**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное**

**учреждение высшего образования**

**ПЕТРОЗАВОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Физико-технический институт**

Дисциплина «Технология программирования»

Отчет по командному проекту:

«Написание 2D-игры на языке С++ с использованием библиотеки SFML»

Выполнили:

Студенты 3 курса ФТИ группы 21317

**Тужикова Елена Эдуардовна**

**Власов Андрей Андреевич**

Проверил:

кандидат физико-математических наук

**Бульба Артем Владимирович**

Петрозаводск 2018

**ЦЕЛЬ РАБОТЫ**

Командный проект по созданию простой 2D игры с использованием SFML (Simple and Fast Multimedia Library) на языке С++ в Visual Studio.

**ЗАДАЧИ РАБОТЫ**

1. Командная работа.
2. Научиться взаимодействовать с командой в системе контроля версий Git, создавая репозиторий, ветки, фиксируя изменения. Объединять отдельные части программы, созданные членами команды, воедино.
3. Научиться строить UML - диаграммы с использованием CASE - системы WhiteStarUML (диаграмма классов (class diagram) и диаграмма вариантов использования (use case diagram) разработанной программы).
4. Написать простую 2D - игру на языке C++ с использованием SFML.
5. Составить отчет по проделанной работе.

**ТРЕБОВАНИЯ К ПРОГРАММЕ**

1. Программа не должна вызывать системных сбоев или зависать.

2. Программа должна освобождать всю выделенную память.

3. Должны отсутствовать конструкции, приводящие к неопределенному поведению.

4. Не должно быть неиспользуемых переменных.

5. Алгоритмы не должны быть избыточны (нет циклов, массивов, ... без которых можно обойтись).

6. Свои классы необходимо реализовывать в отдельных файлах.

7. Необходимо отделять интерфейс от реализации.

8. Необходимо использовать STL-контейнеры и итераторы.

**СЮЖЕТ ИГРЫ**

Игра «Space dodger» - это игра, где необходимо уворачиваться от летящих пуль и врагов. Главный герой – это самолет. Его противники – враждебные самолеты, выпускающие свои пули. Каждое соприкосновение игрока с недружественными объектами отнимает жизнь. По истечению всех жизней игра заканчивается.

**ИСПОЛЬЗОВАННЫЕ ПРОГРАММНЫЕ СРЕДСТВА**

Каждый из членов команды использовал среду разработки приложений Visual Studio 2017 с подключенной библиотекой SFML последней версии.

**СТРУКТУРА ПРОГРАММЫ**

Код программы написан на языке С++ и удовлетворяет принципам объектно-ориентированного подхода (инкапсуляция, наследование, полиморфизм), присутствует раздельная компиляция.

**Заголовочные файлы программы:**

1. **<SFML/Graphics.hpp>** - работа с графикой. SFML.
2. **"map.h" -** интерфейс класса карты.(map.cpp)
3. **"globals.h" -** обеспечение взаимодействия объектов с картой, вызов из других классов. Содержит extern и static const.
4. **"player.h" -** интерфейсная часть класса игрока.(player.cpp)
5. **"Enemy.h" -** интерфейсная часть классов врага (легких и сложных противников). (Enemy.cpp)

**ГЛАВНЫЙ ФАЙЛ ИСХОДНОГО КОДА**

Главный файл исходного кода(main.cpp) включает в себя:

1. Подключение необходимых заголовочных файлов:

#include <iostream>

#include <SFML/Graphics.hpp>

#include "map.h"

#include "globals.h"

#include "player.h"

#include "Enemy.h"

#include <list>

1. Тексты, при завершении игры

Text gameOverText;

gameOverText.setFont(font);

gameOverText.setCharacterSize(50);

gameOverText.setFillColor(Color::Red);

gameOverText.setPosition(100.f, window.getSize().y / 2);

gameOverText.setString("GAME OVER!");

Text gameWinText;

gameWinText.setFont(font);

gameWinText.setCharacterSize(50);

gameWinText.setFillColor(Color::Red);

gameWinText.setPosition(100.f, window.getSize().y / 2);

gameWinText.setString("GRAC, YOU WIN");

1. Объекты соответствующих классов:

Map drawing;

Player z("hero\_d.png", 48, 48, 23.0, 23.0);

Map m;

1. Списки и соответствующие итераторы:

А) Противников

std::list<Enemy\*> Vrag;

std::list<Enemy\*>::iterator ee;

Б) пуль

std::list<Bullet\*> bullets;

std::list<Bullet\*>::iterator bul;

1. Вывод врагов на карту:

for (int i = 0; i < HEIGHT\_MAP; i++) {

for (int j = 0; j < WIDTH\_MAP; j++) {

if (TileMap[i][j] == 'L') {

Vrag.push\_back(new EasyE(j, i));

}

if (TileMap[i][j] == 'P') {

Vrag.push\_back(new HardE(j, i));

}

6. Функции прорисовки и взаимодействия объектов:

m.drawing();

z.draw\_p();

for (ee = Vrag.begin(); ee != Vrag.end(); ee++) {

if ((\*ee)->life) {

(\*ee)->drawing(z.getplayercoordinateX(),z.getplayercoordinateY(), z);

if ((\*ee)->shoot > 200) {

(\*ee)->shoot = 0;

FloatRect polozh = (\*ee)->polozenie();

float uskor = (\*ee)->uskorenie();

bullets.push\_back(new Bullet(polozh, uskor));

}

(\*ee)->shoot++;

}

}

}

for (bul = bullets.begin(); bul != bullets.end(); bul++) {

if ((\*bul)->life) {

(\*bul)->WithPlayer(z);

(\*bul)->drawing();

}

7.Удаление пули и врагов при взаимодействии со стеной/игроком:

else {

delete((\*bul));

bullets.remove(\*bul);

break;

}

}

else {

delete((\*ee));

Vrag.remove(\*ee);

break;

**ЗАКЛЮЧЕНИЕ**

При командной работе была создана простая 2D-игра в жанре головоломка на языке С++ в среде разработке приложений Visual Studio 2017 с использованием SFML библиотек.

При создании программы все участники команды руководствовались принципами ООП и раздельной компиляцией.

В коде программы реализована очистка памяти, отсутствуют неиспользуемые переменные.

Были использованы STL-контейнеры и итераторы.

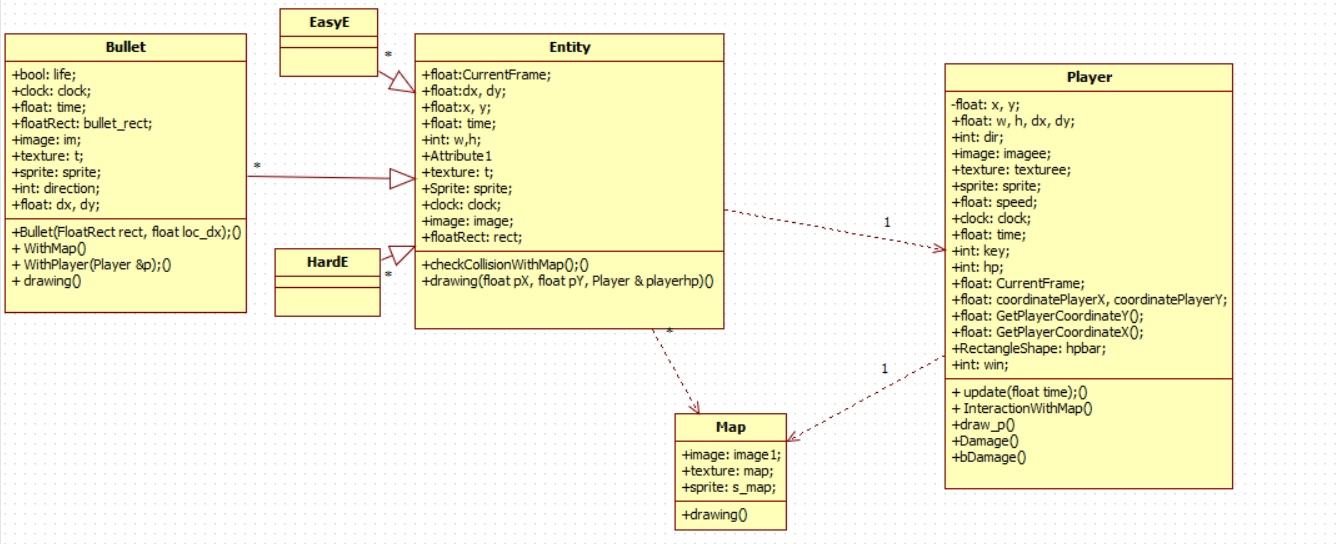
Программа удовлетворяет поставленным требованиям к проекту.

Сбоев или зависаний при запуске программы не выявлено.

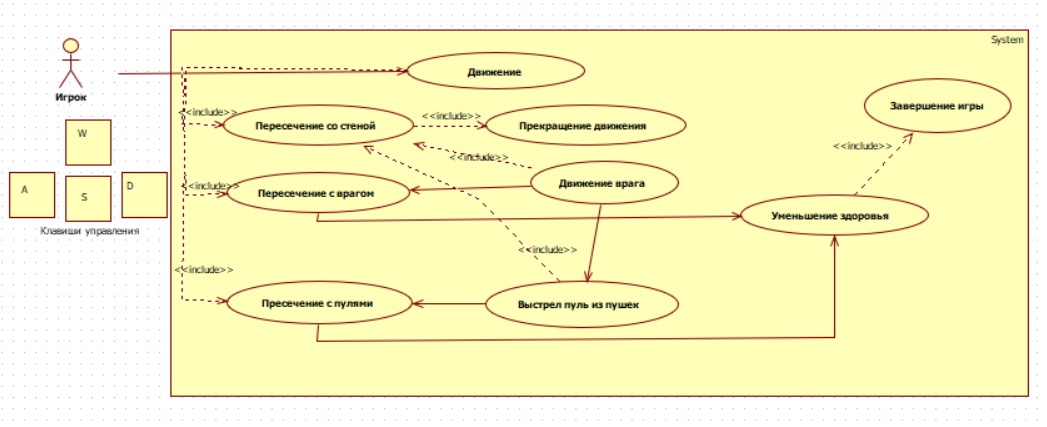
Цель работы достигнута – игра исправно работает.

ПРИЛОЖЕНИЯ

**Приложение 1.** UML-диаграмма классов



**Приложение 2.** Диаграмма вариантов использования



**Приложение 3.** Листинг главного кода программы main.cpp

#include <iostream>

#include <SFML/Graphics.hpp>

#include "map.h"

#include "globals.h"

#include "player.h"

#include "Enemy.h"

#include <list>

using namespace std;

using namespace sf;

int main()

{

Map drawing;

Player z("ss.png", 128, 128, 32, 32);

Map m;

Clock clock;

window.clear();

Font font;

font.loadFromFile("Dosis-Light.ttf");

Text gameOverText;

gameOverText.setFont(font);

gameOverText.setCharacterSize(50);

gameOverText.setFillColor(Color::Red);

gameOverText.setPosition(100.f, window.getSize().y / 2);

gameOverText.setString("GAME OVER!");

Text gameWinText;

gameWinText.setFont(font);

gameWinText.setCharacterSize(50);

gameWinText.setFillColor(Color::Red);

gameWinText.setPosition(100.f, window.getSize().y / 2);

gameWinText.setString("GRAC, YOU WIN");

std::list<Enemy\*> Vrag;

std::list<Enemy\*>::iterator ee;

std::list<Bullet\*> bullets;

std::list<Bullet\*>::iterator bul;

while (window.isOpen())

{

for (int i = 0; i < HEIGHT\_MAP; i++) {

for (int j = 0; j < WIDTH\_MAP; j++) {

if (TileMap[i][j] == 'L') {

Vrag.push\_back(new EasyE(j, i))

}

if (TileMap[i][j] == 'P') {

Vrag.push\_back(new HardE(j, i));

}

sf::Event event;

while (window.pollEvent(event))

{

if (event.type == sf::Event::Closed)

window.close();

}

if (z.hp > 0) {

m.drawing();

z.draw\_p();

for (ee = Vrag.begin(); ee != Vrag.end(); ee++) {

if ((\*ee)->life) {

(\*ee)->drawing(z.getplayercoordinateX(), z.getplayercoordinateY(), z);

if ((\*ee)->shoot > 200) {

(\*ee)->shoot = 0;

FloatRect polozh = (\*ee)->polozenie();

float uskor = (\*ee)->uskorenie();

bullets.push\_back(new Bullet(polozh, uskor));

}

(\*ee)->shoot++;

}

else {

delete((\*ee));

Vrag.remove(\*ee);

break;

}

}

for (bul = bullets.begin(); bul != bullets.end(); bul++) {

if ((\*bul)->life) {

(\*bul)->WithPlayer(z);

(\*bul)->drawing();

}

else {

delete((\*bul));

bullets.remove(\*bul);

break;

}

}

if (z.hp <= 0)

window.draw(gameOverText);

if (z.win >= 1)

window.draw(gameWinText);

window.draw(z.hpbar);

window.display();

}

}

}

}

return 0;