­Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Компьютерные системы и сети (КСиС)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту на тему

Программное средство «Игровое приложение танки»

БГУИР КП 1-40 01 01 014 ПЗ

Студент: гр. 151004 Дубовский А.В.

Руководитель: Красковский П.Н

Минск 2022

содержание

[Введение 3](#_Toc136645322)

[1 Аналитический обзор литературы 4](#_Toc136645323)

[1.1 Обзор существующих аналогов 4](#_Toc136645324)

[1.2 Перечень функциональных требований 6](#_Toc136645325)

[1.3 Состав и параметры технических и программных средств 6](#_Toc136645326)

[2 Моделирование предметной области 7](#_Toc136645327)

[2.1 Протокол UDP 7](#_Toc136645328)

[3 Проектирование программного средства 8](#_Toc136645329)

[3.1 Структура программы 8](#_Toc136645330)

[3.2 Проектирование интерфейса программного средства 8](#_Toc136645331)

[3.3 Проектирование функционала программного средства 9](#_Toc136645332)

[4 Разработка программного средства 12](#_Toc136645333)

[4.1 Проверка столкновения танка 12](#_Toc136645334)

[4.2 Движение танка 13](#_Toc136645335)

[4.3 Создание снаряда 14](#_Toc136645336)

[5 Тестирование программного средства 15](#_Toc136645337)

[6 Руководство пользователя 16](#_Toc136645338)

[Заключение 17](#_Toc136645339)

[Список использованной литературы 18](#_Toc136645340)

[Приложение А 19](#_Toc136645341)

# Введение

Данный курсовой проект посвящён обновлению легендарной игры «Танчики», для преобщения молодёжи к класике.

Первая версия «Танчиков» выпущина в 1980г компание «Namco».

В основе игрового процесса «Танчиков» лежит перемещение по карте с последующим отстрелом врагов.

В 1985г компания «Namco» выпустила обновлённую версию танчиков для консоли «Famicon».

В 2004г компаания «Бука» выпустила 3D версию «Танчиков».

# Аналитический обзор литературы

## Обзор существующих аналогов

### Игророве приложение «Танчики Денди 1990»

Программное средство, изображённое на рисунке 1.1, является наиболее известной итерацией игрового приложения танчики. Ввиду это являеться основным конкурентом разрабатываемого приложения.

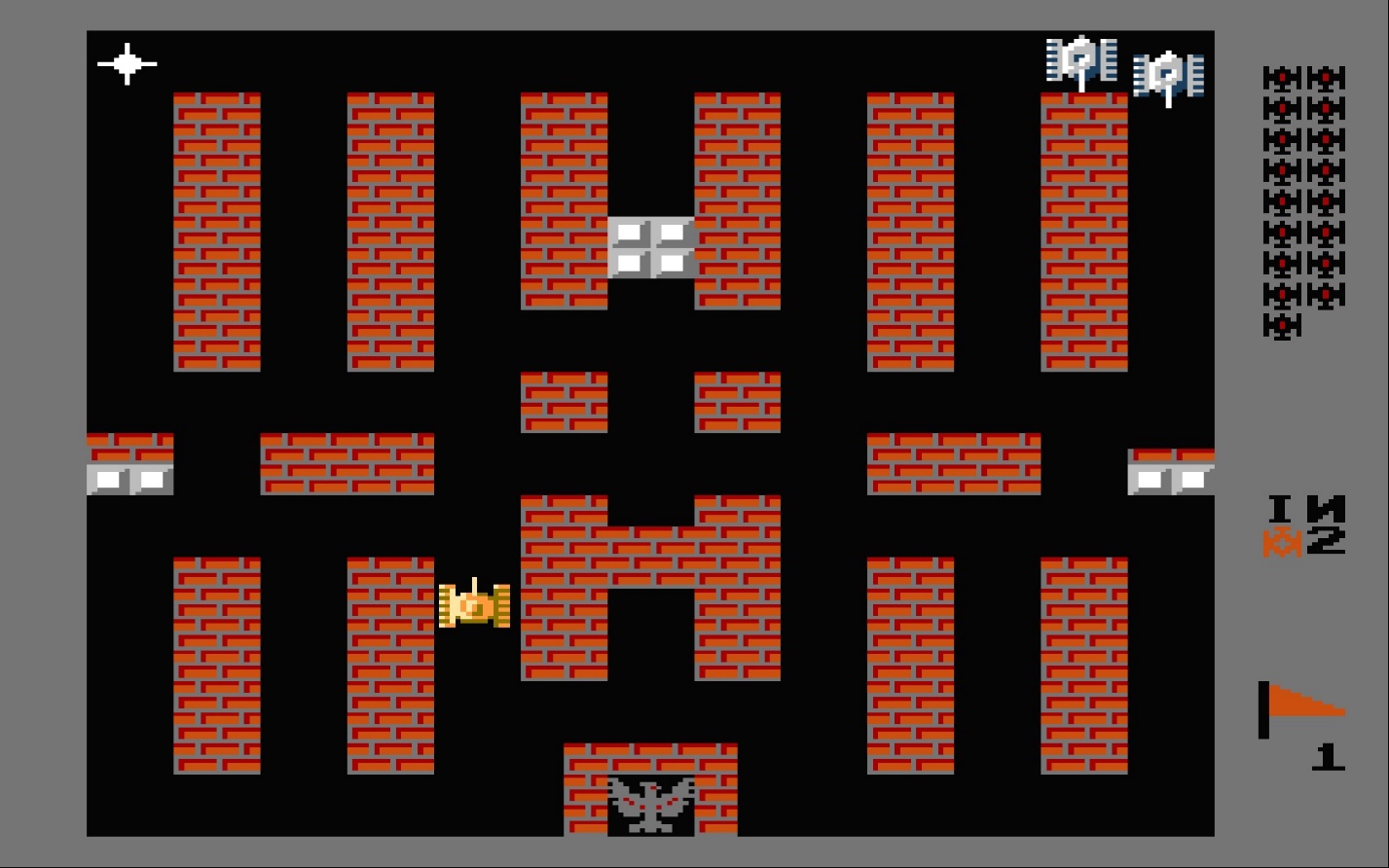


Рисунок 1.1 – Игророве приложение «Танчики Денди 1990»

Список выполняемых функций:

* управление танком;
* сражение с ботами.

Рассмотрим достоинства данного приложения:

* наличие 14 уровней;
* разнообразие врагов и препядствий;
* разруемость препядствий;

Рассмотрим недостатки данного приложения:

* запуск только на платформе Денди.

### Игровое приложение «Реалистичный симулятр танковых сражений»

Программное средство, изображённое на рисунке 1.2, сохраняет привычный гемплей танчиков, превнося в него 3D.



Рисунок 1.2 – Игровое приложение «Реалистический симулятр танковых сражений»

Список выполняемых функций:

* управление танком;
* сражение с ботами;
* Растановка танков на карте.

Рассмотрим достоинства данного приложения:

* 3D графика;
* использование браузера, что позволяет запускать игровое приложение

на любом устройстве.

Рассмотрим недостатки данного приложения:

* невозможность играть при отсутствии подключения к интернету;
* низкая производительность.

## Перечень функциональных требований

После анализа аналогов был составлен следующий список выполняемых функций:

* управление танком;
* перезапуск игры;
* перезарядка танка;
* отрисовка карты;
* режим мультиплеера.

Для разработки программного средства будет использоваться язык программирования C# и среда разработки Visual Studio 2022, а также Windows Form .

Windows Form используется с целью создания графического интерфейса.

## Состав и параметры технических и программных средств

Приложение должно функционировать на персональных компьютерах со следующими характеристиками:

* процессор Intel Celeron N4020 или лучше;
* оперативная память 128 MB или лучше;
* свободное место на диске в размере 3 мегабайта или больше.

Приложение должно функционировать в окружении операционной системы Windows XP или новее.

В данном разделе указаны минимальные технические требования для запуска программного средства. Для эксплуатации в реальных могут потребоваться более мощные технические средства. Программное средство должно корректно функционировать и на более мощном оборудовании.

# Моделирование предметной области

В качестве сетевого протокала используется протокол UDP.

## Протокол UDP

Протокол UDP (User Datagram Protocol) - это простой протокол передачи данных в компьютерных сетях, который не гарантирует доставку пакетов данных и не устанавливает соединение между устройствами.

Протокол UDP используется для передачи данных, которые не требуют гарантированной доставки или которые могут быть повторно переданы в случае потери. Это может включать потоковую передачу данных, мультимедийные данные и другие типы данных, для которых короткие задержки важнее гарантированной доставки.

UDP-пакет состоит из заголовка и данных. Заголовок включает в себя порты отправителя и получателя, длину пакета и контрольную сумму. UDP не предоставляет механизмов контроля перегрузки и управления потоком, поэтому для этого обычно используются другие протоколы, например, TCP.

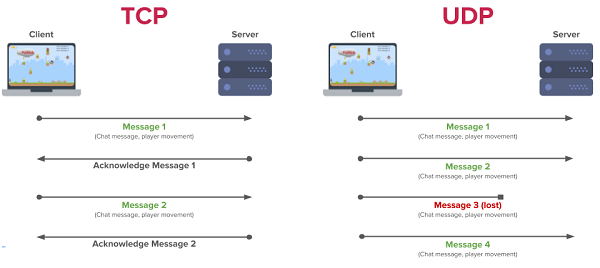


Рисунок 2.3 – Сравнение TCP и UDP

# Проектирование программного средства

## Структура программы

При разработке приложения будет использовано 9 модулей:

* MainWindow – модуль графического вывода и обработки сообщений

отправленыных окну

* Logic – модуль ии и обработки взаимодействия объектов
* Shell – модуль подпрограмм и данных снарядов
* BaseTank – модуль отвечающий за отрисовку таноков.
* Tank – модуль подпрограмм и данных танков
* Wall – модуль подпрограмм и данных стен
* UdpClient – модуль клиента.
* UdpServer – модуль сервера.
* Const – модуль хранит общие для всей программы типы и

константы.

## Проектирование интерфейса программного средства

### Главное меню

Лаунчер (главное меню приложения) должен состоять из простого окна с 3 кнопками Сервер, Клиент и Выход. Макет главного окна приложения представлен на рисунке 3.1.



Рисунок 3.1 – Главное меню

## Проектирование функционала программного средства

При создании игры очень важно сразу определить задачу и цели, а также хорошо составить рабочие алгоритмы. Искать ошибки в коде придётся в любом случае, но хорошо написанный алгоритм упрощает это в разы. В игре должны быть использованы следующие алгоритмы:

* обработчик нажитий пользователя;
* обработчик поведения ИИ;
* схема работы лаунчера;

### Обработчик нажатий пользователя

Обработчик нажатий пользователя необходим, чтобы позволять пользователю осуществлять управление танком.. Блок-схема алгоритма данной функции приведена на рисунке 3.2.



Рисунок 3.2 – Обработчик нажатий пользователя

### Обработчик повеведия ИИ

Обработчик повеведия ИИ необходим, чтобы организовать самостоятельное движение и выстрелы со стороны противников. При создании будет использоваться процедура Game, вынесенная в отдельный поток. Блок-схема алгоритма данной функции приведена на рисунке 3.3.



Рисунок 3.3 – Обработчик поведения ИИ

### Проверка попадания снаряда

Проверка попадания снаряда необходимо, чтобы избежать прохождения снарядов через объекты и их правильное движение по карте. Блок-схема алгоритма данной функции приведена на рисунке 3.4.



Рисунок 3.4 – Проверка попадания снаряда

# Разработка программного средства

## Проверка столкновения танка

Основой танка является алгоритм проверки столкновения с объектами.

Вызов необходимой процедуры осуществляется каждый раз, когда происходит обработка игровой логики. Сперва происходит проверка со стенами и выходы за границы поля, потом происходит проверка на столкновение с другим танком. Код процедуры будет предоставлен ниже.

private void CheckTankCollision(Tank selectTank,

Const.Direction direction)

{

bool move = true;

// Проверка на столкновение с преградами (стенами)

foreach (Wall wall in listWalls)

{

if (CheckCollision(selectTank.X1,

selectTank.Y1, selectTank.X2,

selectTank.Y2, direction, wall.X1, wall.Y1,

wall.X2, wall.Y2))

{

// Если произошло столкновение, остановить

движение

move = false;

break;

}

}

// Проверка на столкновение с другими танками

foreach (Tank tank in listTanks)

{

if (tank != selectTank && CheckCollision(

selectTank.X1, selectTank.Y1,

selectTank.X2, selectTank.Y2, direction,

tank.X1, tank.Y1, tank.X2, tank.Y2))

{

// Если произошло столкновение, остановить

движение

move = false;

break;

}

}

// Если нет столкновений, переместить танк

if (move)

{

selectTank.Move(direction);

}

}

## Движение танка

После проверки столкновений необходимо осуществить движение танка.

Сначала танк поворачивает и движется в указаную сторону, потом происходит изменение значений танка. Код функций будет предоставлен ниже.

public void Move(Const.Direction direction)

{

// Изменение координат танка в зависимости

от направления

switch (direction)

{

case Const.Direction.UP:

Y -= speed;

break;

case Const.Direction.RIGHT:

X += speed;

break;

case Const.Direction.DOWN:

Y += speed;

break;

case Const.Direction.LEFT:

X -= speed;

break;

}

// Обновление текущего направления танка

this.Direction = direction;

// Изменение состояния переменной FirstDark

для анимации гусиниц танка

FirstDark = !FirstDark;

tankGraphis.FirstDark = FirstDark;

// Обновление границ хитбокса танка

X1 = X - SIZE\_HITBOX;

X2 = X + SIZE\_HITBOX;

Y1 = Y - SIZE\_HITBOX;

Y2 = Y + SIZE\_HITBOX;

}

## Создание снаряда

Для возможности прохождения игры необходима возможность уничтожения противников или игрока.

Сначала происходит получнение параметров из танка, потом запись положения снаряда. Код данной функции будет приведён ниже.

public Shell(int x, int y, Const.Direction direction,

int damage, int speed)

{

// Установка урона, направления и скорости снаряда

this.Damage = damage;

this.Direction = direction;

this.Speed = speed \* 2;

// Установка начальных координат снаряда в

зависимости от направления

switch (direction)

{

case Const.Direction.UP:

this.X = x;

this.Y = y - 25;

break;

case Const.Direction.RIGHT:

this.X = x + 25;

this.Y = y;

break;

case Const.Direction.DOWN:

this.X = x;

this.Y = y + 25;

break;

case Const.Direction.LEFT:

this.X = x - 25;

this.Y = y;

break;

}

}

# Тестирование программного средства

# Руководство пользователя

# Заключение

# Список использованной литературы

Приложение А

(обязательное)