СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Battle City (с англ. — «Город битвы») — компьютерная игра для игровых приставок Famicom и Game Boy. В России и странах СНГ выпускалась на пиратских картриджах как в оригинальном виде, так и в модификации Tank 1990, и известна под неофициальным названием «Та́нчики». Её предшественником была аркадная игра Tank Battalion, выпущенная фирмой Namco в 1980 году.

Это игра сразу полюбилась многими , даже до сих пор многими вспоминается со слезами счастья на глазах.

Полигон действий виден сверху. Игрок должен, управляя своим танком, уничтожить все вражеские танки на уровне, которые постепенно появляются вверху игрового поля. Враги пытаются уничтожить штаб игрока (внизу игрового поля в виде орла) и его танк. На каждом уровне нужно уничтожить двадцать единиц бронетехники противника разных видов. Если противник сможет разрушить штаб или лишит игрока всех жизней — игра окончена [1]

Задача данного курсового проекта состоит в том, чтобы создать игру «Battle City», используя язык программирования JavaScript, HTML, CSS.

Игра должна содержать:

– HTML-страница презентация игры, размещенная на github pages;
– верстка для мобильных устройств;
– css в отдельном файле;
– репозиторий на github;
– размещение в сети (heroku или любой другой сервер);

Функционал игры:

– начать новую игру;
– таймер затраченного времени;
– проверка окончания игры;
– информация о результате игры.

1 АНАЛИЗ ТЕХНИЧЕСКОГО ЗАДАНИЯ

Цель данного курсового проекта создать игру «Battle City», используя язык программирования JavaScript, HTML, CSS.

JavaScript – мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией языка ECMAScript (стандарт ECMA-262). JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам. Основные архитектурные черты: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса. На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на Java, но при этом лёгким для использования непрограммистами. Языком JavaScript не владеет какая-либо компания или организация, что отличает его от ряда языков программирования, используемых в веб-разработке. Название «JavaScript» является зарегистрированным товарным знаком компании Oracle Corporation.

JavaScript является объектно-ориентированным языком, но используемое в языке прототипирование обуславливает отличия в работе с объектами по сравнению с традиционными класс-ориентированными языками. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам – функции как объекты первого класса, объекты как списки, карринг, анонимные функции, замыкания – что придаёт языку дополнительную гибкость. Несмотря на схожий с Си синтаксис, JavaScript по сравнению с языком Си имеет коренные отличия: объекты с возможностью интроспекции, функции как объекты первого класса, автоматическое приведение типов, автоматическая сборка мусора, анонимные функции. В языке отсутствуют такие полезные вещи, как: стандартная библиотека: в частности, отсутствует интерфейс программирования приложений по работе с файловой системой, управлению потоками ввода-вывода, базовых типов для бинарных данных; стандартные интерфейсы к веб-серверам и базам данных; система управления пакетами, которая бы отслеживала зависимости и автоматически устанавливала их. [2]

HTML (от англ. HyperText Markup Language – «язык гипертекстовой разметки») – стандартизированный язык разметки документов во Всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML (или XHTML). Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства. Язык HTML до 5-й

версии определялся как приложение SGML (стандартного обобщённого языка разметки по стандарту ISO8879). Спецификации HTML5 формулируются в терминах DOM (объектной модели документа). Во всемирной паутине HTML-страницы, как правило, передаются браузерам от сервера по протоколам HTTP или HTTPS, в виде простого текста или с использованием шифрования. [3]

Данная игра носит развлекательный характер, тренирует логическое мышление, способность выстраивать стратегии в определенных условиях. Многие в детстве играли в данную игру с помощью листочка и тратили время и бумагу на прорисовку кораблей, теперь есть возможность вспомнить про эту интересную игру, но уже используя компьютер с его искусственным интеллектом.

Игра будет реализована в виде html-страницы. Для удобства работы в программе будет организован интерфейс в виде меню, так как именно эта форма интерфейса позволяет одним нажатием клавиши, не вводя никаких команд, выполнить ту или иную процедуру. В меню можно будет найти разделы «Играть», «Примеры», «Правила», «Об авторе».

Для ознакомления с краткой справочной информацией по правилам игры в программе будет реализован раздел «Правила». Естественно, если игрок забыл правила игры или вообще их не знает, то ему нужно будет прочитать правила, чтобы понять, как играть.

Раздел «Об авторе» будет включать в себя информацию об авторе разработанной программы. Этот раздел можно будет найти в меню, а перейти можно будет, нажав на название раздела.

Раздел «Играть» будет содержать саму игру «Battle City», то есть для того, чтобы начать играть, игрок должен будет кликнуть на раздел «Играть», чтобы перейти на html-страницу с игрой. Игра автоматически запускается, подтверждением этого будет служить запущенный таймер.

Игра будет представлять собой поле, содержащее игровое поле десять на десять и все возможные корабли, которые могут быть в этом поле.

Когда игрок захочет сыграть сначала, то необходимо будет реализовать процедуру «Заново», которая позволит игроку начать игру заново. Если же он закончил играть, либо же захотел вернуться в главное меню, то необходимо будет добавить процедуру «Меню», которая вернёт игрока к главной странице.

Как и любой программный продукт, данная программа будет иметь информационную заставку, из которой можно будет узнать название игры.

2 ВЫБОР МЕТОДА РЕШЕНИЯ ЗАДАЧИ

Данный курсовой проект реализован с помощью Visual Studio Code – редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией.

Visual Studio Code основан на Electron – фреймворк, позволяющий с использованием Node.js разрабатывать настольные приложения, которые работают на движке Blink. Несмотря на то, что редактор основан на Electron, он не использует редактор Atom. Вместо него реализуется веб-редактор Monaco, разработанный для Visual Studio Online.

Visual Studio Code – это редактор исходного кода. Он поддерживает ряд языков программирования, подсветку синтаксиса, IntelliSense, рефакторинг, отладку, навигацию по коду, поддержку Git и другие возможности. Многие возможности Visual Studio Code не доступны через графический интерфейс, зачастую они используются через палитру команд или JSON файлы (например, пользовательские настройки). Палитра команд представляет собой подобие командной строки, которая вызывается сочетанием клавиш.

Visual Studio также позволяет заменять кодовую страницу при сохранении документа, символы перевода строки и язык программирования текущего документа.

С 2018 года появилось расширение Python для Visual Studio Code с открытым исходным кодом. Оно предоставляет разработчикам широкие возможности для редактирования, отладки и тестирования кода.

Visual Studio Code имеет поддержку плагинов, доступных через Visual Studio Marketplace. Они могут включать в себя дополнения к редактору, поддержку дополнительных языков программирования, статические анализаторы кода. [4]

Цель данного курсового проекта создать игру «Battle City», используя язык программирования JavaScript, HTML, CSS. JavaScript является объектно-ориентированным языком, но используемое в языке прототипирование обуславливает отличия в работе с объектами по сравнению с традиционными класс-ориентированными языками. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам – функции как объекты пер-

вого класса, объекты как списки, карринг, анонимные функции, замыкания – что придаёт языку дополнительную гибкость. [2]

Язык HTML был разработан британским учёным Тимом Бернерсом-Ли приблизительно в 1986–1991 годах в стенах ЦЕРНа в Женеве в Швейцарии. HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. HTML успешно справлялся с проблемой сложности SGML путём определения небольшого набора структурных и семантических элементов – дескрипторов. Дескрипторы также часто называют «тегами». С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Помимо упрощения структуры документа, в HTML внесена поддержка гипертекста. Мультимедийные возможности были добавлены позже.

Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен был без стилистических и структурных искажений воспроизводиться на оборудовании с различной технической оснащённостью (цветной экран современного компьютера, монохромный экран органайзера, ограниченный по размерам экран мобильного телефона или устройства и программы голосового воспроизведения текстов). Однако современное применение HTML очень далеко от его изначальной задачи. Например, тег <table> предназначен для создания в документах таблиц, но иногда используется и для оформления размещения элементов на странице. С течением времени основная идея платформонезависимости языка HTML была принесена в жертву современным потребностям в мультимедийном и графическом оформлении. [3]

CSS (/siːɛsɛs/ англ. Cascading Style Sheets – каскадные таблицы стилей) – формальный язык описания внешнего вида документа, написанного с использованием языка разметки. Преимущественно используется как средство описания, оформления внешнего вида веб-страниц, написанных с помощью языков разметки HTML и XHTML, но может также применяться к любым XML-документам, например, к SVG или XUL. До появления CSS оформление веб-страниц осуществлялось исключительно средствами HTML, непосредственно внутри содержимого документа. Однако с появлением CSS стало возможным принципиальное разделение содержания и представления документа. За счёт этого нововведения стало возможным лёгкое применение единого стиля оформления для массы схожих документов, а также быстрое изменение этого оформления. Преимущества: несколько дизайнов страницы для разных устройств просмотра; уменьшение времени загрузки страниц сайта за счет переноса правил представления данных в отдельный CSS-файл; простота последующего изменения дизайна; дополнительные возможности оформления. [5]

3 ПРОЕКТИРОВАНИЕ ФУНКЦИОНАЛЬНЫХ ЧАСТЕЙ

На основании выбранного метода решения можно выделить следующие примерные функциональные части:

- процедура открытия программы будет производится запуском html-файла;
- организация интерфейса в виде меню с такими разделами, как «Work», «About», «Blog», «Contact»;
- процедура открытия новой html-страницы с игрой «BattleCity»;
- Открытие по нажатию Battle City;
- Новая html-страница с интерфейсом «Player1» , «Constraction» ;
- Начало игры при нажатие на «Player1»;
- Создание игрового поля ;
- Создание вражеских танков ;
- Событие завершения игры в зависимости от исхода сражения;
- Переход на следующий уровень , при уничтожении всех вражеских танков ;
- Событие “конец игры” при уничтожении вражескими танками нашей базы , либо нашего танка .
- Событие “Статистика” при окончании игры.

4 РАЗРАБОТКА ПРОГРАММЫ

Игра «Battle City» была создана при помощи языка программирования JavaScript, HTML, CSS в редакторе исходного кода Visual Studio Code.

Основные этапы проектирования игры приведены на рисунке 4.1.

Рисунок 4.1 – Этапы проектирования

Для начала , всё поле в котором будет отображаться игра , расположено в

```
function Gamefield(sceneManager) {
  Rect.call(this);

  this._sceneManager = sceneManager;
  this._eventManager = this._sceneManager.getEventManager();

  this._x = Globals.UNIT_SIZE;
  this._y = Globals.TILE_SIZE;
  this._w = 13 * Globals.UNIT_SIZE;
  this._h = 13 * Globals.UNIT_SIZE;

  this._spriteContainer = new SpriteContainer(this._eventManager);
  this._painter = new Painter(this._spriteContainer);
  this._updater = new Updater(this._spriteContainer);
```

```
  var bounds = new Rect(this._x, this._y, this._w, this._h);
  new CollisionDetector(this._eventManager, bounds, this._spriteContainer);
}

Gamefield.subclass(Rect);

Gamefield.prototype.update = function () {
  this._updater.update();
};

Gamefield.prototype.draw = function (ctx) {
  ctx.fillStyle = "#808080";
  ctx.fillRect(0, 0, ctx.canvas.width, ctx.canvas.height);

  ctx.fillStyle = "black";
  ctx.fillRect(this._x, this._y, this._w, this._h);

  this._painter.draw(ctx);
};
```

Рисунок 4.2 – Игровое поле

Процесс игры производится многочисленными событиями , например таким как взрыв танка , или разрушение текстуры-спрайта , в итоге мы либо выигрываем игру , либо её выигрываем , ниже приведён пример того , как происходит добавление событий .

```
function EventManager() {
  this._subscribers = {};
}

EventManager.prototype.addSubscriber = function (subscriber, events) {
  for (var i in events) {
    if (!this._subscribers[events[i]]) {
      this._subscribers[events[i]] = [];
    }
    this._subscribers[events[i]].push(subscriber);
  }
};

EventManager.prototype.removeSubscriber = function (subscriber) {
  for (var i in this._subscribers) {
    for (var j in this._subscribers[i]) {
      if (this._subscribers[i][j] === subscriber) {
        this._subscribers[i].splice(j, 1);
      }
    }
  }
};

EventManager.prototype.removeAllSubscribers = function () {
```

```
  this._subscribers = {};
};

EventManager.prototype.fireEvent = function (event) {
  var subscribers = this._subscribers[event.name];
  for (var i in subscribers) {
    subscribers[i].notify(event);
  }
};
```

Рисунок 4.3 – Фрагмент кода рандомной расстановки кораблей

Фрагмент кода функциональной части «Таймер» приведён на рисунке 4.4:

```
SceneManager.prototype.timer = function () {
  var min, sec, ms, count, malt, salt, msalt;
  var stopwatch = {
    start: function () {

      ms = 0;
      sec = 0;
      min = 0;
      count = setInterval(function () {
        if (ms == 100) {
          ms = 0;
          if (sec == 60) {
            sec = 0;
            min++;
          }
          else {
            sec++;
          }
        }
        else {
          ms++;
        }

        malt = stopwatch.pad(min);
        salt = stopwatch.pad(sec);
        msalt = stopwatch.pad(ms);

        stopwatch.update(malt + ":" + salt + ":" + msalt);
      }, 10);
```

```
    },
    stop: function () {
      clearInterval(count);
    },
    reset: function () {
      stopwatch.stop();
      var temp = document.getElementById("timer");
      temp.firstChild.nodeValue = "00:00:00";
    },
    update: function (txt) {
      var temp = document.getElementById("timer");
      temp.firstChild.nodeValue = txt;
    },

    pad: function (time) {
      var temp;
      if (time < 10) {
        temp = "0" + time;
      }
      else {
        temp = time;
      }
      return temp;
    }
  }
  return stopwatch;
}
```

Рисунок 4.4 – Фрагмент кода функциональной части «Таймер»

В программе использовались такие структуры, как циклы, которые организовывались таким оператором, как for.

С помощью цикла for можно выполнять оператор или блока выписок повторно до тех пор, пока указанное выражение не будет оценки к false. Этот тип цикла полезен для перебора массивами и для других приложений, в которых известны заранее, сколько раз необходимо цикл повторных.

При использовании в качестве типа возвращаемого значения для метода, void указывает, что метод не возвращает значение.

Оператор if служит для того, чтобы выполнить какую-либо операцию в том случае, когда условие является верным. Условная конструкция всегда записывается в круглых скобках после оператора if.

Внутри фигурных скобок указывается тело условия. Если условие выполнится, то начнется выполнение всех команд, которые находятся между фигурными скобками.

Конструкция switch заменяет собой сразу несколько if. Она представляет собой более наглядный способ сравнить выражение сразу с несколькими вариантами. Переменная x проверяется на строгое равенство первому значению value1, затем второму value2 и так далее. Если соответствие установлено

– switch начинает выполняться от соответствующей директивы case и далее, до ближайшего break (или до конца switch). Если ни один case не совпал – выполняется (если есть) вариант default. При этом case называют вариантами switch.

Зачастую нам надо повторять одно и то же действие во многих частях программы. Например, красиво вывести сообщение необходимо при приветствии посетителя, при выходе посетителя с сайта, ещё где-нибудь. Чтобы не повторять один и тот же код во многих местах, придуманы функции (function). Функции являются основными «строительными блоками» программы. Примеры встроенных функций – это alert(message), prompt(message, default) и confirm(question). Но можно создавать и свои. Объявление функции происходит следующим образом: вначале идет ключевое слово function, после него имя функции, затем список параметров в скобках и тело функции – код, который выполняется при её вызове.

Функция может содержать локальные переменные, объявленные через var. Такие переменные видны только внутри функции. Блоки if/else, switch, for, while, do..while не влияют на область видимости переменных. При объявлении переменной в таких блоках, она всё равно будет видна во всей функции.

Функция может возвратить результат, который будет передан в вызвавший её код. Для возврата значения используется директива return**.** Она может находиться в любом месте функции. Как только до неё доходит управление – функция завершается и значение передается обратно. Директива return может также использоваться без значения, чтобы прекратить выполнение и выйти из функции.

Имя функции следует тем же правилам, что и имя переменной. Основное отличие – оно должно быть глаголом, т.к. функция – это действие. Как правило, используются глагольные префиксы, обозначающие общий характер действия, после которых следует уточнение. Функции, которые начинаются с "show" – что-то показывают. Функции, начинающиеся с "get" – получают, и т.п. Это очень удобно, поскольку взглянув на функцию – мы уже примерно представляем, что она делает, даже если функцию написал совсем другой человек, а в отдельных случаях – и какого вида значение она возвращает.

Собственно без чего бы не заработал этот курсовой , так это без прототипирования .

В JavaScript-е есть понятие объекта, но нет понятия класса.Весь JavaScript основан на объектах, поэтому почти все данные в JavaScript-е являются объектами, или могут быть использованы как объекты. А в качестве альтернативы классам есть возможность прототипирования — назначать объекту прототип со свойствами и методами «по умолчанию».

Работать с объектами в JavaScript очень просто — нужно всего лишь объявить объект и назначить ему свойства и методы. Т.е. если у какого-либо

объекта попытаться получить свойство или вызвать функцию, которой у объекта нет, то JavaScript интерпретатор, прежде чем сгенерировать ошибку, попытается найти это свойство/функцию в объекте-прототипе и, если оно будет найдено, будет использоваться свойство/функция из прототипа.

5 ТЕСТИРОВАНИЕ И АНАЛИЗ РЕЗУЛЬТАТОВ

5.1 Разработка тестовых примеров

Тест №1. Первый запуск

Запуск программы на экран осуществляется нажатием на картинку с названием “NAMCO”. Должно произойти открытие страницы в браузере с основной информацией и меню. Чтобы перейти к самой игре, нужно нажать на кнопку «1 players». В итоге должна открыться страница с игрой Battle City.

Тест №2. Начало игры. Ход игры

Чтобы начать играть в «Battle City», необходимо нажать на ячейку игрового поля с надписью “1 Players”.

Тест №3. Переход на следующий уровень

Протестируем переход на следующий уровень.

Тест №4.Таймер

Таймер должен исправно работать , и показывать затраченное время на уровень , и обнуляться при каждом новом уровне.

Тест №5. Счёт

При прохождении или поражении игроку будет выводиться его счёт , с убитыми врагами за которых будут начисляться очки.

Тест №6. Поражение

При уничтожении нашей базы или нашего танка , должно выводиться соответственная надпись “Game Over”.

5.2 Тестирование программы

Тест №1. Первый запуск

Запуск программы на экран осуществляется нажатием на картинку с названием “NAMCO”. Должно произойти открытие страницы в браузере с основной информацией и меню. Чтобы перейти к самой игре, нужно нажать на кнопку «1 players». В итоге должна открыться страница с игрой Battle City.

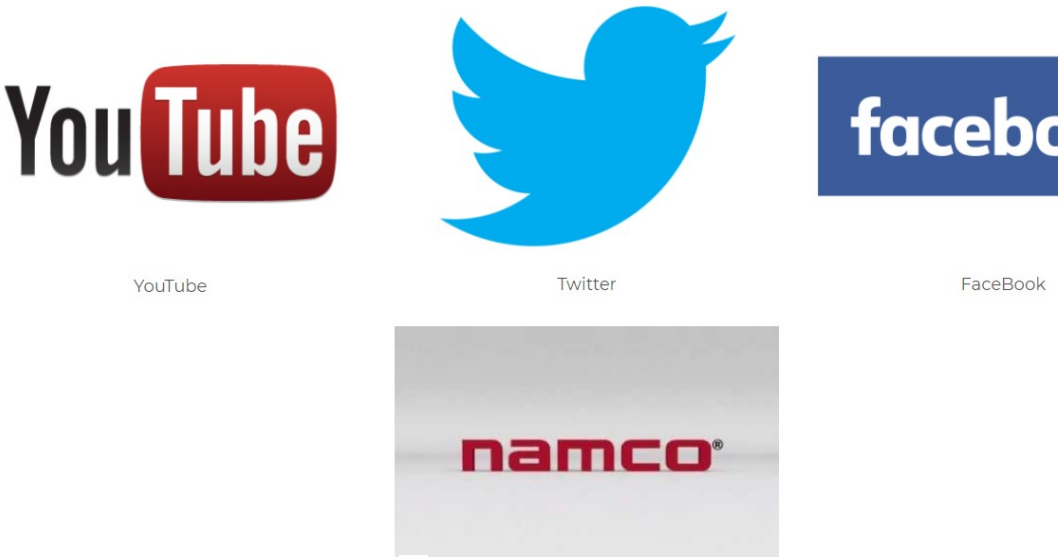

Рисунок 5.1 – Первый запуск

Чтобы перейти к самой игре, нужно нажать на кнопку «Играть» (рис.5.2):

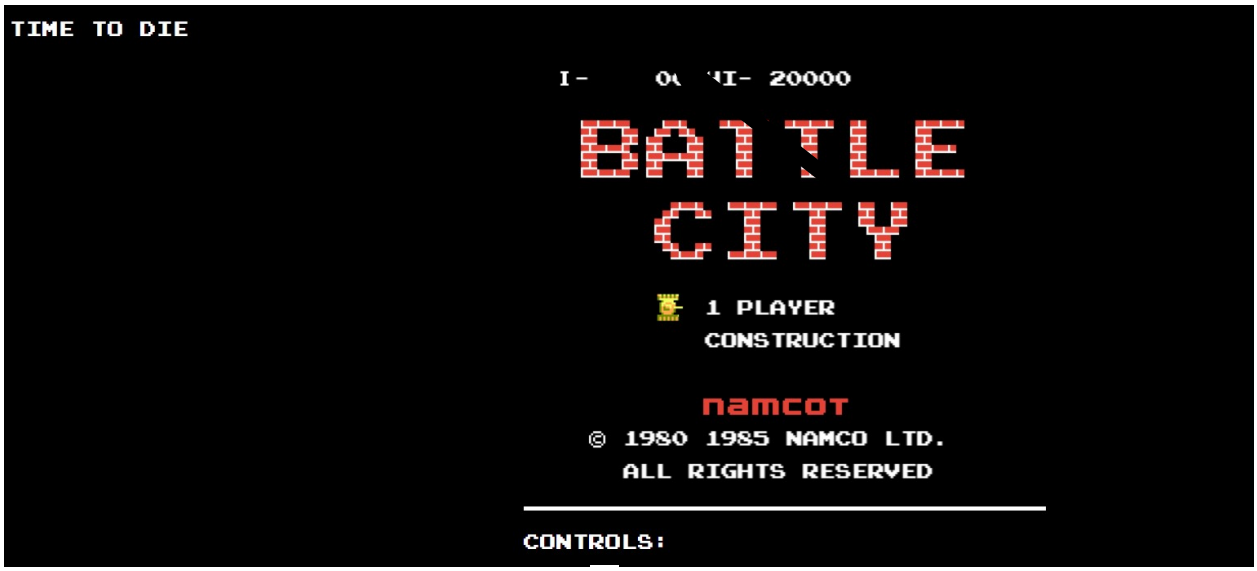

Рисунок 5.2 – Открытие игры

Тест №2. Начало игры. Ход игры

Чтобы начать играть в «Battle City», необходимо нажать на ячейку игрового поля с надписью “1 Players”.

(рис.5.3):

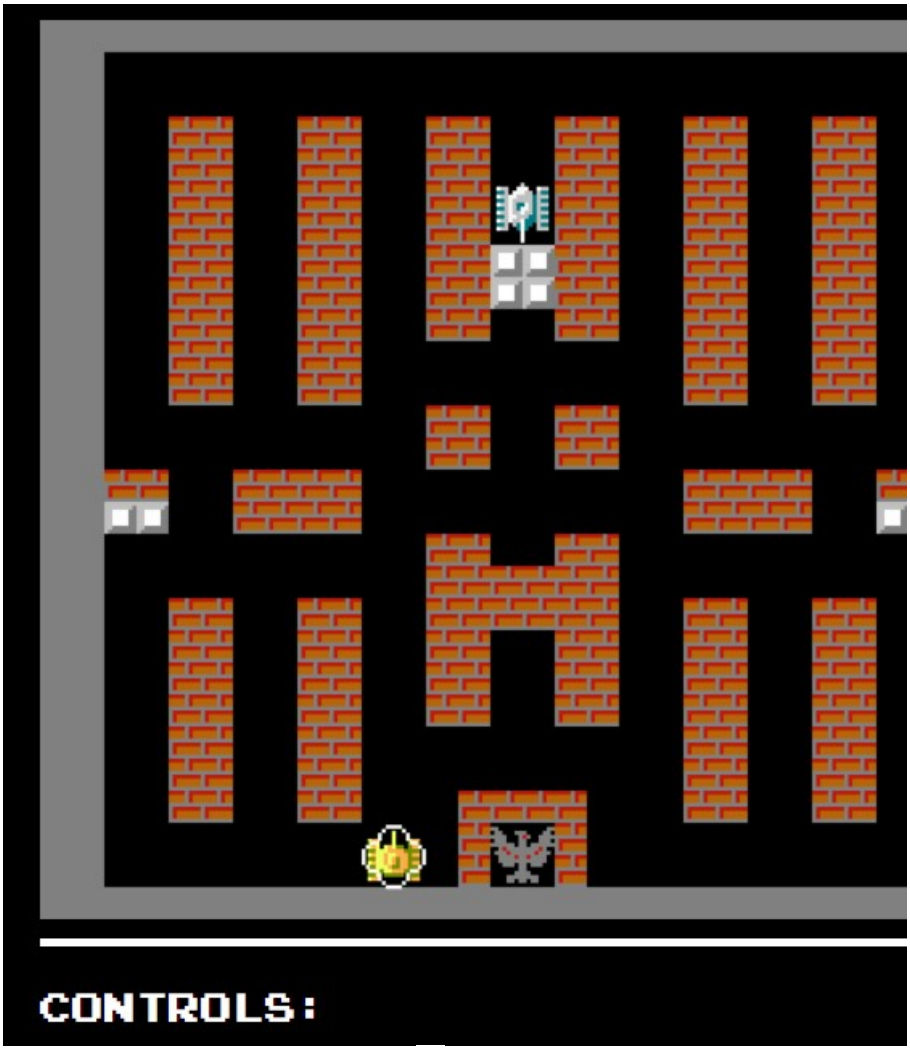

Рисунок 5.3 – Ход игры

Тест №3. Переход на следующий уровень

Для того чтобы перейти на следующий уровень требуется пройти предыдущий , что потребует некоторых усилий. После нескольких часов сражений с искусственным интеллектом , мы победили (рис 5.4):

Рисунок 5.4 – Кнопка «Меню»

Тест №4.Таймер

Таймер должен исправно работать , и показывать затраченное время на уровень , и обнуляться при каждом новом уровне.

Рисунок 5.5 – Таймер

Как видим таймер после перехода на новый уровень обнуляется и идёт заново. (рис 5.8):

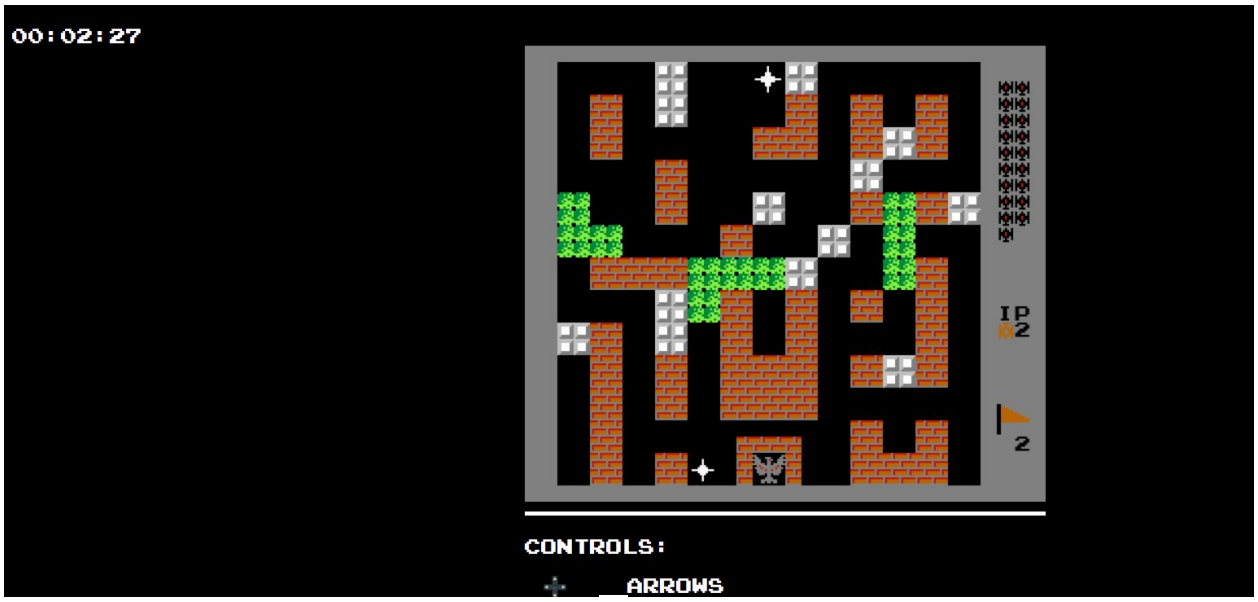

Рисунок 5.6 – Новый таймер

Тест №5. Счёт

При прохождении или поражении игроку будет выводиться его счёт , с убитыми врагами за которых будут начисляться очки.

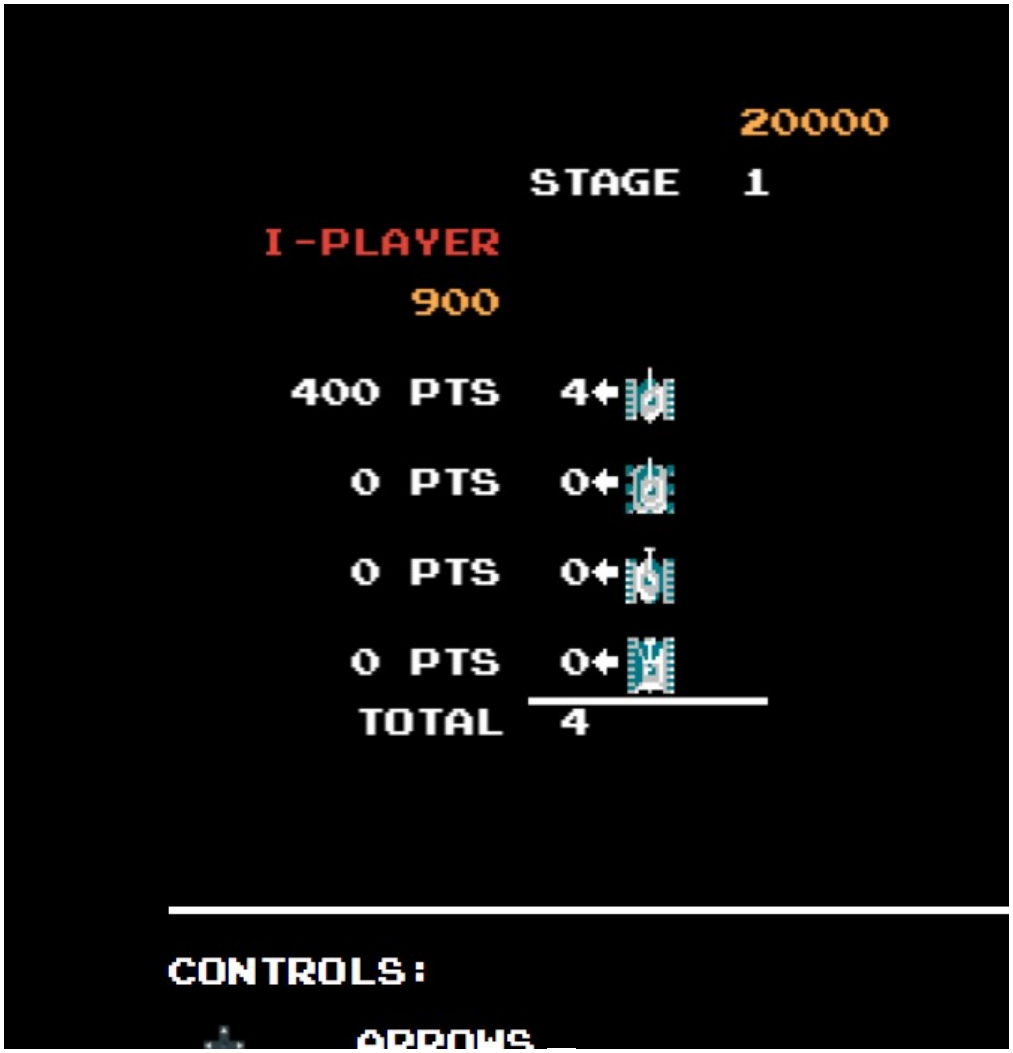

Рисунок 5.7 – Счёт

Тест №6. Поражение

При уничтожении нашей базы или нашего танка , должно выводиться соответственная надпись “Game Over”.

Рисунок 5.8 – Окончание игры

ЗАКЛЮЧЕНИЕ

В результате проделанной работы был разработан алгоритм и написана сама программа, реализующая работу игры «Battle City», которая была создана при помощи среды быстрой разработки приложений – Visual Studio Code с использованием языка программирования JavaScript, HTML, CSS. Данная программа реализует запуск новой игры, проверку на попадание/не попадание в кораблик, вывод данных о выигрыше, затраченных ходах и затраченном времени, также html-страницу с меню, в котором есть краткая информация об игре, правилах игры и авторе игры.

Для запуска программы необходимо открыть html-файл index.html.

Данная программа может быть использована в учебных целях.

СПИСОК ЛИТЕРАТУРЫ

1. Введение [Электронный ресурс]. Режим доступа: http://historygames.ru/.html. – Дата доступа 04.11.18.

2. JavaScript [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/JavaScript. – Дата доступа 14.11.18.

3. HTML [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/HTML. – Дата доступа 14.11.18.

4. Visual Studio Code [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Visual_Studio_Code. – Дата доступа 14.11.18.

5. CSS [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/CSS. – Дата доступа 14.11.18.