**Урок №7. Циклы. Генерация элементов. Коллизия объектов**

**Вводный блок (10 минут)**

**Цели и задачи урока:**

1. Изучить синтаксис работы с циклами.
2. Научиться обрабатывать массивы используя циклы.
3. Сгенерировать игровой уровень с использованием циклов.
4. Изучить понятие «Коллизия» и применить его на практике.

На занятии мы рассмотрим синтаксис создания циклов и попробуем сгенерировать несколько объектов на холсте, помимо этого зададим для каждого объекта коллизию.

Сперва разберемся с тем, что такое цикл.

**Познание нового (20 минут)**

Цикл - повторение одних и тех же команд с разными входными данными. Это может быть элемент массива, может быть какое-либо случайное значение или событие, которое происходит, при этом структура команд остается неизменной.

**Вопросы для учеников**

1. Как вы думаете, используете ли вы циклы в повседневной жизни?
2. Можно ли каждый день представить как один обход цикла?

Предположим, у нас есть массив

const array = [100, 200, 300, 400, 500];

И нам нужно обойти все эти элементы и вывести их на экран.

Для этого можно воспользоваться двумя разными способами:

**Способ №1. По итератору**

В данном случае мы создаем переменную, в которую попеременно будут записываться все числа из массива (сначала 100, потом 200 и тд)

const array = [100, 200, 300, 400, 500];

for (let el of array) {

context.fillText(el)

}

**Способ №2. По счетчику**

В данном случае создается счетчик по которому происходит перебор элементов. В этом случае в цикле for через точку с запятой записывается три элемента - начальное значение счетчика; условие окончания работы цикла; изменение значения счетчика.

const array = [100, 200, 300, 400, 500];

for (let i = 0; i < array.length; i++) {

context.fillText(array[i])

}

**Некомпьютерная активность (15 минут)**

Игра «Цепочка»

Ход игры: ученики становятся в круг и передают друг другу какой-либо предмет (например, игрушку или мячик). Первый ученик называет условие цикла (например: «Если у меня в руках игрушка, то я передаю её следующему игроку»). Задача учеников — продолжить цепочку условий, чтобы каждый следующий игрок повторяет действия предыдущего до тех пор, пока предмет не достигнет первого участника.

Например:

* Первый участник говорит: «Если у меня в руках мяч, я бросаю его второму участнику».
* Второй участник продолжает: «Я ловлю мяч и бросаю третьему участнику».
* Третий участник добавляет: «Когда мяч попадает ко мне, я отбиваю его четвертому участнику».
* Четвертый участник говорит: «Я возвращаю мяч первому участнику».
* Пятый участник завершает: «И игра начинается заново».

Задача игроков — придумать интересное и логичное продолжение цепочки условий. Побеждает тот, кто сможет наиболее увлекательно продолжить игру.

**Работа над проектом (40 минут)**

Тема: Создание нового проекта и написание программы, в которой создается лабиринт с коллизией стен.

**Задания:**

1. Задание на цикл по счетчику. Создать массив из 5 элементов и задать в него любые строки. Вывести их на холст в столбец используя функцию fillText (7.1)
2. Задание на цикл по итератору. Создать массив из 3 элементов, содержащих в себе строку в формате "rgb(число, число, число)". Вывести 3 квадрата с заданными цветами (7.2)

**Генерируем уровень**

Генерацию уровня мы будем производить с помощью способа №2 (по счетчику).

Одним из вариантов генерации уровня является использование массива строк, где каждый отдельный элемент будет хранить в себе цифру в зависимости от типа создаваемого элемента.

Например, 0 - пустое место, 1 - осязаемый блок, 2 - монета, 3 - старт и 4 - финиш.

Предположим, создается уровень 10 на 10 вот в таком вот формате

| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 1 | 4 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 2 | 0 | 1 |
| 1 | 0 | 1 | 2 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Эту таблицу нам нужно перенести в программный код и задать все необходимые составляющие.

Для этого создается массив из строк

const level = [

"1111111111",

"1314001001",

"1011101201",

"1012101101",

"1010000001",

"1011111101",

"1000000101",

"1011110101",

"1012000001",

"1111111111"

]

const size = 70;

В данном случае size - размер одной клетки.

В функции move создаются стены уровня, где в цикле for перебирается каждая строка из массива level, а поскольку в каждой строке нужно перебрать еще и все символы, то используются вложенный цикл for, где зная номер строки и столбца можно обращаться к каждой ячейки.

Во вложенном цикле идет проверка значения в определенной ячейки, если 1 - стена, если 2 - монета, 3 - старт и 4 - финиш

function move() {

for (let i = 0; i < level.length; i++) {

for (let j = 0; j < level[i].length; j++) {

if (level[i][j] == 1) {

context.fillRect(j \* size, i \* size, size, size);

}

}

}

requestAnimationFrame(move);

}

move();

**Задание**

1. Добавить остальные элементы уровня в соответствии с их номерами (7.3)
2. Добавить стартовые координаты и отобразить персонажа на уровне в виде квадрата (7.4)

**Настраиваем коллизию**

**Обсуждение**

1. Как вы думаете, что такое коллизия? Где она используется в играх?

Коллизия - взаимодействие между объектами, а если точнее, то их столкновение (еще можно встретить понятие пересечение) и его результат.

Зададим движение персонажа.

Для этого нам необходимо проверять все блоки, имеющие код 1 и являющиеся стенами.

Данный код записывается после отображения всех элементов (можно выдать, если времени не будет хватать)

for (let i = 0; i < level.length; i++) {

for (let j = 0; j < level[i].length; j++) {

if (level[i][j] == 1) {

if (start\_x > j \* size

&& start\_x < j \* size + size

&& start\_y > i \* size

&& start\_y < i \* size + size

) {

start\_x -= dx;

start\_y -= dy;

}

if (start\_x + size / 2 > j \* size

&& start\_x + size / 2 < j \* size + size

&& start\_y > i \* size

&& start\_y < i \* size + size

) {

start\_x -= dx;

start\_y -= dy;

}

if (start\_x + size / 2 > j \* size

&& start\_x + size / 2 < j \* size + size

&& start\_y + size / 2 > i \* size

&& start\_y + size / 2 < i \* size + size

) {

start\_x -= dx;

start\_y -= dy;

}

if (start\_x > j \* size

&& start\_x < j \* size + size

&& start\_y + size / 2 > i \* size

&& start\_y + size / 2 < i \* size + size

) {

start\_x -= dx;

start\_y -= dy;

}

}

}

}

**Рефлексия (5 минут):**

"Что нового вы узнали сегодня?"

"Что такое циклы? Какие типы циклов бывают? В какой момент останавливается работа цикла?"

“Что такое коллизия объекта?”