Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Операционные Системы и Системное Программирование (ОСиСП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе
на тему:

«Игра “Спасение инопланетянина”»

БГУИР КП 1-40 01 01 620 ПЗ

Студент: гр. 851006 Морозов И. М.

Руководитель: Жиденко А. Л.

Минск 2020

Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ
Заведующий кафедрой ПОИТ

_____
(подпись)
Лапицкая Н.В. 2020 г.

ЗАДАНИЕ
по курсовому проектированию

Студенту Морозову Илье Матвеевичу

1. Тема работы «Игра “Спасение инопланетянина”»
2. Срок сдачи студентом законченной работы 01.12.2020
3. Исходные данные к работе Документация по WinApi
4. Содержание расчётно-пояснительной записки (перечень вопросов, которые подлежат разработке)

Введение.

1.Анализ прототипов, литературных источников и формирование требований к проектироемому ПС;

2.Разработка алгоритма;

3. Разработка программного средства;

4.Тестирование, экспериментальные исследования и анализ полученных результатов;

5. Руководство пользователя программы;

Заключение, список литературы, ведомость, приложения.

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

1. Схема программы на А1

6. Консультант по курсовому проекту Жиденко А. Л.

7. Дата выдачи задания 05.09.2020 г.

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и процентом от общего объёма работы):

раздел 1 к 20.09.2020 – 15 % готовности работы;

разделы 2, 3 к 13.10.2020 – 30 % готовности работы;

раздел 4 к 02.11.2020 – 60 % готовности работы;

раздел 5 к 26.11.2020 – 90 % готовности работы;

оформление пояснительной записки и графического материала к 01.12.2020 – 100 % готовности работы. Защита курсового проекта с 01.12 по 22.12 2020 г.

РУКОВОДИТЕЛЬ_____ Жиденко А. Л.

(подпись)

Задание принял к исполнению Морозов И.М. _____ 05.09.2020 г.

(дата и подпись студента)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Компьютерные игры сейчас имеют огромную популярность среди пользователей компьютеров. Можно выделить несколько целей, для которых была создана та или иная игра:

– Обучение. Например пасьянс «Косынка», добавленный в Windows 3.0, был предназначен для обучения пользователей пользоваться мышью и графическим интерфейсом. Перетаскивая карты с места на место, можно было улучшить свои навыки пользования мышью. Похожую цель преследовал и «Сапер». Целью игры было также образование — пользователи учились использовать обе клавиши мыши и улучшали скорость и точность движений.

– Развлечение. Большинство современных компьютерных игр создано именно с целью организации времяпрепровождения с получением удовольствия и развлечения пользователя. Эту цель стараются достичь практически все игры.

На данный момент очень популярны игры в жанре шутер. Яркий представитель – онлайн-игра «Counter-Strike: Global Offensive», в которую ежедневно играет около 1 миллиона человек.

Игры в жанре шутер в общем случае развивают стратегическое планирование, командную работу, скорость реакции и мышечную моторику. Также если игра онлайн, то можно встретить единомышленников или интересных людей для дальнейшего общения и совместного времяпрепровождения.

Исходя из вышеперечисленных плюсов неудивительно, что каждый второй человек играет или хотя бы раз играл в компьютерную игру, а за последнее десятилетие количество подростков, играющих в игры, увеличилось с 25% до 63%.

Целью данного курсового проекта является создание одновременно развлекающей и обучающей компьютерной игры в жанре шутер. Она сможет обеспечить развлечение на некоторое время, а также подойдет для начинающих пользователей, ведь постоянные попытки уничтожить враждебные элементы совместно с маневрированием от этих самых элементов помогут освоить клавиатуру и чувствовать себя уверенно при работе с компьютерной мышью.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Обзор аналогов

Существует ряд компьютерных игр 2D в жанре шутер, каждая из них имеет свои преимущества и недостатки.

Хорошим выбором станет Hotline Miami [1]. Она является одним из лучших 2D шутеров с видом сверху. Скриншот из игры представлен на рисунке 1.1.

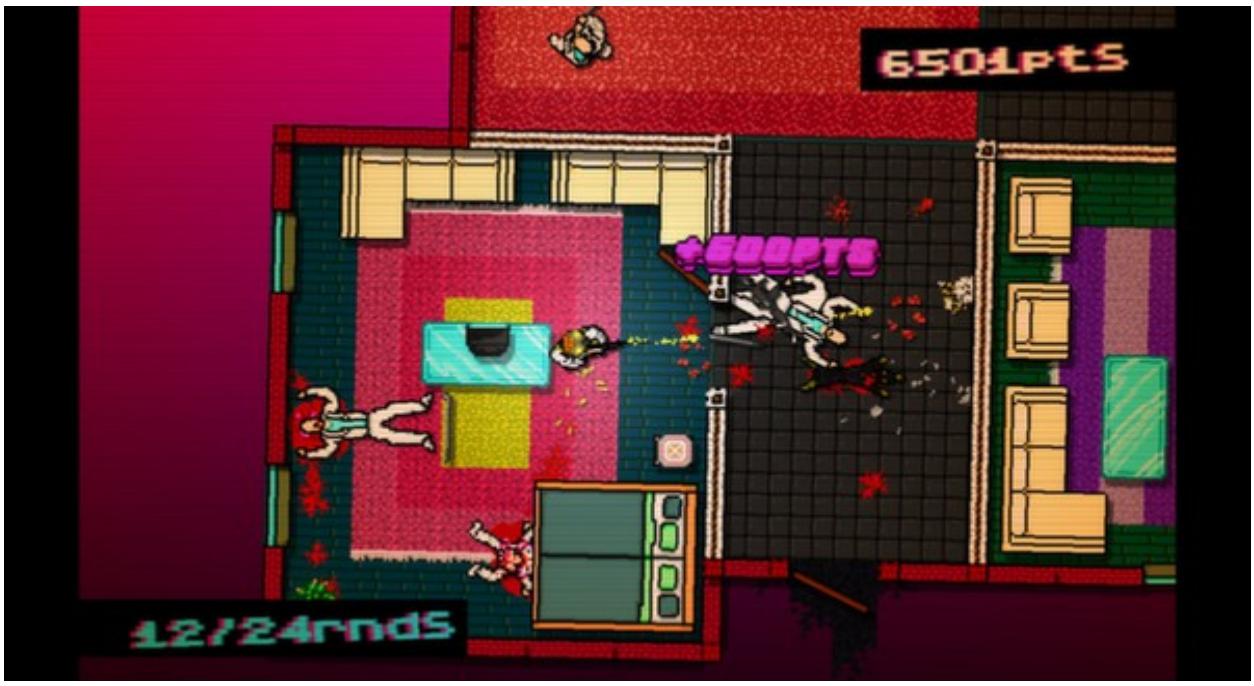

Рисунок 1.1 – Игра Hotline Miami

Hotline Miami разделена на несколько глав, каждая из которых поделена на этапы. На каждом этапе игроку даётся цель — почти всегда она сводится к тому, чтобы убить всех противников. Иногда в конце этапа игроку предстоит сражение с боссом или поиск необходимых для продолжения предметов. Преимуществом данной игры является то, что она имеет проработанный сюжет. Также в игре доступен большой выбор разного оружия. Возможно управление камерой с помощью клавиатуры и мыши. Прохождение игры сопровождается динамичным и приятным саундтреком. Основным минусом является то, что игра платная. Также она не ориентирована на новых пользователей ПК. А именно для новых пользователей будет затруднительно совершать большое количество разнообразных действий.

Еще одним хорошим выбором для любителей шутеров станет Alien Shooter — начало вторжения [2]. Игра не обладает замысловатым сюжетом. Главная цель – убить большое количество монстров, которые совершили побег из секретной правительственной/корпоративной лаборатории, разрабатывающей оружие массового поражения. Скриншот из игры представлен на рисунке 1.2.

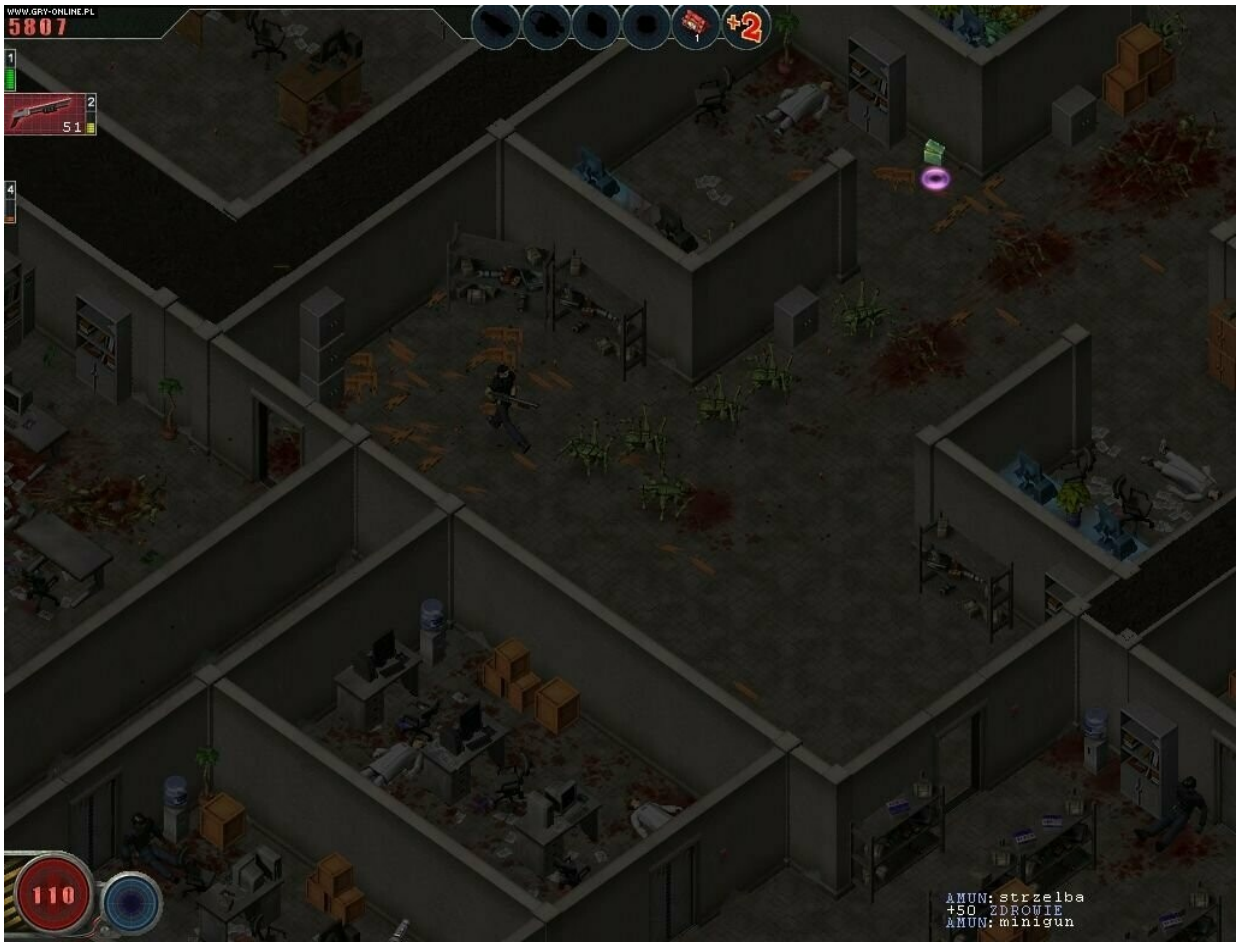

Рисунок 1.2 – Игра Alien Shooter — начало вторжения

Преимуществом данной игры является то, что она бесплатна и мало весит. Сюжет незамысловат, справиться сможет даже начинающий пользователь. Управление камерой интуитивно понятно и выполняется при помощи мыши и клавиатуры. Присутствует возможность сохраняться. Минусом является неоптимизированность игры, а также саундтрек, который надоедает спустя пару часов игры.

Неплохим выбором будет мрачный шутер в киберпанковом стиле — игра RUINER [3]. Она представлена на рисунке 1.3.

Рисунок 1.3 – Игра RUINER

Сюжет игры достаточно банален: спасти похищенного брата, попутно убивая множество врагов. Преимуществом является то, что игра достаточно оптимизирована и имеет большое разнообразие оружия. Также присутствует органичный саундтрек. В игре можно сохраняться, и приятным бонусом будет то, что она является бесплатной. Основной минус этой игры – неудобное управление. Опытные игроки жалуются, что не могут пройти уровни исключительно из-за плохого управления, так что тем, кто только начал освоение компьютера лучше не заходить в эту игру совсем.

1.2 Постановка задачи

В рамках данного курсового проекта планируется разработка программного средства «Игра “Спасение инопланетянина”».

Будут разработаны алгоритмы перемещения объектов, проверка объектов на столкновение, следования камеры за персонажем, уничтожения объектов.

В программном средстве планируется реализовать следующие функции и алгоритмы:

- динамическое генерирование вражеских объектов;
- динамическое генерирование пуль;
- передвижение летающей тарелки по всем направлениям;
- преследование вражеских объектов летающей тарелки;
- фокусировка камеры на летающую тарелку;
- динамическая отрисовка фона;
- отображение набранных очков в течение игры;
- отображение набранных очков в конце игры.

Главной задачей является создать игру, которая поможет обучиться использованию клавиатуры и компьютерной мыши для новых пользователей, а также будет неплохим развлечением для опытных пользователей.

Для разработки программного средства будет использоваться язык программирования C, WinAPI и операционная система Windows 10.

2 РАЗРАБОТКА ПРОГРАММНОГО СРЕДСТВА

2.1 Интерфейс программного средства

Был проведен анализ существующих аналогов и был выдвинут ряд пожеланий к созданному пользовательскому интерфейсу:

- Доходчивость интерфейса, обеспечивающая понимание структуры приложения пользователем;
- Лаконичность интерфейса, обеспечивающая отсутствие перегрузки пользователя информацией;
- Восприимчивость интерфейса, обеспечивающая короткое время отклика приложения на действия пользователя;
- Эффективность интерфейса, обеспечивающая быстродействие приложения.

В результате рассмотрения пожеланий к пользовательскому интерфейсу было разработано приложение, соответствующее пунктам. Внешний вид программного средства представлен на рисунках 2.1.

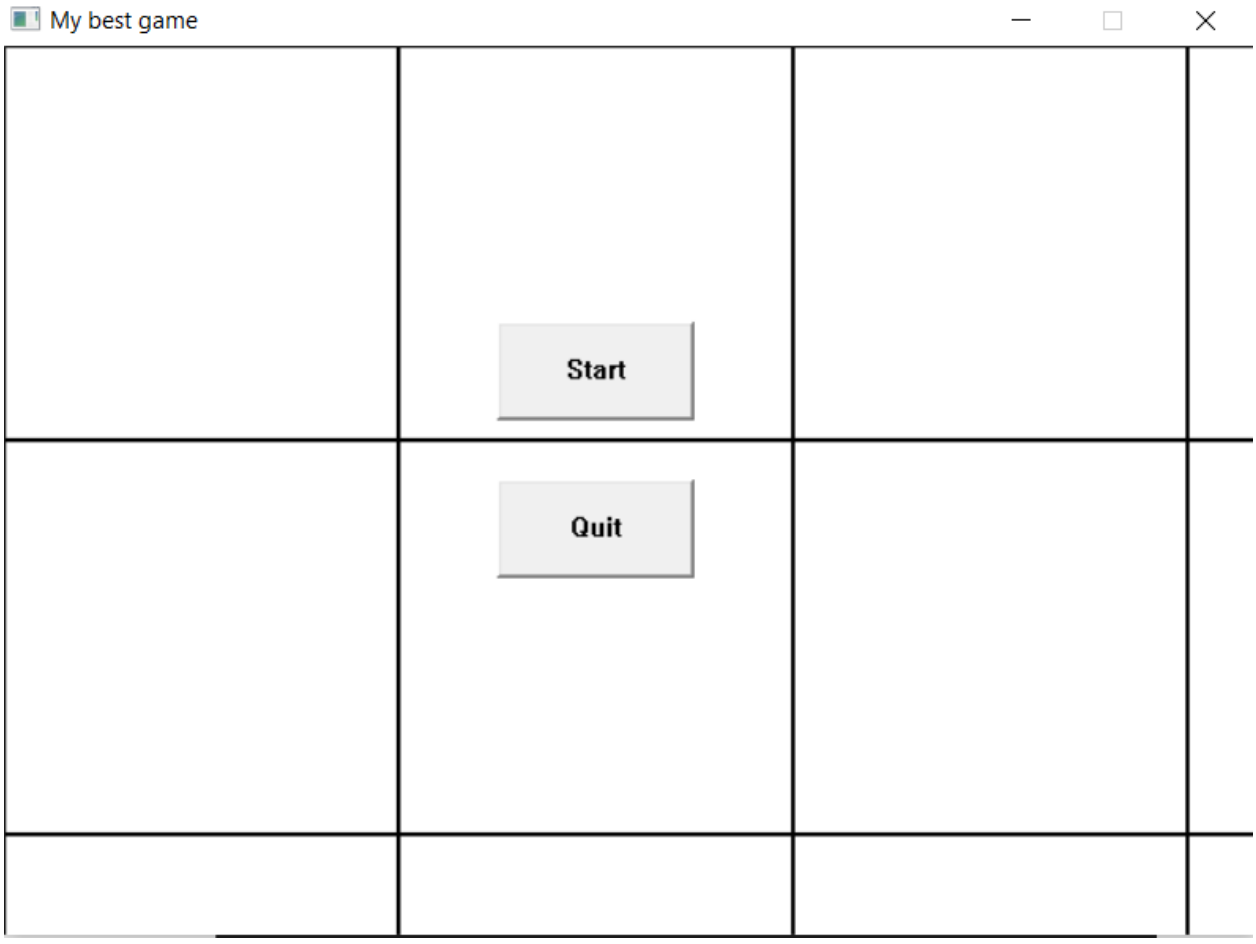

Рисунок 2.1 – Интерфейс программы

2.2 Основной цикл программы

Основной цикл программы отличается от стандартного приложения на Windows API. Он представлен на рисунке 2.2.

```
MSG msg;
while (1)
{
    if (PeekMessageA(&msg, NULL, 0, 0, PM_REMOVE))
    {
        if (msg.message == WM_QUIT) break;
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    else
    {
         if (startGame){
            WinMove();
            WinShow(dc);
            Sleep(5);
         }
    }
}
```

Рисунок 2.2 – Основной цикл программы

Основное отличие заключается в том, что в стандартном приложении на Windows API обработка всех событий сопровождается сообщениями, которые обрабатываются в соответствующей функции.

Функция PeekMessageA проверяет наличие сообщений в очереди. Если сообщение есть – тогда она записывает его в msg и затем идет обработка сообщения. Флаг PM_REMOVE означает, что после получения сообщения из очереди это сообщение удаляется из очереди. Если сообщений нет, то происходит перерисовка.

2.3 Обработка нажатий

Управление осуществляется клавишами W, A, S, D. При нажатии клавиши W – происходит перемещение вверх, A – влево, S – вниз, D – вправо. Также есть движение происходит по диагонали (зажаты сразу две клавиши), то происходит корректировка скорости, чтобы летающая тарелка не двигалась слишком быстро. Блок-схема алгоритма представлена на рисунке 2.3.

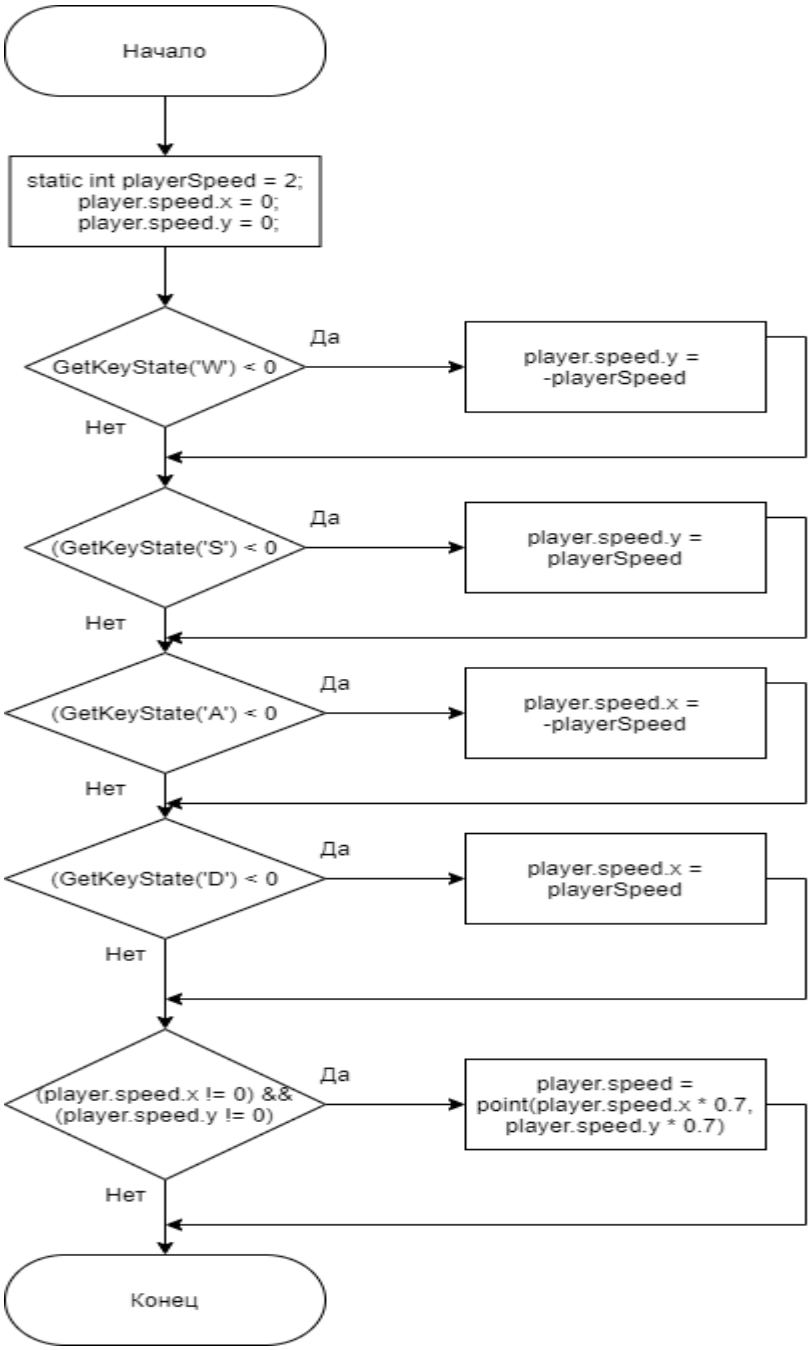

Рисунок 2.3 – Блок-схема алгоритма обработки клавиш

2.4 Генерирование врагов

Для набора очков пользователю необходимы вражеские объекты, которые он может уничтожать. Враги генерируются в случайном месте на расстоянии 300 от летающей тарелки. Пользователь может убить вражеский объект, если попадет в него пулей. Каждое убийство вражеского объекта прибавляет 1 очко. Пример появления врага представлен на рисунке 2.4.

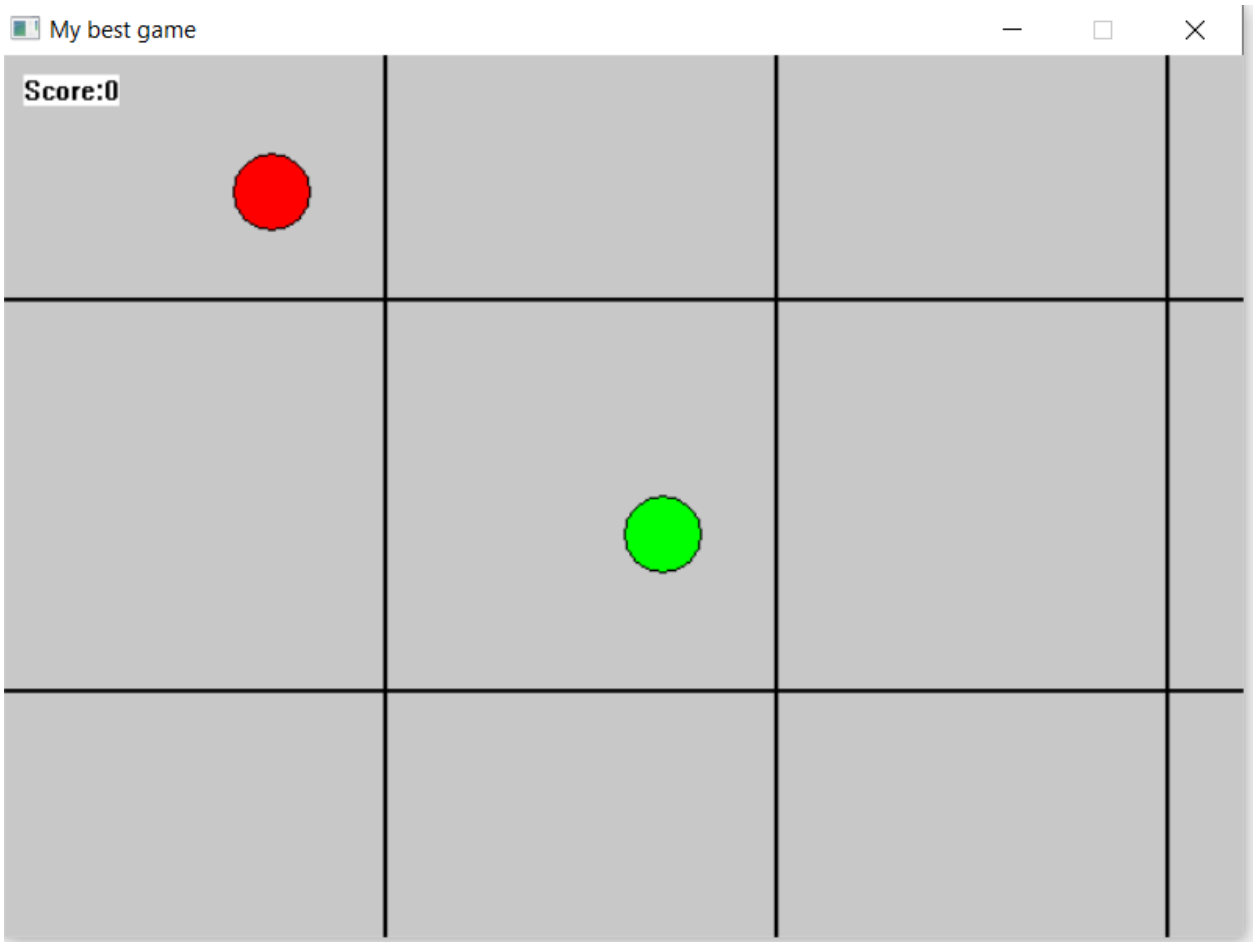

Рисунок 2.4 – Пример появления вражеского объекта

2.5 Отображение набранных очков в конце игры

Проведя анализ прототипов, было выявлено, что не каждое приложение отображает количество набранных очков в конце игры. Я решил создать эту возможность, чтобы у пользователя было время отдохнуть после продолжительной игры. Пример отображения количества набранных очков представлен на рисунке 2.5.

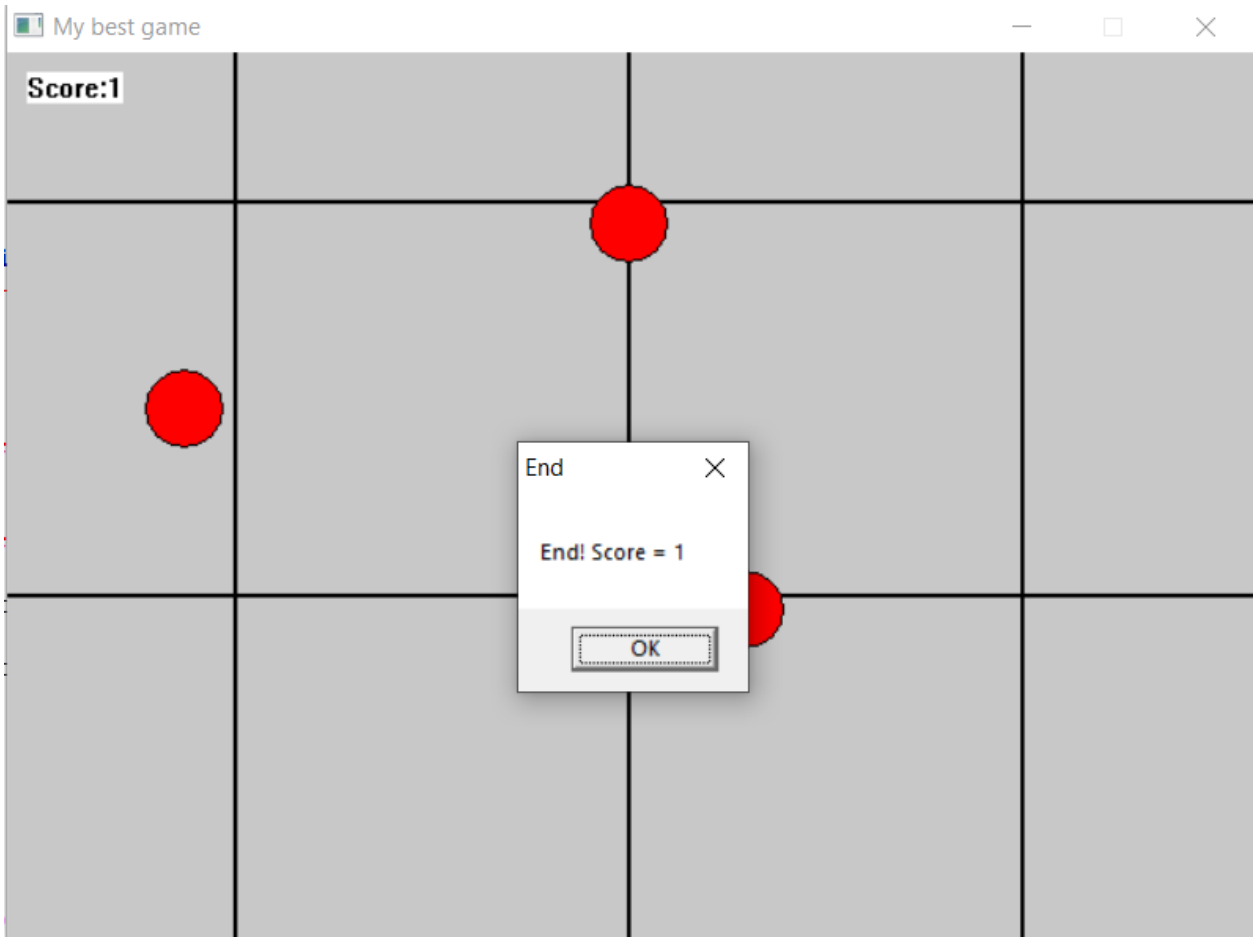

Рисунок 2.5 – Отображение набранных очков

3 ТЕСТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

В ходе тестирования приложения не было выявлено недостатков программного средства. Была составлена таблица 3.1, показывающая ожидаемые и реальные результаты, полученные при заданных условиях, она представлена ниже.

Таблица 3.1 – Ожидаемые и реальные результаты тестирования

№	Тестовые случаи	Ожидаемый результат	Полученный результат
1.	Запуск программы	Успешный запуск и инициализация начальных значений параметров программы	Тест пройден
2.	Запуск игры	Появление летающей тарелки и заднего фона, генерация врагов	Тест пройден
3.	Движение по горизонтали	Перемещение летающей тарелки вверх при нажатии W и вниз при нажатии S	Тест пройден
4.	Движение по вертикали	Перемещение летающей тарелки влево при нажатии A и вправо при нажатии D	Тест пройден
5.	Движение по диагонали	Перемещение летающей тарелки по диагонали при нажатии комбинаций W+A, W+D, S+A, S+D	Тест пройден
6.	Убийство врага	Попадание пулей во врага с помощью нажатия левой кнопки мыши	Тест пройден
7.	Начисление очков	Попадание пулей во врага с помощью нажатия левой кнопки мыши и прибавление 1 очка	Тест пройден

8.	Конец игры при столкновении с врагом	Завершение игры	Тест пройден
9.	Отображение набранных очков	Отображения сообщения о конце игры с количеством набранных очков	Тест пройден
10.	Выход из игры	Выход из игры путем нажатия кнопки Quit	Тест пройден

Разработка приложения велась с использованием системы контроля версий GitHub, позволившая сохранять состояние программы на каждом отдельном этапе по ходу добавления нового функционала или изменения уже существующего. Появление новых точек возврата происходит посредством группировки изменённых файлов, затем они объединяются под общим именем «коммита», в котором кратко изложена суть изменений. Также можно добавлять к каждому этапу новые файлы, или удалять устаревшие варианты. После накопления определённого количества групп изменений, их следует отправить на удалённый репозиторий, где видна вся история приложения и разница между каждым новым «коммитом».

Путем тщательной проверки тестами, было выявлено, что ошибок в работе программы нет.

4 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

4.1 Основные требования для запуска

- ОС: Версия Microsoft Windows от 8 и выше;
- Процессор: не менее 1 ГГц;
- RAM: От 1 Гб;
- Место на диске: от 2 Мб свободного места.

Исходя из данных требований следует, что данная программа может запускаться практически на любом компьютере.

4.2 Руководство по установке

Установка программы не требуется, т.к она является переносимым приложением.

Переносимое приложение — программное обеспечение, которое для своего запуска не требует процедуры установки и может полностью храниться на съёмных носителях информации, что позволяет использовать данное ПО на многих компьютерах.

4.3 Руководство по использованию

1) Запустить программу (рисунок 4.1)

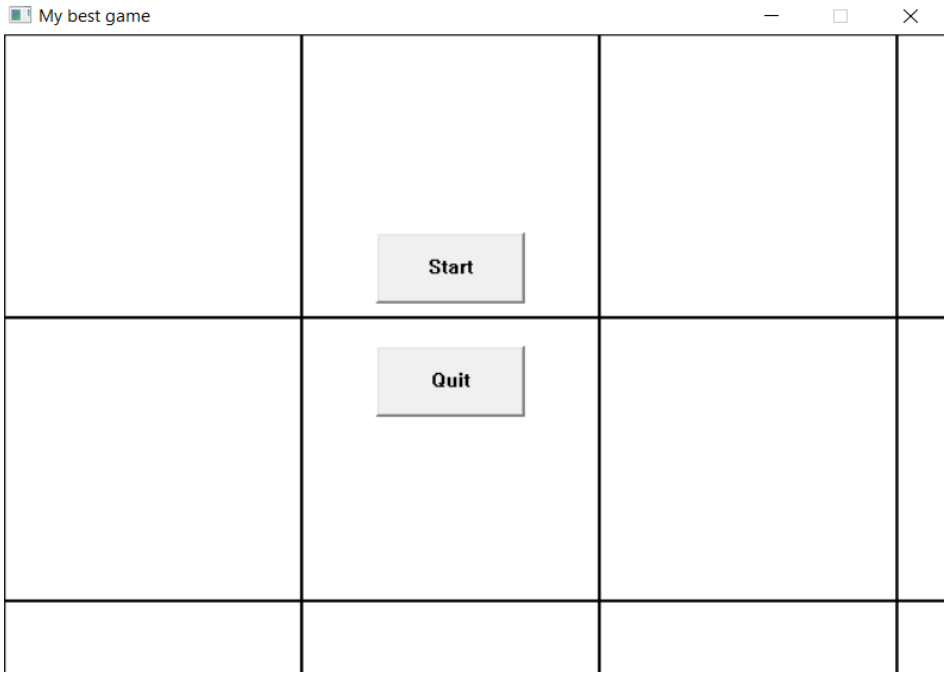

Рисунок 4.1 – Скриншот запущенной программы

2) Запустить игру путем нажатия на кнопку Start (рисунок 4.2)

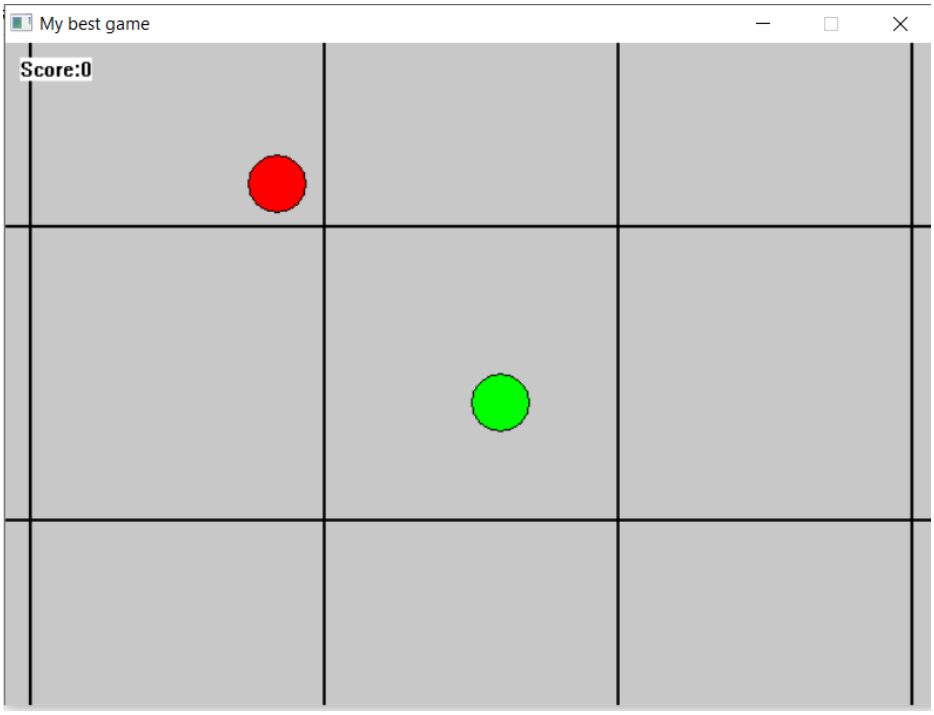

Рисунок 4.2 – Скриншот начатой игры

3) Убийство врага и заработок 1 очка путем направления курсора на врага и нажатия левой кнопки мыши (рисунок 4.3)

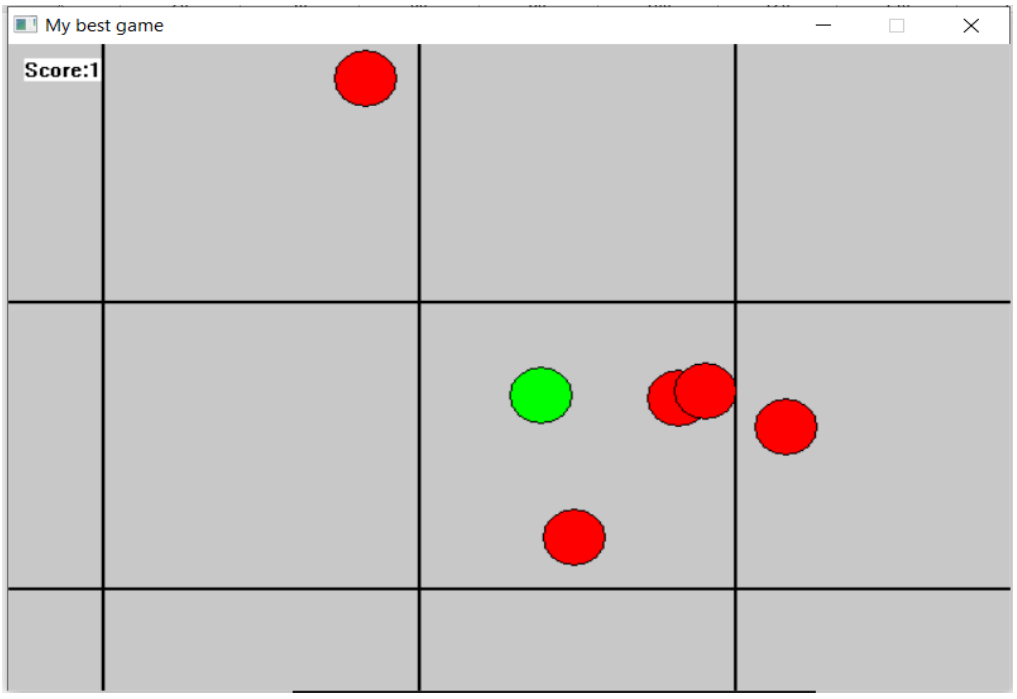

Рисунок 4.3 – Скриншот убийства врага и заработок 1 очка

4) При соприкосновении с противником игра будет закончена и выведется количество очков (рисунок 4.4)

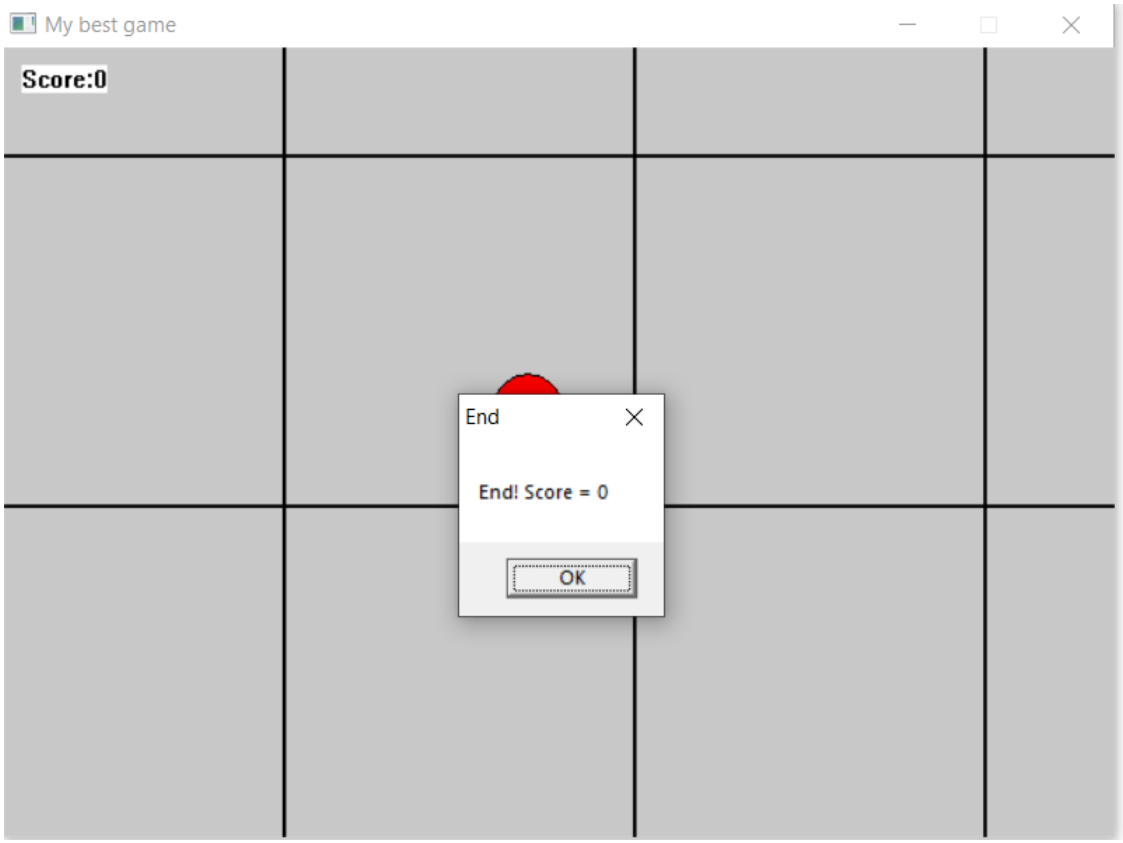

Рисунок 4.4 – Скриншот конца игры и вывода очков

5) Нажать ОК и закрыть программу

ЗАКЛЮЧЕНИЕ

В настоящее время существует много компьютерных игр в жанре шутер с разными сюжетами, локациями и видами стрельбы. Благодаря столь большой популярности этих игр среди молодежи, которая довольно опытна в использовании компьютера, большинство шутеров не рассчитаны на пользователей, которые только сели постигать компьютер.

В рамках данного курсового проекта было разработано игровое приложение, позволяющее развлечь себя на некоторое время, а также обучиться и отточить навыки использования клавиатуры и компьютерной мыши.

Стоит отметить, что также были улучшены навыки владения таким языком программирования как C и детально изучено взаимодействие с WinAPI.

Разработанное программное средство имеет способность осуществлять следующий основной функционал:

- динамическое генерирование вражеских объектов;
- динамическое генерирование пуль;
- передвижение летающей тарелки по всем направлениям;
- преследование вражеских объектов летающей тарелки;
- фокусировка камеры на летающую тарелку;
- динамическая отрисовка фона;
- отображение набранных очков в течение игры;
- отображение набранных очков в конце игры.

Написанный код возможно модифицировать для добавления нового функционала. В дальнейшем планируются улучшения и доработки, вносящие в программу такие возможности как игра по интернету, испытания.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] www.store.steampowered.com [Электронный портал]. – Электронные данные. – Режим доступа: https://store.steampowered.com/app/219150/Hotline_Miami/

[2] www.dtf.ru [Электронный портал]. – Электронные данные. – Режим доступа: https://dtf.ru/games/178558-alien-shooter-nachalo-vtorzheniya

[3] www.store.steampowered.com [Электронный портал]. – Электронные данные. – Режим доступа: https://store.steampowered.com/app/464060/RUINER/

ПРИЛОЖЕНИЕ 1
Исходный код программы

Файл main.c

```
#include <stdio.h>
#include <stdlib.h>

#include <windows.h>
#include <string.h>

#define bt1_id 1
#define bt2_id 2

#define AddEnemy(a,b) ObjectInit(NewObject(), a, b, 40, 40, 'e')

typedef struct SPoint {
   float x, y;
} TPoint;

TPoint point (float x, float y)
{
   TPoint pt;
   pt.x = x;
   pt.y = y;
   return pt;
}

TPoint cam;
HWND hwnd;
HWND bt1;
HWND bt2;

typedef struct SObject {
   TPoint pos;
   TPoint size;
   COLORREF brush;
   TPoint speed;
   char oType;
   float range, vecSpeed;
   BOOL isDel;
} TObject, *PObject;

BOOL ObjectCollision(TObject o1, TObject o2)
```

```
{
   return ((o1.pos.x + o1.size.x) > o2.pos.x) && (o1.pos.x < (o2.pos.x +
o2.size.x)) &&
          ((o1.pos.y + o1.size.y) > o2.pos.y) && (o1.pos.y < (o2.pos.y +
o2.size.y));
}

void ObjectInit (TObject *obj, float xPos, float yPos, float width, float height,
char objType)
{
   obj->pos = point(xPos, yPos);
   obj->size = point(width, height);
   obj->brush = RGB(0, 255, 0);
   obj->speed = point(0,0);
   obj->oType = objType;
   if (objType == 'e') obj->brush = RGB(255,0,0);
   obj->isDel = FALSE;
}

void ObjectShow(TObject obj, HDC dc)
{
   SelectObject(dc, GetStockObject(DC_PEN));
   SetDCPenColor(dc, RGB(0,0,0));
   SelectObject(dc, GetStockObject(DC_BRUSH));
   SetDCBrushColor(dc, obj.brush);

   BOOL (*shape)(HDC, int, int, int, int);
   shape = Ellipse;
   shape(dc, (int)(obj.pos.x - cam.x), (int)(obj.pos.y - cam.y),
            (int)(obj.pos.x + obj.size.x - cam.x), (int)(obj.pos.y + obj.size.y -
cam.y));
}

void ObjectSetDestPoint(TObject *obj, float xPos, float yPos, float vecSpeed)
{
   TPoint xyLen = point(xPos - obj->pos.x, yPos - obj->pos.y);
   float dxy = sqrt(xyLen.x * xyLen.x  + xyLen.y * xyLen.y);
   obj->speed.x = xyLen.x / dxy * vecSpeed;
   obj->speed.y = xyLen.y / dxy * vecSpeed;
   obj->vecSpeed = vecSpeed;
}

RECT rct;
```

```
TObject player;

PObject mas = NULL;
int masCnt = 0;
int score = 0;

BOOL needNewGame = FALSE;
BOOL startGame = FALSE;

void SetCameraFocus(TObject obj)
{
   cam.x = obj.pos.x - rct.right / 2;
   cam.y = obj.pos.y - rct.bottom / 2;
}

void ObjectMove(TObject *obj)
{
   if (obj->oType == 'e')
   {
      if (rand() % 40 == 1)
      {
         static float enemySpeed = 1.5;
         ObjectSetDestPoint(obj, player.pos.x, player.pos.y, enemySpeed);
      }
      if (ObjectCollision(*obj, player))
         needNewGame = TRUE;
   }
   obj->pos.x += obj->speed.x;
   obj->pos.y += obj->speed.y;

   if (obj->oType == '1')
   {
      obj->range = obj->vecSpeed;
      if (obj->range < 0)
        obj->isDel = TRUE;
      for (int i = 0; i < masCnt; i++)
         if ((mas[i].oType == 'e') && (ObjectCollision(*obj, mas[i])))
         {
            mas[i].isDel = TRUE;
            obj->isDel = TRUE;
            score++;
         }
   }
```

```
}

PObject NewObject()
{
   masCnt++;
   mas = realloc(mas, sizeof(*mas) * masCnt);
   return mas + masCnt - 1;
}

void GenNewEnemy()
{
   static int rad = 300;
   int pos1 = (rand() % 2 == 0 ? -rad : rad);
   int pos2 = (rand() % (rad * 2)) - rad;
   int k = rand() % 100;
   if (k == 1)
      AddEnemy(player.pos.x + pos1, player.pos.y + pos2);
   if (k == 2)
      AddEnemy(player.pos.x + pos2, player.pos.y + pos1);
}

void DelObjects()
{
   int i = 0;
   while (i < masCnt)
   {
      if (mas[i].isDel)
      {
         masCnt--;
         mas[i] = mas[masCnt];
         mas = realloc(mas, sizeof(*mas) * masCnt);
      }
      else
         i++;
   }
}

void AddBullet(float xPos, float yPos, float x, float y)
{
   PObject obj = NewObject();
   ObjectInit(obj, xPos, yPos, 10, 10, '1');
   ObjectSetDestPoint(obj, x, y, 4);
   obj->range = 300;
```

```
}

void PlayerControl()
{
   static int playerSpeed = 2;
   player.speed.x = 0;
   player.speed.y = 0;
   if (GetKeyState('W') < 0) player.speed.y = -playerSpeed;
   if (GetKeyState('S') < 0) player.speed.y = playerSpeed;
   if (GetKeyState('A') < 0) player.speed.x = -playerSpeed;
   if (GetKeyState('D') < 0) player.speed.x = playerSpeed;
   if ((player.speed.x != 0) && (player.speed.y != 0))
      player.speed = point(player.speed.x * 0.7, player.speed.y * 0.7);
}

void WinInit()
{
   needNewGame = FALSE;
   masCnt = 0;
   score = 0;
   mas = realloc(mas, 0);

   ObjectInit(&player, 100, 100, 40, 40, 'p');
}

void ShowResult()
{
   char str[20];
   str[0] = 'E';
   str[1] = 'n';
   str[2] = 'd';
   str[3] = '!';
   str[4] = ' ';
   str[5] = 'S';
   str[6] = 'c';
   str[7] = 'o';
   str[8] = 'r';
   str[9] = 'e';
   str[10] = ' ';
   str[11] = '=';
   str[12] = ' ';
   str[13] = 0;
   int radix = 10;
```

```
        char buffer[20];
        char *p;
        p = itoa(score, buffer, radix);
        MessageBox(hwnd, (LPCSTR)(strcat(str, buffer)), (LPCSTR)("End"),
MB_OK);
    }

    void WinMove()
    {
        if (needNewGame)
        {
            ShowResult();
            EnableWindow(bt1, TRUE);
            EnableWindow(bt2, TRUE);
            ShowWindow(bt1,SW_SHOW);
            ShowWindow(bt2,SW_SHOW);

            startGame = FALSE;

            WinInit();
        }

        PlayerControl();
        ObjectMove(&player);
        SetCameraFocus(player);

        for (int i = 0; i < masCnt; i++)
            ObjectMove(mas + i);

        GenNewEnemy();
        DelObjects();
    }

    void ShowBackground(HDC memDC)
    {
        static int rectSize = 200;
        int dx = (int)(cam.x) % rectSize;
        int dy = (int)(cam.y) % rectSize;
        for (int i = -1; i < (rct.right / rectSize) + 2; i++)
            for (int j = -1; j < (rct.bottom / rectSize) + 2; j++)
                Rectangle(memDC, -dx+(i*rectSize), -dy+(j*rectSize), -
dx+((i+1)*rectSize), -dy+((j+1)*rectSize));
    }
```

```
void WinShow(HDC dc)
{
   HDC memDC = CreateCompatibleDC(dc);
   HBITMAP memBM = CreateCompatibleBitmap(dc, rct.right - rct.left, rct.bottom - rct.top);
   SelectObject(memDC, memBM);

   SelectObject(memDC, GetStockObject(DC_PEN));
   SelectObject(memDC, GetStockObject(DC_BRUSH));
   SetDCBrushColor(memDC, RGB(200, 200, 200));

   ShowBackground(memDC);

   char str[20];
   str[0] = 'S';
   str[1] = 'c';
   str[2] = 'o';
   str[3] = 'r';
   str[4] = 'e';
   str[5] = ':';
   str[6] = 0;
   int radix = 10;
   char buffer[20];
   char *p;
   p = itoa(score, buffer, radix);

   TextOutA(memDC, 10, 10, (LPCSTR)(strcat(str, buffer)), lstrlen(strcat(str, buffer)));


   ObjectShow(player, memDC);

   for (int i = 0; i < masCnt; i++)
      ObjectShow(mas[i], memDC);

   BitBlt(dc, 0, 0, rct.right - rct.left, rct.bottom - rct.top, memDC, 0, 0, SRCCOPY);
   DeleteDC(memDC);
   DeleteObject(memBM);
}

LRESULT  WndProc (HWND hWnd, UINT message, WPARAM wParam,
```

```
LPARAM lParam)
	{
	  if (message == WM_DESTROY)
	  {
	    PostQuitMessage(0);
	  }
	  else if (message == WM_MOUSEMOVE)
	  {
	    int xPos = LOWORD(lParam);
	    int yPos = HIWORD(lParam);
	  }
	  else if (message == WM_LBUTTONDOWN)
	  {
	    int xPos = LOWORD(lParam);
	    int yPos = HIWORD(lParam);
	    AddBullet(player.pos.x + player.size.x / 2, player.pos.y + player.size.y /
2, xPos + cam.x, yPos + cam.y);
	  }
	  else if (message == WM_SIZE)
	  {
	    GetClientRect(hWnd, &rct);
	  }
	  else if (message == WM_COMMAND)
	  {
	    if (LOWORD(wParam) == bt1_id)
	    {
	      PostQuitMessage(0);
	    }
	    if (LOWORD(wParam) == bt2_id)
	    {
	      startGame = TRUE;
	      EnableWindow(bt1, FALSE);
	      EnableWindow(bt2, FALSE);
	      ShowWindow(bt1,SW_HIDE);
	      ShowWindow(bt2,SW_HIDE);
	    }
	  }

	  else return DefWindowProcA( hWnd,  message,  wParam,  lParam);
	}

	int main()
	{
```

```
WNDCLASSA wcl;
   memset(&wcl, 0, sizeof(WNDCLASSA));
   wcl.lpszClassName = "My window";
   wcl.lpfnWndProc = WndProc;
   wcl.hCursor = LoadCursorA(NULL, IDC_CROSS);
RegisterClassA(&wcl);

hwnd = CreateWindow("My window", "My best game",
WS_OVERLAPPEDWINDOW&~(WS_MAXIMIZEBOX|WS_THICKFRAME),
                    450, 150, 640, 480, NULL, NULL, NULL, NULL);

HDC dc = GetDC(hwnd);

ShowWindow(hwnd, SW_SHOWNORMAL);

ShowBackground(dc);
bt1 = CreateWindow("button", "Quit", WS_VISIBLE | WS_CHILD,
                    250, 220, 100, 50, hwnd, bt1_id, NULL, NULL);

bt2 = CreateWindow("button", "Start", WS_VISIBLE | WS_CHILD,
                    250, 140, 100, 50, hwnd, bt2_id, NULL, NULL);

WinInit();

MSG msg;
while (1)
{
   if (PeekMessageA(&msg, NULL, 0, 0, PM_REMOVE))
   {
      if (msg.message == WM_QUIT) break;
      TranslateMessage(&msg);
      DispatchMessage(&msg);
   }
   else
   {
       if (startGame){
         WinMove();
         WinShow(dc);
         Sleep(5);
       }
   }
}
printf("Hello world!\n");
```

```
        return 0;
}
```

Обозначение	Наименование	Дополнительные сведения
	Текстовые документы	
БГУИР КП 1–40 01 01 620 ПЗ	Пояснительная записка	32 с.
	Графические документы	
ГУИР 851006 01 ПД	Схема программы на А1	Формат А1

					БГУИР КП 1-40 01 01 620 Д1			
Изм.	Л.	№ докум.	Подп.	Дата			Лист	Листов
Разраб.		Морозов И.М..			Игра “Спасение инопланетянина”	Т	1	32
Пров.		Жиденко А. Л.				Кафедра ПОИТ гр. 851006		