:   * ученикът реализира движението на обекта с клавишите със стрелки, използвайки събития и взема предвид ограниченията, * ученикът използва сензорен цветен блок, за да получи булева стойност за четене на еднократно или многократно сензориране, * ученикът осъзнава, че състоянието на обекта може да бъде изразено с цветовете, които обектът докосва, * ученикът прави разлика между две (основни), пет (пълни) различни състояния и знае как да ги изрази с логически изрази, * 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.  **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  Индивидуална работа / работа по двойки / групова работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Размисъл и оценка)  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.  **[Базова версия]**  В основната версия трябва да правим разлика между две състояния.  [Стъпка 1]  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]  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.  След това обсъждаме със студентите какви са различните позиции на хамелеона на сцената и как можем да ги изразим, използвайки блока touching color?.  Има две:   1. Той докосва синия цвят -> Touching color [blue]? 2. Той докосва пясъчния цвят -> Touching color [sandy]?   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.  Защото има само две възможности, които можем да използваме блока за условие “if - else”.  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 a *point in direction* block).   |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  |   [Стъпка 4]  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.  След това обсъждаме с учениците какви са различните позиции на хамелеона на сцената и как можем да ги изразим с докосващ цвят? блок.  Бързо откриваме, че има пет:   1. Той е изцяло от синята част -> Touching color [blue]? 2. Той е между синята и пясъчната част -> Touching color[blue]? AND Touching color [sand]? 3. Той е изцяло от пясъчната част -> Touching color [sand]? 4. Той е между пясъчната и зелената част -> Touching color[sand]? AND Touching color [green]? 5. Той е изцяло от зелената част -> Touching color [green]?   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]*.  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 e, 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.  Предложен шаблон за код: |
| **Инструменти и ресурси за учителя** | * Цялата дейност в Snap!:   Basic:  <https://snap.berkeley.edu/project?user=zapusek&project=chameleon_simple>  Full:  <https://snap.berkeley.edu/project?user=zapusek&project=chameleon>   * Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. * Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK. |
| **Ресурси / материали за студентите** | * Шаблон в Snap!:   <https://snap.berkeley.edu/project?user=zapusek&project=chameleon_template>   * Полуготов файл в Snap!:   <https://snap.berkeley.edu/project?user=zapusek&project=chameleon_half_baked>   * Инструкции за студент (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  Движение на обект с клавиши със стрелки с помощта на събития  Използването на условие за спрайт докосва друг обект.  Използване на събития |
| **Очаквани резултати** | Основни очаквани резултати:   * Условия за спрайт да докосва даден цвят * Придвижване до определени координати * Молив горе, Молив долу * Цвят на молива   Специфични очаквани резултати свързани с алгоритмично мислене:   * Учениците използват оператор if за проверка дали даден обект докосва даден цвят * Учениците задават начални координати на спрайт * Учениците използват блокове „молив горе“ и „молив долу“ за изчертаване на линия/пътека на движение. * Учениците променят цвета в ззависимост от двойката, която свързва. * 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.  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.  **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  Индивидуална работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)  Първоначално се дава на студентите:   * Фон * Спрайт на момиче * Код на движение за една посока   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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/Screen Shot 2019-12-11 at 12.48.37.png  [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:  /Users/mateja.bevcic/Downloads/C4G7_Pomagaj_princu_in_princeski - Cela script pic (2).png  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.  /Users/mateja.bevcic/Desktop/slike LS/7/Screen Shot 2020-03-27 at 11.49.41.png  [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*.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/8.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/9.png  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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/10.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/1_risanje_park/11.png  [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.  /Users/mateja.bevcic/Desktop/slike LS/7/Helping Prince and Princess to find their animals script pic (5).png  [Окончателен код]  Момиче    Например Принцесата  /Users/mateja.bevcic/Desktop/slike LS/7 Screen Shot 2020-03-27 at 11.21.09.png |
|  | [Допълнителни задачи]  Учениците могат да добавят допълнителни задачи според техните желания или могат да следват задачите по-долу:   * 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. * Добавете друг спрайт (животно) за момичето. * Всеки спрайт трябва да рисува с различен цвят. * Регулирайте първоначалните инструкции. * Добавете инструкции за преместване на спрайт и рисуване, като щракнете върху спрайт. Напр. принцесата казва: „Премествате 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> * 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>   * Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. * 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>   * 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 * Рисуване с молив (pen up, pen down, set color) * Преместване на определен брой стъпки * Използване на цикъл * Използване на събития |
| **Очаквани резултати** | Основни очаквани резултати:   * Цикъл repeat * Завъртане на 90 градуса * Определяне посока на движение (Point in direction) * Смяна фон на сцена   Специфични очаквани резултати, свързани с алгоритмично мислене:   * Ученикът използва цикъл repeat когато се повтарят едни и същи блокове няколко пъти * Ученикът използва завъртане на 90 градуса когато изчертава различни фигури (квадрат, правоъгълник, буква “T”) * Ученикът описва заначението на установяване на посока на движение 90 * 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“.  **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.  **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  Индивидуална работа / Работа по двойки |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)  Първоначално се дава на учениците:   * Три фона с всички точки, които трябва да свържат * Крейда спрайт   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!  [Стъпка 1]  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/boardS.png  Учениците започват с този фон. Те пишат код за рисуване на квадрат. Започвайки от точка "А", 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).  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/1.png  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*.  *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.  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*.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/2.png  Ако искаме да видим какво нарисуваме, трябва да поставим блок *pen down* преди цикъла *repeat*.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/3.png  Ако искаме тебеширът да не се върти при завъртане, щракваме върху блока don’t rotate in direction.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/rotation.png  [Стъпка 2]  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).  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*.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/4.png  [Стъпка 3]  След изчертаването на квадрат искаме да нарисуваме правоъгълник. Това означава, че трябва да променим фона. Ще направим това с две стъпки:   1. Щракваме върху фона (име на дъска, от дясната страна на екрана).   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/back1.png  Щраквайки върху *Backgrounds* можем да видим трите фона (*boardSquare, boardRectangle, boardT*), които предварително са създадени за този урок.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/back2.png  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*.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/back3.png   1. Щракваме обратно върху тебешира.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/back4.png  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“.“  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/12.png  [Стъпка 4]  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/boardR.png  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/8.pngСлед 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 „Т“. Това означава, че трябва да сменят фона, така че в тази стъпка те действително повтарят [Стъпка 3], просто сменят буквата („T“) и костюма (boardT):   1. 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*.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/9.png   1. 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 „T“.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/13.png  [Стъпка 6]  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/boardT.png  След натискане на клавиша “T”, 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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/11.png  [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].   1. 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*.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/15.png   1. 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“.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/14.png  [Финален код]  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/2_risanje_kvadrat/final code.png  [Допълнителни задачи]  Учениците могат да добавят допълнителни задачи според техните желания или могат да следват задачите по-долу:   * Добавете нов фон и нарисувайте няколко точки. * 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> * Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. * 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>   * 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 * Задаване размер на спрайта * Добавяне на текст към спрайта * Придвижване на обект със стрелки, използвайки събития * Използване на условия за проверка дали даден обект докосва друг обект |
| **Очаквани резултати** | Общи очаквани резултати:   * Променливи * Показване и скриване на спрайтове * Дублирани спрайтове * Дублиран блок от код * Условия   Специфични очаквани резултати за развитие на алгоритмично мислене:   * Ученикът използва променлива за преброяване на събраните отпадъци * Ученикът използва скрит спрайт, когато е докоснат спрайт и показва спрайт в началото * Ученикът знае как да дублира спрайт (от една бутилка до например 4 бутилки) * Ученикът знае как да дублира блок код (от спрайт на бутилка на хартиен спрайт) * 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.  **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.  **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  Индивидуална работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)  Първоначално се дава на студентите:   * Фон * Спрайт за момичета (с кода за движение), спрайт за бутилки, спрайт за хартия и спрайт за боклук   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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/ozadje.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/3.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/1.png/Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/4.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/2.png  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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/5.png  [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?  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/6.png /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/7.png  Избираме оранжев блок *Variables*, след това щракнете върху бутона *Make a variable*, запишете име на променлива (*Variable name) и щракнете върху* OK. Появявасе блок *points* appears.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/8.png  Ако квадратчето е отметнато, променливата с нейната стойност ще се вижда на екрана:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/9.png  В началото на играта стойността на променливата трябва да бъде 0, тъй като няма прибран боклук. Под кода от [Стъпка 1] студентът добавя блок *set \_\_ to 0. Чрез щракване върху падащото меню те избират подходяща променлива, която е* *points*.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/10.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/11.png  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).  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/12.png  [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).  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/13.png  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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/14.png  И го пуснете в хартиения спрайт, като щракнете с мишката върху спрайта за хартия.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/15.png  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.  [Стъпка 6]  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.  Също така този код ще се активира, когато момичето го докосне.  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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/16.png  [Финален код]  Момиче  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/final code girl.png  Бутилка/ Хартия  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/final code bottle - paper.png  Кошче  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/3_spremenljivke_smeti/final code trash can.png  [Допълнителни задачи]  Учениците могат да добавят допълнителни задачи според техните желания или могат да следват задачите по-долу:   * Добавете друг вид отпадъци (например биоотпадъци). * В кошчето пише напр. „Взехте X бутилки, Y хартии и Z дини“. * Ако играч вземе цялото кошче, кошчето казва: „Поздравления! Взехте всички боклуци! “ * 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> * Допълнителни задачи (възможни решения): <https://snap.berkeley.edu/project?user=mateja&project=Picking%20up%20trash%20and%20cleaning%20the%20park%20%2B%20Add.%20Task> * Lajovic, S. (2011). *Scratch. Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. * 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> * Инструкции з аучениците (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) * Извеждане на текст (block say) |
| **Очаквани резултати** | Основни очаквани резултати:   * Задаване и учеличаване стойност на променлива, * присвояване стойност на променлива вътре / извън цикъла, * цикъл for (repeat n times), * случайни числа, * слепване на стрингове, * логически и аритметични оператори, * вход   Специфични цели на обучение ориентирани към алгоритмичното мислене:   * Ученикът разпознава ситуацията за използване на повторение n цикъл пъти, * Ученикът прави разлика между присвояване на стойността във всяка итерация на цикъла и веднъж преди цикъла. * Ученикът използва блок за въвеждане, за да получи номера от играч, * Ученикът знае как да използва аритметични оператори, за да генерира правилния отговор, * Ученикът използва if - else, за да провери правилността на въведеното от играча и дава подходящ отговор, * 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.  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.  **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  Индивидуална работа / работа по двойки / групова работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)  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>   * Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. * 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>   * 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) * Извеждане на текст (block say) |
| **Очаквани резултати** | Основни очаквани резултати:   * Случайни числа, * Присвояване стойност на променлива, * Вход, * Цикъл repeat until, * Оператори за сравнение, * Брояч   Специфични очаквани резултати, ориентирани към алгоритмично мислене:   * ученикът присвоява произволна стойност на променливата, * ученикът използва блок за въвеждане, за да получи номера от играч, * ученикът използва повторение до цикъл, за да поиска многократно играч да въведе числото и да извърши тестване на стойност * ученикът извършва тестване на стойност с оператори за изречение и сравнение и дава подходящ отговор, * ученикът задава условието на цикъла за повторение, за да тества дали играта е приключила, * ученикът осъзнава, че не трябва да тества дали играта е приключила, тъй като тя е намерена в състояние, * 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.  **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.  **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  Индивидуална работа / работа по двойки / групова работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)  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. Студентите 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 й. 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]  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 Snap!: <https://snap.berkeley.edu/project?user=zapusek&project=cats_in_a_shelter> * Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. * 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>   * Инструкции (C4G11\_InstructionsForStudent\_BG.docx |

Сценарии за напреднали

Учебен сценарий 12 – Улов на здравословна храна

|  |  |
| --- | --- |
| **Учебен сценарий Име** | Улов на здравословна храна |
| **Предишен опит в програмирането** | * Добавяне на текст към спрайт * Показване и скриване на спрайт * Използване на посока на завъртане (point in direction) * Използване на случайни числа * Използване н апроменливи за изчисляване на точки * Използване на цикъл repeat * Using forever loop * Using conditionals |
| **Очаквани резултати** | Основни очаквани резултати   * Променливи * Условия * Цикли * Посока на завъртане * Случайни числа   Специфични очаквани резултати, ориентирани към алгоритмично мислене:   * Ученикът използва променлива за предотвратяване започването на играта преди момичето да приключи да говори (по избор) * Ученикът използва if за проверка (с помощта на променлива) дали храната може да започне да се движи * Ученикът използва цикъл (repeat until) за движение на храната, докато точките не са по-малки от 5 * Ученикът използва посока на завъртане (point in direction) 180 (надолу) за спрайтове, движещи се надолу * Ученикът използва произволно число за избор на брой стъпки * Ученикът използва случайно число за преместване в произволно положение * 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!  **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.  Освен това учениците могат да добавят променлива и ако изявление за предотвратяване движението на храната, преди момичето да спре да говори.  **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)  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.    [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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/2.png  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).  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/6.png  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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/4.png/Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/ozadje1.png  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)./Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/ozadje2.png  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/5.png  [Стъпка 3]  Учениците вече могат да направят променлива, точки, която ще използват за броене. Точките трябва да имат начална стойност 0 (спрайта на момичето)./Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/11.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/7.png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/9.png  [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).  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/10.png  [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:    /Users/mateja.bevcic/Downloads/Catching healthy food script pic (1).png  [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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/4_lovljenje_hrane/16.png  [Финален код]  Girl/Момиче    Apple/ябълка    [Допълнителни задачи]  Студентите 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. * 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). * Променете правилата - оставете играта да приключи, когато играчът набере 20 точки (той печели) или когато вземе 3 нездравословни храни (загуби). * Добавете още хранителни спрайтове, за да направите играта по-интересна. * 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> * Дейност в Snap! С допълнителни задачи: <https://snap.berkeley.edu/project?user=mateja&project=Catching%20healthy%20food%20%2B%20Add.%20Task> * Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. * 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>   * Инструкции (C4G12\_InstructionsForStudent\_BG.docx) * 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 * Използване на посока на движение (Point in direction) * Използване на условни изрази и условен блок * Използване на блок К*ажи* * Използване на блок Изчакай за ... секунди |
| **Резултати от обучението** | **Общи резултати от обучението:**   * Основни очаквани резултати: * Преместване и промяна на размера на спрайт * Промяна на размера * Съобщения * Съставете структура на разказването на истории * Промяна на фона на сцените   **Специфични резултати от обучението, ориентирани към алгоритмично мислене:**   * Ученикът планира диалози и дейности на спрайтовете в историята * Ученикът изпращане на предавания за диалог между спрайтове * Ученикът използва движещи се и променящи се размери за спрайтове * Ученикът използва показване и скриване на спрайта * 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.  Разговорът е представен на картинката.  **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.  **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)  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>  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“.  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 | |  |  | |  |  | |  |  |  1. 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.   1. Обсъжда се групово как да се довърши историята от картинката. Как да се наименоват съобщенията: напр. Съобщението от Cat към Alice да бъде Alice2, A от Alice to Cat – Cat1. 2. Учениците довършват историята по двойки. 3. 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. 4. Учителят демонстрира проектът <https://snap.berkeley.edu/project?user=ddureva&project=Alice>         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.  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).  След съобщението на Заека Alice 1 историята продължава, така както е направена в проекта Alice 2.   1. 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. 2. 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!:  <https://snap.berkeley.edu/project?user=ddureva&project=Alice> |
| **Ресурси за учениците** | <https://snap.berkeley.edu/project?user=ddureva&project=Alice_1>  <https://snap.berkeley.edu/project?user=ddureva&project=Alice_2>  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. |
| 1. 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) |
| 1. 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.  Със смяната на сцената Заекът продължава да разказва историята.  По-късно се провежда разговорът между алиса и Чеширския котарак. |

Герои (Спрайтове)

|  |  |  |
| --- | --- | --- |
| Спрайт | Действие | Фон на сцена |
|  | При старт:  Казва: Здравей! за 2 сек.  Казва: Чувал ли си някога за Алиса и нейните приключения в Страната на чудесата? за 6 сек.  Казва: Сега нека да видим нейната история! за 6 сек.  Изпраща съобщение Go to forest /Иди в гората |  |
|  | При старт  Скрива се от сцената. Позиционира се в центъра на сцената и се задава размер 100% в готовност за показване в новия фон. |  |
|  | При старт  Скрива се от сцената. Позиционира се в х: -74, у:113 (Позициите са предварително определени след наместване на спрайта на котарака върху сцената Meeting.) |  |
|  | Получава съобщение Go to forest:  Показва се  5 пъти се повтаря: изчакване за 1 сек.; преместване с 5 стъпки; намаляване на размера (смяна с -10); „Мисли“ – „Къде съм?“  Подготвя се за следващия декор: Изчаква 5 сек; Възстановява размера си (смяна 100%) и се позиционира в х:-187, у:-67  Изпраща съобщение: Meeting with Cheshiere Cat/ Среща с котарака. |  |
|  | Няма действие. Само е показан от предходния декор. |  |
|  | Получава съобщение: Meeting with Cheshiere Cat.  Сменя размера си на 80%  Казва: „Алиса спряла н акръстопътя и се чудела на къде да тръгне.“ за 10 сек.  Казва: „Тя видяла Чеширския котарак на дървото.“ за 8 сек.  Изпраща съобщение Alice1 |  |
|  | Получава съобщение: Alice1.  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).  Казва: „Hi!“ за 2 сек.  Казва: „Would you tell me please, which way I ought to go from here!“ за 10 сек.  Изпраща съобщение (broadcast) към Котаракът: Cat1. |  |
|  | Получава съобщението Cat1.  Показва се.  Казва: „А ти къде искаш да отидеш?“ за 10 сек.  Изпраща съобщение „Alice2“ |  |
|  | Получава съобщение: Alice 2.  Казва: …………………………………………………………………………  Изпраща съобщение Cat2 |  |
|  | Получава съобщение Cat2.  Казва: …………………………………………………………………………  Изпраща съобщение: Rabbit1 |  |
|  | Получава съобщение Rabbit1  Казва: „Каква е поуката от тази история?“ за 8 сек.  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   * Задаване на посока на завъратнае (point in direction) * Използване на променливи за изчисляване на точки * Използване на цикъл * Използване на разклонения (условия) |
| **Очаквани резултати** | Осовни очаквани резултати:   * Променливи * Разклонения и условия * Цикъл * Посока на джижение * оператори   Специфични очаквани резултати, ориентирани към алгоритмично мислене:   * ученикът използва писалка за рисуване * ученикът използва цикли за рисуване * ученикът променя стойността на променлива при рисуване * ученикът използва точка в посока, за да рисува предмети на сцената * ученикът използва излъчване за управление на спрайт * ученикът използва условни условия за смяна на сцената * 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!  **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)  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.  [Стъпка 1]  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).  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.    За да нарисува ствола на дъба, спрайтът трябва да се обърне надясно на 90 градуса и да се премести с 10 стъпки.    Тази част се повтаря три пъти.    Когато рисуването е готово, е необходимо да вдигнете молива нагоре, така че да не очертава линията, когато спрайта на молива се движи.    След изрисуването на дъба, моливът трябва да се премести в произволна позиция.    [Стъпка 4]  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.    [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    Stage |
| **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) * Инструкции за ученици (C4G14\_InstructionsForStudent\_BG.docx) |

Учебен сценарий 15 – Хвани мишката

|  |  |
| --- | --- |
| **Учебен сценарий Име** | Хвани мишката |
| **Предишен опит в програмирането** | * Ученикът добавя фон. * Ученикът добавя нов спрайт. * Ученикът добавя нов звук. * Ученикът кара спрайт да каже нещо. * Ученикът променя костюма на спрайта, за да направи анимация. * Ученикът е реализира движението на обекта с клавишите със стрелки, използвайки събития и взема предвид ограниченията * Ученикът прави разлика между две различни състояния и знае как да ги изрази с логически изрази. * Ученикът използва условни условия. |
| **Очаквани резултати** | Основни очаквани резултати:   * Цикъл forever; * Случайни числа; * Брояч; * Таймер.   Специфични очаквани резултати, ориентирани към алгоритмичното мислене:   * ученикът използва цикъл forever за преместване на спрайтовете; * ученикът използва случайни числа, за да определи позицията на спрайта, премества спрайта за произволни стъпки и завърта спрайта за случайни градуси; * ученикът прилага брояч за отчитане на улова на мишки и използва крайната стойност, за да обобщи доколко е успешен играчът; * 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.  **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.  **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  Работа по двойки / групова работа |
| **Резюме на обучението** | (Мотивация-Въведение, Прилагане, Рефлексия и оценка)  Мотивация-Въведение  Мотивираме учениците, като показваме играта. Обсъждаме с тях как биха започнали да програмират тази игра. Заедно със учениците определяме последователността на стъпките, например:   1. изберете фон и добавете спрайтове; 2. програмирайте котката да се движи с клавишите със стрелки; 3. програмирайте мишката да се движи произволно; 4. програмирайте мишката да се скрие (и да се появи на произволно място), когато докосне котката; 5. брояч на програми; 6. добавете таймер и определете края на играта; 7. добавете момичето и я програмирайте, за да обобщите колко успешен играч е бил; 8. програмирайте момичето да скача, когато докосне мишката; 9. добавете звук на котката / мишката; 10. и т.н.   Учениците могат да помогнат със стъпките или да създадат свои собствени правила на играта (но трябва да следват смели стъпки).  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 09.22.42.png  Въвеждаме оператор за случайно присвояване на стойност.  /Users/tadejanemanic/Downloads/Catch the mouse script pic (1).png  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.  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 09.43.14.png /Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 09.43.25.png  [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. Студентите правят това с блоковете *Motion/move[x]steps* и *Motion/turn[x]degrees*, в които тв вмъкват оператора *pick random[x]to[y]* operator.  /Users/tadejanemanic/Downloads/Catch the mouse script pic (3).png  [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]*.  /Users/tadejanemanic/Downloads/Catch the mouse script pic (6).png  [Стъпка 5]  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]*).    [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 х мишки! "  /Users/tadejanemanic/Downloads/Catch the mouse script pic (9).png  [Допълнителни задачи]  Учениците могат да добавят всякакви елементи към играта си. Например момичето, което скача всеки път, когато докосне мишка.  /Users/tadejanemanic/Downloads/Catch the mouse script pic (10).png  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*  /Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 11.37.43.png  *The background*  */Users/tadejanemanic/Desktop/Screen Shot 2019-12-17 at 11.38.29.png* |
| **Инструменти и ресурси за учителя** | * Цялата дейност в Snap!: <https://snap.berkeley.edu/project?user=tadeja&project=Catch%20the%20mouse> * Безплатни изображения: <https://pixabay.com/> * Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. * 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>   * Безплатни изображения: <https://pixabay.com/> * 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 * Показване и скриване на спрайтове * Използване на оператори * Използване на променливи * Използване на слепване на стрингове * Използване на условия и разклонения |
| **Очаквани резултати** | Основни очаквани резултати:   * Променливи * Условни блокове * Оператори   Специфични очаквани резултати, ориентирани към алгоритмично мислене:   * Ученикът използва променливи за определяне на цената за различни спрайтове - храни * Ученикът променя стойността на променливите, тъй като бюджетът се променя, когато играчът купи храна * Ученикът използва if оператор за проверка на наличността на пари * Ученикът използва оператори за присъединяване на текст - стойност на променливите - текст * Ученикът използва оператори за сравняване на цени и бюджет * 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.  **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.  **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)  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.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/ozadje.png  [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 .:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/Grocery 2 script pic.png  [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:   1. *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).   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/3.png   1. *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:   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/4.png   1. *When clicked*: тук трябва да направят малка рефлексия (да помислят). 2. В кой случай играчът може да закупи продукта и в кой не? 3. Какво се случва с бюджета, ако той купи храната? 4. Как да броим закупените продукти? 5. Какво се случва с храната на рафта? 6. 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.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/5.png   1. 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*.  Студентите могат да добавят малко текст и тук.  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/6.png   1. Преброяването на броя на закупените продукти ще се осъществи с промяна на променливата *healthy\_food* va с 1.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/7.png   1. Когато върху храната се щракне, тя се скрива.   /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/8.png  Едно възможно репение е:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/9.png  [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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/10.png  [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:  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/12.png  [Финален код]  Girl  Food  /Users/mateja.bevcic/Desktop/Learning Scenario/igre/5_nakupovanje/Final Food.png  Finish Button    [Допълнителни задачи]  Учениците могат да добавят допълнителни задачи според техните желания или могат да следват задачите по-долу:   * Променете играта, за да можете да купувате всяка храна 3 пъти. * Дайте повече пари на играча в началото. * В края момичето разказва също колко продукти сте закупили. Напр. “Купихте 2 пъти диня, 1х грозде, 2 пъти пържени картофи”. |
| **Инструменти и ресурси за учителя** | * Цялата дейност в Snap!: <https://snap.berkeley.edu/project?user=mateja&project=Buying%20food%20for%20a%20picnic> * Допълнителните задачи (възможни решения): <https://snap.berkeley.edu/project?user=mateja&project=Buying%20food%20for%20a%20picnic%20%2B%20Add.%20Task> * Lajovic, S. (2011). *Scratch. Nauči se programirati in postani računalniški maček.* Ljubljana: Pasadena. * Vorderman, C. (2017). *Računalniško programiranje za otroke.* Ljubljana: MK. |
| **Инструменти и ресурси за учениците** | Инструкции за учениците  (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; * Използване на случайно число за избор на костюм на сцената и на спрайта. * Използване на цикъл за повторение * Използване на условни блокове * Използване на операции за сравнение * Използване на блок за диалогов прозорец (попитайте ... и изчакайте) * Използване на съобщения за синхронизиране на действията |
| **Резултати от обучението** | **Общи резултати от обучението:**   * + Променливи   + Условни блокове   + Цикъл   + Сензорни блокове   + Синхронизиране на действията чрез съобщения   **Специфични резултати от обучението, ориентирани към алгоритмично мислене:**   * Ученикът използва променливи за преброяване на точки и за костюми на сцената и на запазването на спрайт. * Ученикът използва променливи за преброяване на точки * Ученикът инициализира променливи за преброяване на точки * Ученикът използва условни и логически операции * Ученикът използва съобщения за промяна на спрайта и изчисляване на крайния резултат. |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:**  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.  **Задача:** Студентите 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.  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.  **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)  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>.     1. Обсъжда се как да се формулира условието на задачата. Прави се формулировка на задачата.   *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.*   1. Коментират се променливите и начина на тяхното дефиниране, инициализиране и промяна на стойността им. 2. Припомнят се командите за random number, аритметичните и логическите операции, broadcast event 3. 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.  C:\Users\leo1.Portege\AppData\Local\Temp\operations3 script pic.png  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)      1. 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.   1. Щракни с десния бутон на мишката върху дадена операция. Ще с епояви контекстно меню.      1. Появява се списък с операции.      1. Избери операция   Забележка: Ако възрастта и знанията на учениците за аритметичните операции позволяват задачата може да се разшири с операциите степенуване (^) и деление по модул (mod)   1. 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!: <https://snap.berkeley.edu/project?user=ddureva&project=operations3>  Цялата дейност е в Scratch:   * 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>  Инструкции за учениците  (C4G17\_InstructionsForStudent\_BG.docx) |

Учебен сценарий 18 – Рециклиране

|  |  |
| --- | --- |
| **Учебен сценарий** | Рециклиране |
| **Предишен опит с програмирането** | * Показване и скриване на спрайт * Използване на променливи за преброяване на точки * Използване на цикъл завинаги * Използване на условни блокове * Използване на операции за сравнение * Използване на чувствителност на цветовете |
| **Резултати от обучението** | **Общи резултати от обучението:**   * + Променливи   + Условни блокове   + Цикъл   + Придвижване до точка в посока   + Сензорни блокове   + Рефакторинг на код   **Специфични резултати от обучението, ориентирани към алгоритмично мислене:**   * + Ученикът използва **Изчакай до (Wait until)** и логически операции за приключване на играта   + Ученикът използва **Изчакай до** **(Wait until)** и блок за промяна на сцената   + Ученикът използва променливи за преброяване на точки   + Ученикът използва условни и логически операции   + Ученикът сравнява кодовете на подобни спрайтове. * Ученикът прави рефакторинг на кода * Ученикът използва позициониране на спрайтовете (в допълнителна задача използвайте произволно позициониране) |
| **Цел, задачи и кратко описание на дейностите** | **Кратко описание:**  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.  **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:   * + промяна на положението на отпадъчните спрайтове с произволен избор на координати на спрайтовете;   + намалете броя на етапите и извлечете робота като отделен спрайт. (Роботът е част от фона на сцената).   **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 игрален 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)  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.   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?      1. 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.     1. 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.       1. 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.  По преценка на учителя може да се усложни условието на задачата:   * 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.      * 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!: <https://snap.berkeley.edu/project?user=ddureva&project=recycling>  Цялата дейност е в Scratch:   * 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>  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;  Използване на събитие Когато съм под натиск  Използване на цикъл за повторение (repeat loop)  Използване на условни блокове  Използване на съобщения за промяна на костюмите на сцената и за управление на дейностите на спрайта |
| **Резултати от обучението** | **Общи резултати от обучението**:  • Променливи  • Условни блокове  • Цикъл  • Съобщения;  • Звуци;  • Програмиране на музика;  **Специфични резултати от обучението, ориентирани към алгоритмично мислене:**  • Ученикът използва променливи за преброяване на точки.  • Ученикът инициализира променливи за преброяване на точки  • Ученикът използва условни блокове, за да оцени постигнатите точки  • Ученикът използва съобщения събитие за смяна на костюмите на сцената и за действията на спрайтове.  • Ученикът използва блокове от Група Звук, за да композира мелодии;  • Ученикът идентифицира необходимостта от цикъл за повторение, за да намали броя на блоковете в скриптовете.  • Ученикът разширява функционалността на играта. |
| **Цели, Задачи и кратко описание на дейностите** | **Кратко описание:**  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.  **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.  **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 игрален 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)  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.  2. Учителят демонстрира част от играта с композиране на мелодия. <https://snap.berkeley.edu/project?user=ddureva&project=Play_a_Piano_1>     1. Показва кода и обяснява използването на командите от групата Sound.   В Snap! може да се използват звукове от вградената библиотека, файлове от компютъра или да се свирят музикални тонове на различни инструменти.  За избор на инструмент се използва командата:  , Забележка. В Scratch инструментите са много повече.  Учениците тестват звука на различните инструменти.   1. 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.   1. 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)   1. 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) 2. 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!:  <https://snap.berkeley.edu/project?user=ddureva&project=Play_a_Piano_1>  <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>  Инструкции за учениците (C4G19.1\_InstructionsForStudent\_BG.docx) |

Учебен сценарий 19.2 – Свири на пиано

|  |  |
| --- | --- |
| **Учебен сценарий**  **Име** | Свири на пиано |
| **Предишен опит в програмирането** | * Използване на цикъл repeat * Използване на променливи * Използване на условни блокове |
| **Резултати от обучението** | Общи цели на обучението:   * Условия * Цикли   Специфични цели на обучението, ориентирани към алгоритмичното мислене:   * Ученикът използва цикъл с повторение за възпроизвеждане на музика * Ученикът използва код, за да накара спрайтовете да реагират на въвеждането * Ученикът добавя звуци към спрайт * 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.  **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.  **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.  [Стъпка 1]  Отворете програмата *Play a piano.* Програмата съдържа всички фонове и спрайтове, необходими за тази задача.  Даден е фонът, а също и спрайтът за клавиш C и един черен клавиш.  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:    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.  Ако черният клавиш е скрит зад белите клавиши, използвайте следния код    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.  https://lh4.googleusercontent.com/pmiOVv3KDx9Lqme3HaZ7wqSNhkyRIguqCCiLscVKGv6A2XGFf7K9qgR3g7IVPI7LJpljJ44Pncj21EdAyMcDgOUelrI9qFW-udwahQgYLkDI5LlGsftDS51AXbGKfAQc6JZYBtc  [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.  https://lh5.googleusercontent.com/8MF8aW7s-GatgdsXipBFNS93dBM8ZNpjEUtio3FFKukBrnqvMFCSw7h5jVQigX2pb55--lJlPpkTCSURZu9l7wjfSq6oeaRdyvyUsUYW1VYV7iIN29iWt7VnedrfhrJNHkO1-rU  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.  https://lh5.googleusercontent.com/zN_OWT89pHZ3gh81ciL8OMpmClNgQ3_tpUf2E_rk6xBmtArXOqVe6NptTF8hMD2RrJNPANONbQa7XR82o6h5vJnmbC1gTLbDrBunroxrZBssFXuQ3BmwJygtM9uBnuOhQ8wtwUA  [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 c Sprite, и копирайте останалата част от кода от блока "When I am clicked".  https://lh4.googleusercontent.com/qjEJtsH7EVaZ87nmK3pnogMIRQpDy12BtzojPJEhekD_zq0Ka5pocFgAXY_Ynqc0blCH7T4moiFaV5VQMxJsszvuCfB3K3hbpYabw0PngDzDZ6vyTQh9N08f0wQELwVVJH8E-eA  Забележете, ако клавиша "c" на клавиатурата е задържан, звукът ще се повтаря, докато клавишът е натиснат. Това се случва, защото съобщението "с" се излъчва многократно. За да спрете излъчването на съобщение, в края на кода добавете блок "изчакайте до""wait until" от палетата за контрол.  https://lh4.googleusercontent.com/SBynFO7adNAUvFhZhsmwqt1xpsvHGXjxb9r7hoXlSN7nhGmnduDAwkS9dytMbgTSyxfPkPKO7DbCMwxoK6ZnArj0F1dOYjOoQtqxVOXMP1CgS2UQeSQ8ZwqRS_VKftofOcWjk8g  За да завършите излъчването на съобщение, използвайте оператора "not" и добавете блок "key c pressed".  https://lh6.googleusercontent.com/fZDsQvwq8EXpzIPClN5maHY5JbZ2fbbxy75qYmYl_4AwLjGxqLTpPTc40hzJuhNtQlh9nGHVVyrKu67ODRvfFfkyvLxdS-yYZSPD7VKcAwAiaP2xLnUhG3tMuumLPLZhxi0xHc4  Направете същото за останалите бели клавиши.  [Стъпка 5]  Време е да програмирате петолинието, ключът сол и нотите, които ще се изпъляват:  https://lh4.googleusercontent.com/2UJVTx-LwmwOaOYvzd6RTblRpTDwXrwdFquL8-3ZPG0b3rNO2xthejEpP5QBN3Am-Cbwxe-nT3aLTcNGF8Rq7AUnFm9UE_Lhq7RDasW_GCam2NN2dGO4ceAcBJfsWfFv0sukufs  За да покажете нотите, активирайте излъчването на съобщението "chords", когато натиснете бутона.  https://lh4.googleusercontent.com/AyPAYK55V9RocYTL1VgTEQP7uyF781Akmn4M6343SG2u0tNtGl-bsPo-0GIaNPfHAB0Suwdh_SASKQawgoEoFqnVfZEWbwaT4DAjppcXcHl8tZ-UNy32GiqZux_jNC-MnuPzbrw  Импортирайте нов костюм "chords" за тази сцена.  https://lh3.googleusercontent.com/TiJxKAHgUuc_mRR3nt0zkr_eRW_IoTWs0Bi8wg2XpYcAGHabrbj0vAqTUm0q5ATssStEv8KGOEXFti3t0kbvrhLVkSpLsWeFBSsc5s8VqSj0nZ3kZWSujWuzRTvspa2TYjmSlmU  И накрая, добавете код, който позволява на сцената да промени костюма на "chords", когато получи съобщение "chords".  https://lh6.googleusercontent.com/PyLrO94zcfY5GTcQPkXbfsbeYFnyG10-180nvElvVX-fZ7rWXIFAPjnlyRjjbJX0rBb3vPWjRgKByLi5mbhr_RDl4bnQMXp93pi08P--DzTk6-Bu0BErVqx3QYlfxon_hO0lql0  [Стъпка 6]  Намерете спрайт с костюм нота. Сега е време да програмирате клавиша за нота за възпроизвеждане на песента, която трябва да се повтори.  https://lh4.googleusercontent.com/mQ1mDLoE4QTZHwGmEvQe1J2iXRr3dwHdiclDum6KvxnndTUrdkHRSPLLT4aT1Cbv04ccoiRs1iiWfrkVfpwLbJhmTGwl0ShuFDz4Lr1V7BHwJdyvuHriNBMwrOpjbgB1VgmJgeY  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.  https://lh3.googleusercontent.com/RULXuvtXbL3sbnK1yw2MBFQoMp6sm7adUDxfuUjtc7-xcplJNq1svvKCsiMUG-HIJqDjVNng71UN_Mm9MPy5CuuYsTCY6TSmqRjmVVQGwHpAV6Oh0lrMRyL2aFvSM6PHiCJpV5c  [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.   https://lh6.googleusercontent.com/womDK0-yotcDamMu2vb7LzLUgsrTcDKaRTuFUy9d_8pLX2h-M_I6tXw3I27sl_C2cLVbyobe3Nc03tdU98dYIT_is7XuPlj5-sKzvR2OgacQBfO9BWbG0mx25slzwXGH6RX7h0k https://lh3.googleusercontent.com/CFvBjyaUJhjcjKzJSVQlkQ8YL_TzW7gK5sbqqqnTvN3XaypLwluDpMGEol_bFkG7ZWt3COKrSYmv7TQgejUBOxiSjHDOk3fDyMxwRIxqADnW_pfmx5596aI0e12sSmHrI3k8MAA  В края добавете към сцената блок “When I receive”, за да промените костюма към „blank“, след като получите съобщението „blank“.https://lh5.googleusercontent.com/H6wopDzfRFhH6oXrSg1gI7_tSokGoA2rRQVrsa7h8dIOdyBWSmYHd-f04BulpAzJIReoUOnfZXDHPDMQp9Uh-1S26ukh-LDbNB9L0nbwAyq5u6I3f-G_M7oQX7cIltrGusyFrtU  [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]  A клавиш  https://lh6.googleusercontent.com/j95Bwr-0EEFBMc54FYUuwRbuaKp-_BIvvx085LSfMJ5bgYsGJCuOT363BsvclhFA-TYsTrkR8ZlO3GR_GlzLiHaFQGiVcy79Q9bI9g5KoQ3OxcGb3HE7rA7q2cIdvVXBV8oKMtA  Клавиш цигулка  https://lh4.googleusercontent.com/6FM7puQhFDEXzQvMqFVQ-0olZqIVZFmW5eVgmaF79IyqdcMFUgESUvRa0qZF6v2RPPAmLYLREeBayXhd-X_ZOgwe9TBigowIg0aoDDCxzTy7HZcjnaKO0kQo7kL_gNc6bw6YYIM  Нота  https://lh3.googleusercontent.com/RULXuvtXbL3sbnK1yw2MBFQoMp6sm7adUDxfuUjtc7-xcplJNq1svvKCsiMUG-HIJqDjVNng71UN_Mm9MPy5CuuYsTCY6TSmqRjmVVQGwHpAV6Oh0lrMRyL2aFvSM6PHiCJpV5c  X  https://lh4.googleusercontent.com/vQQ44DONsWjHrj2_pmjqqMD6XVpYJgmTqoxtKfeteSOKQSM4PUpG4B74UI_3hknXea4IWruZ-XMzHa9xIjZ6gU0Nr-hidxH2fstpA5hx-ONZ-2tYMeCeGk4olAs_C-eEWM4voFw  Сцената  https://lh4.googleusercontent.com/C1MBmMIzBdwrUpXkgYGhBn3rLB9kfGcSz0rizqLq5_bvbwYdhgOfo3EbiNR6bcl5dKb7vf9Niq2D1Sm1AuufKU2bkZkKUUOCJxLAGYPctM2ct-H9YJxCTe6vLnFbA_jw7KBW_UY |
| **Инструменти и ресурси за учителя** | Цялата дейност в Snap!:  <https://snap.berkeley.edu/project?user=ifrankovic&project=Play%20a%20Piano> |
| **Ресурси/материали за ученика** | Полуготов файл в Snap!: <https://snap.berkeley.edu/project?user=ifrankovic&project=Play%20Piano> (27.1.2020)  Изображения:   * Sprite изображения:   + a.png, a1.png   + b.png, b1.png   + violin\_key.png * Backgrounds: notes.png * 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 * Употреба на променливи за преброяване на събраните точки * Използване на цикъл forever * Използване на условни блокове * Използване на оператори за сравнение * Използване на сензорни блокове * Промяна на сцена |
| **Резултати от обучението** | Общи цели на обучението:   * Променливи * Условия * Цикъл * Сензорни блокове   Специфични цели на обучението, ориентирани към алгоритмичното мислене:   * Учениците използват условия за да преценят отговора - правилно или грешно * Учениците използват блокове за смяна на костюма на сцената * Учениците използват променливи за броене на точки * Учениците използват логически операции * 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!  **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 разщирете 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)  1. Трябва да създадете игра-тест, за да тествате знанията си по програмиране.  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/  3. Трябва да използвате алгоритъма за задаване и отговор на въпроси:   1. преминаване към сценичен костюм (съдържа въпрос); 2. задаване на костюм на Аби за задаване на въпрос; 3. Аби казва - Отговорете с Да или Не; 4. Играчът въвежда отговор - Да или Не; 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.   4. Какво се случва след като отговорите на всички въпроси?   1. Смяна на костюм / фон на сцената; 2. Аби посочва броя на правилните и грешни отговори и дава оценка.   5. Изследване на кодовете в играта/Актуализиране на стари знания /индивидуална и фронтална дейност/  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.        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.  Добавяне на костюм в Snap! Може би преработен.  1. Разделяне на групата на отбори от 2 или 3 ученика.  2. Публикуване на темата за тестовите въпроси. Например - Използване на променливи; Цикли; Движение, Сензори, аритметични и логически операции.  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.  4. Създаване на кода. На студентите 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!: <https://snap.berkeley.edu/project?user=ddureva&project=test2>  Цялата дейност в Scratch:   * 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!:   <https://snap.berkeley.edu/snap/snap.html#present:Username=spelac&ProjectName=C4G_20_test_en_tmp>   * 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, * кодиране на множество обекти, * сензорни блокове, * цикли (forever, repeat until), * синхронизиране на действията чрез съобщения, * използване на произволни числа |
| **Резултати от обучението** | Общи цели на обучението:   * Движение на обект базирано на събития, * Засичане (разпознаване) на един цвят, * Употреба на булеви стойности в логически изрази, * Определяне, диференциране, динамично проверяване и реагиране на две различни състояния на играта.   Специфични цели на обучението, ориентирани към алгоритмичното мислене:   * Ученикът осъществява движение на обект с клавишите със стрелки, използвайки събития и отчита ограниченията, * Ученикът използва сензорен цветен блок, за да получи булева стойност за четене на едноцветни сензори, * Ученикът осъзнава, че състоянието на обекта може да бъде изразено с цвета на обекта до който се докосва, * Ученикът прави разлика между две различни състояния и знае как да ги изрази с логически изрази, * Ученикът осъзнава, че позицията на обекта се променя динамично и използва цикъл forever за многократна проверка на текущото състояние, * 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.  **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.  **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  Самостоятелна работа / Работа по двоики/Работа в групи |
| **Ход на урока** | (Мотивация-Въведение, Прилагане, Осмисляне и Оценка)  Играчът трябва да събират произволно разположени звезди, докато е преследван от червен призрак.  Ако той и призракът се сблъскат, играта свършва, ако пък успее да съберете 20-те звезди, преди това играчът печели.  [Стъпка 1]  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.    [Стъпка 6]  Сега трябва да програмирате кога ще спечелите играта.  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.  Ако броячът стигне до 20, PACMAN печели и играта свършва. |
| **Инструменти и ресурси за учителя** | * Цялата дейност в Snap!:   <https://snap.berkeley.edu/project?user=zapusek&project=pacman_clone>   * Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena. * 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>   * Инструкции за ученика (C4G21\_InstructionsForStudent\_BG.docx) |

Литература

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)

Lajovic, S. (2011). Scratch. *Nauči se programirati in postani računalniški maček*. Ljubljana: Pasadena.

Rugelj, J. (2019). Game design based learning of programming.

Vorderman, C. (2017). *Računalniško programiranje za otroke*. Ljubljana: MK.