**Сценарий 21 – Опростена игра PACMAN**

Задача: Програмирайте игра на опростен Pacman. Главният герой (pacman) се движи из лабиринта и събира звезди. В началото има една звезда на произволната позиция, когато pacman я събира, нова звезда се появява на ново произволно място вътре в лабиринта. Пакман е преследван от призрак, който се движи на случаен принцип. Играта приключва, когато играчът събере 20 звезди или ако Пакман докосне призрака.

1. Отворете файла:

<https://snap.berkeley.edu/project?user=zapusek&project=pacman_template>

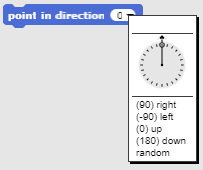
В него ще откриете изображение на pacman, червения призрак, звезда и фон на сцена, която представя лабиринт:

|  |  |
| --- | --- |
|  | |
|  |  |

1. Първо трябва да програмираме движението на pacman. Играчът може да контролира движението си със стрелки. Трябва да имаме предвид, че pacman не може да се движи през стените.

Ще програмираме движението чрез използване на няколко блока  . Ще дефинираме подходящи клавиши.

На всяка стъпка ще трябва да завъртим pacman в съответната посока и след това да извършим ход. Можем да завъртим обект с помощта , и избор на посока, използвайки падащо меню. Не е необходимо да въвеждаме точния ъгъл на завъртане, защото Snap! използва имена на посоките.



Пакман не може да се движи произволно, защото е ограничен от стени. Стените са боядисани в черно, а разрешените пътеки са боядисани в синьо. Това може да се използва за осъществяване на движение в лабиринта. Ако pacman е върху син цвят (пътека), той може да се движи, но ако докосне черния цвят, трябва да спре. Всяка стъпка трябва да се комбинира с това условие.

Кодирайте движението върху синята зона на лабиринта, като използвате тези блокове:

,, ,  and .

Ако разберем, че той докосва черния цвят, знаем, че е докоснал стената. В този случай трябва да го преместим обратно в обратна посока, за да не се забие на стената.

Положителна стойност в блока  ще премести обекта напред, а отрицателна (поставен знак -) ще премести обекта назад със 5 стъпки.

Добавете друг блок if, за да проверите дали pacman докосва черния цвят. В такъв случай го преместваме 5 стъпки в обратна посока.

1. След това искаме да кодираме поведението на звездата. В опростената игра на pacman в момента има само една звезда. Когато играчът я събере, на произволно място се появява нова звезда. За всяка звезда играчът получава по една точка, когато постигне има общо 20 точки, играта приключва.

В играта ще има общо 20 звезди. Ще звездите ще направим като на оригиналната звезда.

**Използваме клонинги в ситуации, когато искаме да използваме няколко екземпляра на оригиналния обект.**

Нов клонинг се създава с блока . От падащото меню избираме обекта, който искаме да клонираме.

Сега трябва да щракнем върху звезда и да кодираме какво се случва, когато се създаде клонингът от оригиналната звезда. Това се прави с помощта на блока  .

Когато нов екземпляр на звездата започва като клонинг, той трябва да се движи в произволна позиция на екрана. Височината и ширината на фона по подразбиране в Snap! е от -140 до 140 пиксела. Ако присвоим произволна стойност между тези две числа за х и у позиция на клонинг, постигаме произволното разположение на звездата. Използвайте комбинация от блокове:  и 

Следващият проблем, който възниква, е когато компютърът избира такива числа от интервала, че звездата е поставена на стена на лабиринта и pacman не може да стигне до нея.

Знаем, че стената е представена с черен цвят, можем да използваме този факт, за да проверим дали на стената е поставена звезда или не. Ако това е вярно, можем да направим друг клонинг и да изтрием този. Това ще се повтаря, докато клонингът не бъде поставен на законно място. Кодирайте това, като използвате следните блокове:

, ,  and .

Проверката дали pacman докосва звездата може да бъде програмирана в pacman или в клониращия код. Тъй като искаме да представим идеята за изпращане на съобщенията, ще го направим в клонинг.

Играчът може свободно да движи pacman из лабиринта, така че трябва да проверим дали той е докоснал клонинга завинаги. За да открием дали обектът е докоснал друг обект, можем да използваме блока: и от падащото меню да изберем докосвания обект.

Когато pacman събере клонинга, играчът ще получи една точка. Нека да приложим това, като зададем променлива на брояч на точки в pacman. Когато клонингът и pacman се сблъскат, клонингът ще изпрати съобщение до pacman, така че той ще знае, че трябва да увеличи брояча на точките.

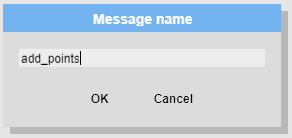
Кликнете върху pacman и създайте нова променлива точки за отчитане на точки. Задайте го на нула, тъй като играчът няма точки в началото на играта.

В Snap! обектите могат да си изпращат съобщения, за да ги уведомят, че нещо се е случило. Това се прави с блока: . Когато съобщението се изпрати, всички обекти на сцената ще го получат. В кода на всеки обект можем да опишем какъв ще бъде неговият отговор. Ако не програмираме нищо, нищо няма да се случи.

Създаваме ново съобщение, като щракваме върху малката стрелка вътре в полето за въвеждане на блока и избираме „ново“.



След това трябва да посочим име на съобщението. Искаме да изберем мнемонично име, така че значението на съобщението ще бъде разпознато от името му.



Попълнете кода на звездния клонинг по начин, който непрекъснато ще проверява дали клонингът се е сблъскал с pacman. Когато това се случи, трябва да изпрати съобщение add\_points. След изпращане на съобщение клонингът трябва да изпрати команда, за да създаде нов клонинг и да се изтрие.

За да програмирате тази функционалност, използвайте тези блокове:

, ,  и .

Нека да видим как можем да реагираме на полученото съобщение в другия обект. В групата „Контрол“ имаме блок за събития  който започва изпълнението си при получаване на определено съобщение. Можем да изберем конкретно съобщение, като използваме падащо меню.

В pacman кода програмирайте увеличаването на брояча на точки, когато съобщението се получи, като използвате тези блокове:  и .

Тук също можем да кодираме условието за крайна игра. Ако увеличим точките с 1 и общият резултат е равен на 20, играта приключва. Кодирайте тази функционалност, като използвате блоковете:

, ,  and .

1. Сега трябва да програмираме призрака. Призракът трябва да се движи произволно из лабиринта, не трябва да излиза извън стената, но трябва да отскача и да продължава в произволна посока. Ако докосне pacman, играта е приключила.

Нека помислим как можем да накараме призрак да се движи в произволна посока след докосване настена. Той може да се движи в следните посоки: наляво, надясно, нагоре и надолу.

Използвайки блока  можем да завъртим обекта. Можем да използваме следните посоки, чрез въвеждане на ъгъла на завъртане:

* 0 градуса = UP (Нагоре),
* 180 градуса = DOWN (Надолу),
* 90 градуса = RIGHT (дясно),
* 270 градуса = LEFT (Ляво)

Искаме да намерим начин, при който посоката на завъртане да бъде произволно зададена на една от тези стойности: 0, 90, 180 или 270. Проблемът е, че в Snap! можем да получим произволна стойност от интервал, а не от списък със стойности. Ако изучим тези числа, можем да забележим, че всички са кратни на числото 90, защото: 0 \* 90 = 0, 1 \* 90 = 90, 2 \* 90 = 180 и 3 \* 90 = 270. Можем да получим произволно число от интервал 0 до 3 с използване на блока , ако умножим това произволно избрано число с 90, ще получим едно от тези числа: 0, 90, 180, 270. Точно това искаме!

Движението на призрака и тестването дали той е докоснал стената трябва да се извършва по време на играта, докато играчът достигне 20 точки или призракът не докосне pacman. Вече програмирахме първата опция, която завършва играта, сега нека се погрижим за втората.

Изходното условие за преместване на призракa е, когато призракът се сблъска с pacman. Ако използваме цикъла **repeat until**, можем да зададем условието, когато той докосва pacman. Докато това не е вярно, призракът ще се движи наоколо, когато той удари pacman, цикълът ще спре и блоковете, поставени след цикъла, ще бъдат изпълнени.

Призрачното движение е съвсем просто. Във всяка итерация на цикъла той трябва да се движи с 1 стъпка в дадена посока.

Можем да проверим дали призракът е докоснал стената с блока за докосване на цвят. Стените са черни и подобни, както направихме с pacman, можем многократно да проверяваме дали призракът може би докосва черния цвят. Ако това се случи, трябва да го преместим с 1 стъпка назад (за да не заседне на стената) и да сменим посоката на случаен принцип, както е описано по-горе.

Цикълът ще спре, когато състоянието в главата му ще бъде вярно. Това ще се случи, когато призрак се сблъска с pacman. Трябва да предоставим обратна връзка и да прекратим играта.

Кодирайте тази функционалност, като използвате тези блокове::

,,,,,,  and .