Санкт-Петербургский политехнический университет Петра Великого

Институт прикладной математики и механики

**Кафедра «Теоретическая механика»**

**КУРСОВОЙ ПРОЕКТ**

**Реализация и визуализация игры “Saper”**

по дисциплине «Математическое моделирование»

Выполнили

студенты гр.13632/1 Черницына А.А

Аминов А.А

Руководитель

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2019 г.

 Санкт-Петербург

 2019

**СОДЕРЖАНИЕ**

Введение. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

Постановка задачи. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

1. Реализация. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .4

1.1. Описание кода HTML и JavaScript. . . . . . . . . . . . . . . . . . . . . . . . . . ..5-14

Результаты. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15-17

Вывод и заключение. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .18-19

**Введение**

JavaScript – многофункциональный язык, который позволяет выполнять задачи различной направленности и сложности. Данная работа посвящена созданию игры "Saper" на подобии игры “Сапер” с заданными условиями и правилами её функционирования. Актуальность темы заключается в том, что в современном мире существует немало всевозможных игр, разработанных на базе различных языков программирования, и поэтому немаловажно понимать основные механизмы и принципы создания и визуализации подобных программ. Цель курсовой работы заключается в изучении основных принципов создания игры "Сапер". Объект исследования – процесс использования языков программирования для создания простейшей игры. Предмет исследования – программные инструменты языков JavaScript и HTML, которые применяются при разработке простейших игр.

**Постановка задачи**

Реализовать и визуализировать игру “Saper” со следующими правилами:

1.Плоское или объёмное игровое поле разделено на смежные ячейки (квадраты, шестиугольники, кубы и т. п.), некоторые из которых «заминированы»; количество «заминированных» ячеек известно. Целью игры является открытие всех ячеек, не содержащих мины.

2.Игрок открывает ячейки, стараясь не открыть ячейку с миной. Открыв ячейку с миной, он проигрывает.

3.Если под открытой ячейкой мины нет, то в ней появляется число, показывающее, сколько ячеек, соседствующих с только что открытой, «заминировано». Используя эти числа, игрок пытается рассчитать расположение мин. Если под соседними ячейками тоже нет мин, то открывается некоторая «не заминированная» область до ячеек, в которых есть цифры. «Заминированные» ячейки игрок может пометить, чтобы случайно не открыть их. Открыв все «не заминированные» ячейки, игрок выигрывает.

**Реализация**

Перечень функций, которые были использованы для написания программы:

* document.getElementById(id) - Возвращает ссылку на элемент по его идентификатору ([ID](https://developer.mozilla.org/en-US/docs/DOM/element.id)); идентификатор является строкой, которая может быть использована для идентификации элемента; она может быть определена при помощи атрибута id в HTML или из скрипта.
* setAttribute() - Добавляет новый атрибут или изменяет значение существующего атрибута у выбранного элемента.
* appendChild()- вставляет DOM-узел в некоторый элемент
* Draw - функция прорисовки
* Document.images - возвращает изображение

<HTML>

<HEAD>

<TITLE>Сапер</TITLE><style>body{scrollbar-base-color: #4444FF}</style></HEAD>

 **Меняем цвет фона**

<BODY BGCOLOR="#333333" onContextMenu="return false;">

<CENTER>

<BR>

**Создаем таблицу**

<TABLE id="GameField" BGCOLOR=#CCCCAA BORDER=1></TABLE>

<BR>

**Создаем функции для визуализации победы и поражения**

<div id="GameOverResult" class="result">Game Over</div>

<div id="WinResult" class="result">WIN</div>

<FORM>

<BUTTON STYLE="{font-size:24px;background-color:#002040;border-width:0;color:red" onClick="NewGameForSer()">НОВАЯ ИГРА</BUTTON>

</FORM>

</CENTER>

<style>

.result {

 display: none;

 position: 50% 50%;

 position: fixed;

 top: 50%;

 left: 0%;

 font-size: 300;

 z-index: 1;

 background-color: pink;

 margin-top: -150px;

 width: 100%;

 opacity: 0.7;

}

</style>

<SCRIPT>

var massivA = new Array (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0);

**Дополнительный массив для поиска пути**

var massivB = new Array (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0);

var Nomer = 0;

var inGame = false;

function createGameField()

{

 var table = document.getElementById("GameField");

 for (var x = 0; x < 9; ++x)

 {

 var tr = document.createElement("tr");

 for (var y = 0; y < 9; ++y)

 {

 var img = document.createElement("img");

 img.src = "Saper/POLE.jpg";

 var item = document.createElement("a");

 item.href = "#";

 item.name = x \* 9 + y + 1;

 item.appendChild(img);

 var args = "(" + y + ", " + x + ")";

 item.setAttribute("onClick", "Move" + args);

 item.setAttribute("onContextMenu", "setFlag" + args);

 var td = document.createElement("td");

 td.appendChild(item);

 tr.appendChild(td);

 }

 table.appendChild(tr);

 }

}

createGameField();

**Вызов новой игры**

NewGameForSer();

function setFlag(x, y)

{

 if (massivB[ 9 \* y + x]!=0 || !inGame) return;

 var flagImg = 'Saper/FLAG.jpg';

 var field = document.images[ 9 \* y + x ];

 if (field.src.match(flagImg))

 field.src = 'Saper/POLE.jpg';

 else

 field.src = 'Saper/FLAG.jpg';

}

**Задаем функцию прорисовки клетки**

function Draw(x1, y1)

{

 if (massivA[ 9 \* y1 + x1]>19)

 {

 }

 else

 document.images[ 9 \* y1 + x1 ].src='Saper/'+massivA[ 9 \* y1 + x1]+'.jpg';

}

**Функция для новой игры**

function NewGameForSer()

{

 inGame = true;

 document.getElementById("WinResult").style="";

 document.getElementById("GameOverResult").style=""

 Nomer = 0;

 **Очистка массивов**

 for (var x = -2; x<= 11; x++)

 for (var y = -2; y<= 11; y++)

 {

 massivA[ 9 \* y + x ] = 0;

 massivB[ 9 \* y + x ] = 0;

 }

 for (var x = 0; x<= 8; x++)

 for (var y = 0; y<= 8; y++)

 document.images[ 9 \* y + x ].src = 'Saper/POLE.jpg';

 **Расстановка мин на поле**

 var i = 1;

 while (i<14)

 {

 x = Math.floor(Math.random()\*9);

 y = Math.floor(Math.random()\*9);

 if (massivA[ 9 \* y + x ]==20) continue;

 massivA[ 9 \* y + x ] = 20;

 i++;

 }

  **Расстановка цифр вокруг мин**

 for (x = 0; x<=8; x++)

 for (y = 0; y<=8; y++)

 if (massivA[ 9 \* y + x ]>19)

 {

 if ((x - 1 >= 0) && ( x - 1 <=8 ))

 massivA[ 9 \* y + x -1 ]++;

 if ((x - 1 >= 0) && ( x - 1 <=8 ) && (y - 1 >= 0) && (y - 1 <= 8))

 massivA[ 9 \* (y - 1) + x -1 ]++;

 if ((y - 1 >= 0) && ( y - 1 <=8 ))

 massivA[ 9 \* (y - 1) + x ]++;

 if ((x + 1 >= 0) && ( x + 1 <=8 ) && (y - 1 >= 0) && (y - 1 <= 8))

 massivA[ 9 \* (y - 1) + x + 1 ]++;

 if ((x + 1 >= 0) && (x + 1 <=8 ))

 massivA[ 9 \* y + x + 1 ]++;

 if ((x + 1 >= 0) && ( x + 1 <=8 ) && (y + 1 >= 0) && (y + 1 <= 8))

 massivA[ 9 \* (y + 1) + x + 1 ]++;

 if ((y + 1 >= 0) && (y + 1 <=8 ))

 massivA[ 9 \* (y + 1) + x ]++;

 if ((x - 1 >= 0) && ( x - 1 <=8 ) && (y + 1 >= 0) && (y + 1 <= 8))

 massivA[ 9 \* (y + 1) + x - 1 ]++;

 }

}

**Выбор клетки**

function Move(x1,y1)

{

 if (!inGame) return;

 if (massivA[ 9 \* y1 + x1]>19)

 {

 GameOver();

 }

 else

 {

 if (massivB[ 9 \* y1 + x1 ] == 0 )

 {

 massivB[ 9 \* y1 + x1 ] = 1;

 Nomer++;

 Draw(x1, y1);

 if (Nomer==68) GamesWin();

 if (massivA[ 9 \* y1 + x1 ] == 0 ) FindZero(x1, y1);

 }

 }

}

 **Функция для поиска пустоты**

function FindZero(x1, y1)

{

 for (var Colonka = x1 - 1; Colonka <= x1 + 1; Colonka++)

 for (var Stolbik = y1 - 1; Stolbik <= y1 + 1; Stolbik++)

 {

 if (Nomer==68) GamesWin();

 if ((Colonka >= 0) && (Colonka <9) && (Stolbik >= 0)

 && (Stolbik <9) && (massivB[ 9 \* Stolbik + Colonka ]==0))

 {

 if ((Colonka == x1-1) && (Stolbik == y1-1) && (massivA[ 9 \* (y1 - 1) + x1 -1 ] == 0)) continue;

 if ((Colonka == x1+1) && (Stolbik == y1-1) && (massivA[ 9 \* (y1 - 1) + x1 + 1] == 0)) continue;

 if ((Colonka == x1+1) && (Stolbik == y1+1) && (massivA[ 9 \* (y1 + 1) + x1 + 1 ] == 0)) continue;

 if ((Colonka == x1-1) && (Stolbik == y1+1) && (massivA[ 9 \* (y1 + 1) + x1 - 1 ] == 0)) continue;

 if (massivA[ 9 \* Stolbik + Colonka ]==0)

 {

 Draw(Colonka, Stolbik);

 Nomer++;

 massivB[ 9 \* Stolbik + Colonka ] = 1;

 FindZero(Colonka, Stolbik);

 }

 else

 {

 Nomer++;

 massivB[ 9 \* Stolbik + Colonka ] = 1;

 Draw(Colonka, Stolbik);

 }

 }

 }

}

**Обработка выигрыша**

function GamesWin()

{

 for (var i=0; i<9\*9; i++)

 if (massivA[i]>19)

 document.images[i].src = 'Saper/FLAG.jpg';

 document.getElementById("WinResult").style.display="initial";

 inGame = false;

}

function GameOver()

{

 for (var i=0; i<9\*9; ++i)

 if (massivA[i]>19)

 document.images[i].src='Saper/MINE.jpg'

 document.getElementById("GameOverResult").style.display="initial";

 inGame = false;

}

</SCRIPT>

</BODY>

</HTML>

**Результаты**

На языке программирования Javascript был составлен структурно простой код, позволяющий пользователю сыграть в аналог игры “Сапер” – “Saper”. Управление производится мышью.

На рисунке, представленном ниже, можно ознакомиться с игровым процессом и интерфейсом игры.



Цифрами показано количество мин по соседству с клеткой, флаг - отметка о предполагаемой мине



Надпись при проигрыше, снизу - кнопка с перезапуском игры



Победа

**Вывод**

Таким образом, реализация и визуализация аналога игры “Сапер” – “Saper” оказались успешными, нам удалось выполнить поставленные задачи и также освоить использование новых функций. В итоге мы получили новый и полезный опыт в работе над подобными заданиями и составлением курсовых работ, который будет важен в дальнейшем.

**Заключение**

Безусловно, есть возможность добавления новых механик к игре в качестве последующей доработки программы и улучшения как динамической, так и визуальной составляющих игры. В заключении хотелось бы сказать, что JavaScript – многофункциональный язык, который позволяет выполнять задачи различной направленности и сложности.

**ФОРМА ЗАДАНИЯ НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА**

Санкт-Петербургский политехнический университет Петра Великого

**ЗАДАНИЕ**

**НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА**

студенту группы 13632/1 Черницыной А.А.

студенту группы 13632/1 Аминову А.А

***1. Тема проекта:*** Реализация и визуализация игры «Saper»

***2. Срок сдачи студентом законченного проекта***

***3. Исходные данные к проекту:*** курс лекций по математическому моделированию за первый и второй семестры

***4. Содержание пояснительной записки*** (перечень подлежащих разработке вопросов): введение, основная часть (раскрывается структура основной части), заключение, список использованных источников, приложения.

Примерный объём пояснительной записки - 9 страниц печатного текста.

***5. Перечень графического материала*** (с указанием обязательных чертежей и плакатов): отсутствует

***6. Консультанты***

***7. Дата получения задания:*** «\_\_\_».\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 *(подпись) (инициалы, фамилия)*

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

  *(подпись студента) (инициалы, фамилия)*

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

  *(подпись студента) (инициалы, фамилия)*

\_\_\_\_\_\_\_\_\_\_\_\_ *(дата)*