Московский авиационный институт (национальный исследовательский университет)
Институт №3.
«Системы управления, информатика и электроэнергетика»
Кафедра №304
«Автоматизированные системы обработки информации и управления»

Пояснительная записка к курсовой работе
по учебной дисциплине:
«Объектно-ориентированное программирование»

Группа М30-207Б
Выполнил:
Гордеев Н.М.

Принял:
Чечиков Ю.Б.

МОСКВА 2020

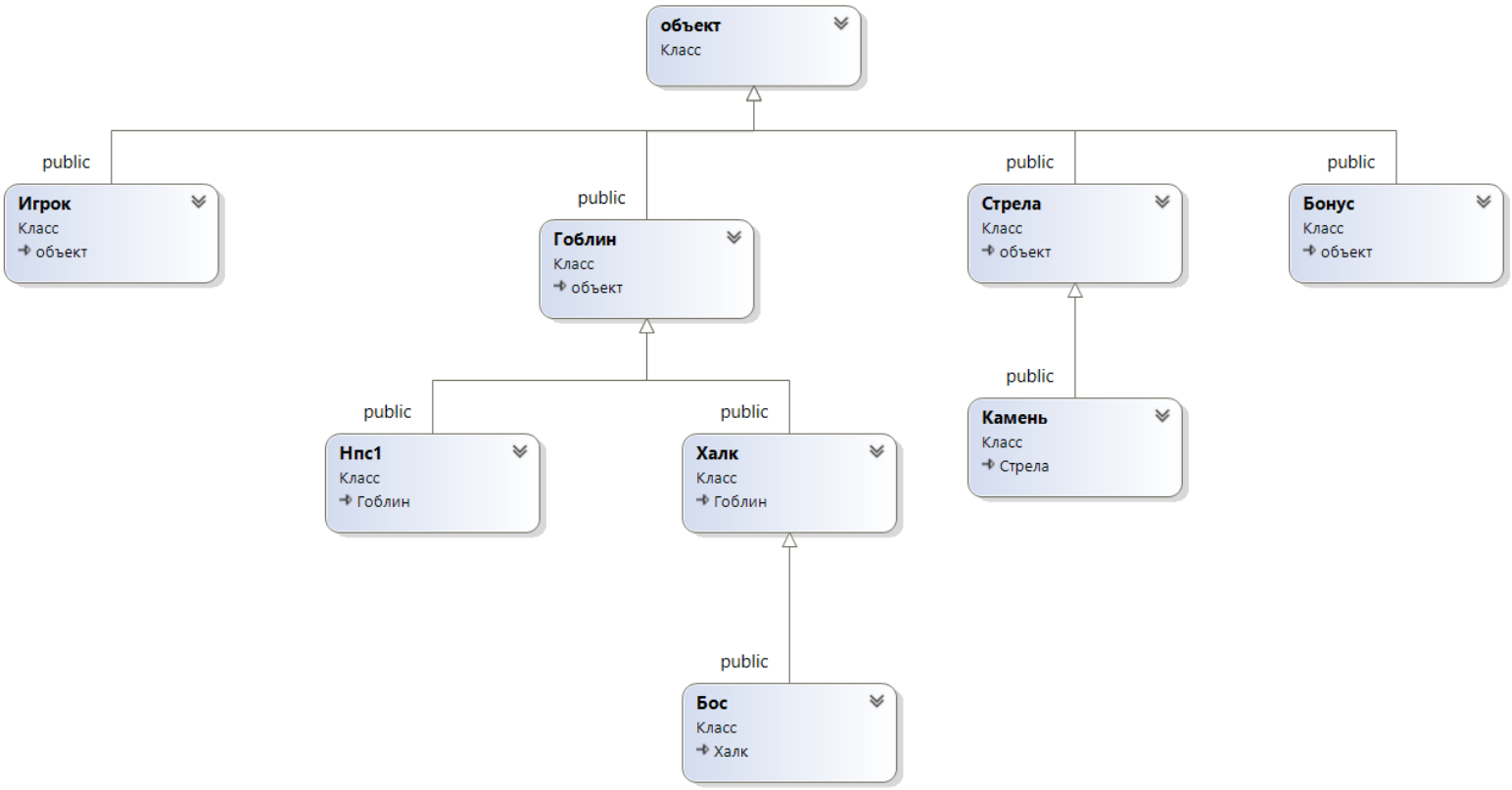
Содержание

Задание 3
Диаграмма классов 4
Классы в порядке наследования....5
Реализация методов классов 8
Основная программа 22
Вывод....43

Задание

Разработать игру в жанре 2D платформера, организовать веерное и вертикальное наследование классов. Разработать абстрактный класс для реализации множественного наследования (чистые виртуальные функции). Для подтверждения правильной реализации динамического полиморфизма использовать вызов виртуальных функций наследников через указатель базового класса.

Диаграмма классов



Классы в порядке наследования

```
#ifndef _МояИгра_H
#define _МояИгра_H
#include <SFML/Graphics.hpp>      //аудио
#include <SFML/Audio.hpp>         //графика
#include <sstream>                //работа с текстом
#include <random>                 //случайные числа
#include <list>                   //массив объектов
//скаченые библиотеки совместимости с программами SpriteDecomposer и Tiled
#include "Анимация.h"
#include "level.hpp"
using namespace sf;

static int Клетка = 100;   //размер 1 клетки в пикселях

//обстрактный базовый класс
class объект
{
protected:
	std::vector<Object> обьект;//объекты, с которыми класс будит взаимодействовать
(стены, бонусы)
	std::string имя;          //имя объекта, нужно чтобы различать их в массиве
	bool выстрел = 0;         //флаг состояния выстрела
	bool смерть = 0;          //флаг состояния удара
	int кадуд;                //сколько кадров в анимации удара
	std::string текст;        //сообщения для вывода над персонажем
	bool жив, зеркало;        //флаг жизни и поворота
	bool приследование = 0;   //флаг состояния преследования персонажа
	double Жизнь, урон;
	bool t = 1; bool уд = 1;  //вспомогательные флаги для анимации
	double x, y, dx, dy, w, h; //размеры и скорость объекта
	менеджер_анимации анимация;//анимация
public:
	//доступ к переменным
	менеджер_анимации пок_анимация() { return анимация; };
	void изм_Жизнь(double x) { Жизнь = x; };
	double пок_Жизнь() { return Жизнь; };
	double пок_урон() { return урон; };
	int пок_кадуд() { return кадуд; };
	bool пок_уд() { return уд; };
	void изм_уд() { уд = 0; };
	bool пок_t() { return t; };
	void изм_t(bool x) { t = x; };
	bool пок_приследование() { return приследование; };
	void изм_приследование(bool x) { приследование = x; };
	bool пок_жив() { return жив; };
	void изм_жив(bool x) { жив = x; };
	bool пок_зеркало() { return зеркало; };
	void изм_зеркало(bool x) { зеркало = x; };
	std::string пок_имя() { return имя; };
	std::string пок_текст() { return текст; };
	bool пок_выстрел() { return выстрел; };
	void сдел_выстрел() { выстрел = 1; };
	double пок_h() { return h; };
	double пок_x() { return x; };
	void изм_x(double x) { x = x; };
	double пок_y() { return y; };
	void изм_y(double x) { y = x; };
	//основные функции
	объект(менеджер_анимации a, int X, int Y); //конструктор
	FloatRect размер(); //возращает размеры персонажа
	virtual void показать(RenderWindow& window); //отобразить персонажа
	virtual void option(std::string NAME, double SPEED = 0, double жизнь = 1,
std::string FIRST_ANIM = ""); //инициализовать начальные характеристики
```

```
	virtual void обновить(double время) = 0; //основная ф-ия, обновляет персонажа
};

class Игрок : public объект
{
private:
	int ЗапасСтрел = 10;			//сколько осталось выстрелов
	double скорасть_игрока = 0.3;		//скорость игрока пешком
	FloatRect позиция;			//координаты
	bool на_земле = 0;			//нужно ли предавать ускорение падения
	Sprite игрок;				//картинка
public:
	//доступ к переменным
	int пок_ЗапасСтрел() { return ЗапасСтрел; };
	void изм_ЗапасСтрел(int x) { ЗапасСтрел = x; };
	void доб_ЗапасСтрел(int x) { ЗапасСтрел += x; };
	//основные функции
	Игрок(менеджер_анимации& анимация, Level& lvl, int x, int y); //конструктор
	void Анимация(double time); //анимация
	void НачПозиц(char буква); //чтение начальной позиции
	void управление();		//управление персонажем
	void обновить(double время); //основная ф-ия, обновляет персонажа
	bool столкновение(bool ось); //взаимоотношение со стенами
};

class Стрела : public объект {
public:
	Стрела(менеджер_анимации& a, Level& lev, int x, int y, bool dir); //свой
конструктор
	void обновить(double time); //основная ф-ия, обновляет персонажа
	Стрела(менеджер_анимации& a, Level& lev, int x, int y) : объект(a, x, y) {}
//конструктор для связи потомков с предками
};

class Камень : public Стрела {
public:
	Камень(менеджер_анимации& a, Level& lev, int x, int y, bool dir); //конструктор
};

class Бонус : public объект {
public:
	Бонус(std::string NAME, менеджер_анимации& a, int x, int y) : объект(a, x, y)
{//конструктор
		option(NAME); //загрузка основных параметров
	}
	void обновить(double time) {}; //обновление не требуется
};

class Гоблин : public объект {
protected:
	bool выстрел2 = 0;		//у наследников много типов ударов, но все пользуются
обновлением гоблина
	bool направление = 0;		//куда двигаться
	int смер = 0;			//нужен чтобы загружать разные по размерам картинки
смерти персонажа
	bool f = 1;			//флаг для проигрывания анимация смерти

	double скорость = 0.3;		//стартовая скорость
	bool на_земле = 0;		//нужно ли предавать ускорение падения

public:
	//доступ к переменным
	void изм_выстрел2(bool x) { выстрел2 = x; };
	//основные функции
	virtual void поведение();//что делать после встречи с припядствием
```

```
        virtual void Анимация(double time);//анимация персонажа
        Гоблин(менеджер_анимации& анимация, Level& lvl, int x, int y, double SPEED, double
жизнь); //конструктор
        Гоблин(менеджер_анимации& a, Level& lev, int x, int y) : объект(a, x, y) {}
//связь наследников с предками
        void обновить(double время); //основная ф-ия, обновляет персонажа
        bool столкновение(bool ось); //взаимоотношение со стенами
};

class Нпс1 : public Гоблин {
protected:
        int удар = 0; //переключатель типов ударов
public:
        virtual void Анимация(double time); //проигрывать анимацию
        Нпс1(менеджер_анимации& анимация, Level& lvl, int x, int y); //конструктор
};

class Халк : public Гоблин {
protected:
        bool прыжок = 0; //флаг начала прыжка
        //флаги для анимацией
        bool t4 = 1;
        bool t3 = 0;
        double t2; //запоминание позиции чтобы сменить поведение при повторении пути
        double выспрышеп = 0.6; //ускорение при прыжке
        int удар = 0; // который удар проигрывать

public:
        //доступ к переменным
        int пок_удар() { return удар; };
        //основные функции
        virtual void Анимация(double time); //проигрывать анимацию
        void поведение(); //что делать при встречи со стеной
        Халк(менеджер_анимации& анимация, Level& lvl, int x, int y); //конструктор
        Халк(менеджер_анимации& анимация, Level& lvl, int x, int y, int t) :
Гоблин(анимация, lvl, x, y) {} //конструктор для связи наследников с предками
};

class Бос : public Халк {
private:
        double прыжокустены; //запоминание позиции чтобы сменить поведение при повторении
пути
public:
        //доступ к переменным
        double пок_прыжокустены() { return прыжокустены; };
        //основные функции
        void Анимация(double time); //проигрыванее анимации
        Бос(менеджер_анимации& анимация, Level& lvl, int x, int y); //конструктор
};
#endif
```

Реализация методов классов

```
#include "заголовок.h"

объект::объект(менеджер_анимации a, int X, int Y)
{
	анимация = a;
	x = X;
	y = Y;
	dx = dy = 0;
	жив = true;
	зеркало = false;
}

Стрела::Стрела(менеджер_анимации& a, Level& lev, int x, int y, bool dir) : объект(a, x, y) {
	option("Bullet", 1.8, 1, "огонь");
	урон = 1;
	if (dir) dx = -1.8;
	обьект = lev.GetObjects("solid");
}

Камень::Камень(менеджер_анимации& a, Level& lev, int x, int y, bool dir) : Стрела(a, lev, x, y) {
	option("cam", 1.8, 1, "огонь");
	урон = 0.2;
	if (dir) dx = -1.8;
	обьект = lev.GetObjects("solid");
}

Гоблин::Гоблин(менеджер_анимации& a, Level& lev, int x, int y, double SPEED, double жизнь = 1) :
объект(a, x, y)
{
	option("goblin", SPEED, жизнь, "идет");
	обьект = lev.GetAllObjects();
	урон = 0.2;
	скорость = SPEED;
	кадуд = 3;
}

Халк::Халк(менеджер_анимации& a, Level& lvl, int x, int y) : Гоблин(a, lvl, x, y)
{
	option("hl", 0.3, 3, "идет");
	обьект = lvl.GetAllObjects();
	урон = 0.2;
	скорость = 0.3;
	кадуд = 1;
}

Нпс1::Нпс1(менеджер_анимации& a, Level& lvl, int x, int y) : Гоблин(a, lvl, x, y)
{
	option("npc1", 0.3, 3, "бег");
```

```
	обьект = lvl.GetAllObjects();
	скорость = 0;
	текст = "спасибо";
}

Бос::Бос(менеджер_анимации& анимация, Level& lvl, int x, int y) : Халк(анимация, lvl, x, y, 1) {
	option("bos", 0.6, 10, "идет");
	обьект = lvl.GetAllObjects();
	урон = 0.2;
	скорость = 0.6;
	выспрышеп = 0.75;
	кадуд = 3;
}

Игрок::Игрок(менеджер_анимации& a, Level& lev, int x, int y) : объект(a, x, y)
{
	option("Player", 0, 1, "стоит");
	обьект = lev.GetAllObjects();
}

FloatRect объект::размер()
{
	return FloatRect(x, y, w, h);
}

void объект::показать(RenderWindow& window) {
	анимация.показать(window, x, y + h);
}

void объект::option(std::string NAME, double SPEED, double жизнь, std::string FIRST_ANIM)
{
	имя = NAME;
	if (FIRST_ANIM != "") анимация.set(FIRST_ANIM);
	w = анимация.шир();
	h = анимация.выс();
	dx = SPEED;
	Жизнь = жизнь;
}

void Стрела::обновить(double время) {
	x += dx * время;
	for (int i = 0; i < обьект.size(); i++)
		if (размер().intersects(обьект[i].rect))
			жив = false;
	if (dx < 0) анимация.зеркало();
	анимация.время(время);

}

void Гоблин::обновить(double время) {
```

```
	Анимация(время);

	зеркало ? dx = -скорость : dx = скорость;
	выстрел ? dx = 0 : выстрел2 ? dx = 0 : зеркало ? dx = -скорость : dx = скорость;

	if (Жизнь <= 0) dx = 0;
	x = x + dx * время;
	if (столкновение(0) == 1 && !приследование)
	{
		поведение();
	}
	if (!на_земле) dy = dy + 0.001 * время;
	y += 2 * dy * время;
	на_земле = 0;
	столкновение(1);
}

void Игрок::обновить(double время)
{
	управление();
	Анимация(время);
	позиция.left += dx * время;
	x += dx * время;
	столкновение(0);
	//падение
	if (!на_земле) dy = dy + 0.001 * время;
	позиция.top += 2 * dy * время;
	y += 2 * dy * время;

	на_земле = 0;
	столкновение(1);
	dx = 0;
}

void Гоблин::Анимация(double время)
{
	if (!выстрел && abs(dx) > 0) анимация.set("идет");
	if (выстрел) {
		анимация.set("удар");
	}

	if (выстрел && анимация.кад() >= анимация.колкад() - 1) {
		выстрел = 0; t = 1; уд = 1;
	}

	if (Жизнь <= 0) {
		if (смер == 0) {
			анимация.set("смерть1");
			if (f) {
				f = 0;
```

```
                        анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
                        смер = 1;
                }
        }

        if (смер == 1) {
                анимация.set("смерть1");
                if (f) {
                        f = 0;
                        анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
                        смер = 2;
                        направление = !направление;
                }
        }

        if (смер == 2) {
                анимация.set("смерть2");
                if (f) {
                        f = 0;
                        анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
                        смер = 3;

                }
        }

        if (смер == 3) {
                анимация.set("смерть3");
                if (f) {
                        f = 0;
                        анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
                        смер = 4;
                }
        }
        if (анимация.кад() >= анимация.колкад() - 1) {
                f = 1;
        }
        if (смер == 4) жив = 0;
    }

    if (dx < 0) направление = 0; else if (dx > 0) направление = 1;
    if (!направление) анимация.зеркало();
    анимация.время(время);
}

bool Гоблин::столкновение(bool ось)
{
    int столкновение = 0;
    for (int i = 0; i < обьект.size(); i++)
        if (размер().intersects(обьект[i].rect))
        {
```

```
                        if (обьект[i].name == "solid")
                        {
                                if ((dx > 0) && (ось == 0)) {
                                        x = обьект[i].rect.left - w; столкновение = 1;
                                }
                                if ((dx < 0) && (ось == 0)) {
                                        x = обьект[i].rect.left + обьект[i].rect.width; столкновение = 1;
                                }
                                if ((dy > 0) && (ось == 1)) {
                                        y = обьект[i].rect.top - h;  dy = 0;   на_земле = 1; столкновение
= 0;
                                }
                                if ((dy < 0) && (ось == 1)) {
                                        y = обьект[i].rect.top + обьект[i].rect.height;   dy = 0;
столкновение = 0;
                                }
                        }

                }
        return столкновение;
}

void Гоблин::поведение() {
        зеркало = !зеркало;
}

void Халк::Анимация(double время)
{
        if (t4 && !выстрел && abs(dx) > 0) анимация.set("идет");

        //удар
        if (dy == 0) {
                if (выстрел && удар % 7 == 0) {
                        анимация.set("удар");
                        if (t) {
                                t = 0;
                                анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
                                удар = 1;
                        }
                }

                if (выстрел && удар % 7 == 1) {
                        анимация.set("удар1");
                        if (t) {
                                удар = 2;
                                t = 0; анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
                        }
                }
```

```
		if (выстрел && удар % 7 == 2) {
			анимация.set("удар2");
			if (t) {
				удар = 3;
				t = 0; анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			}
		}


		if (выстрел && удар % 7 == 3) {
			анимация.set("удар3");
			if (t) {
				удар = 4;
				t = 0; анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			}
		}

		if (выстрел && удар % 7 == 4) {
			анимация.set("удар вверх");
			if (t) {
				удар = 5;
				t = 0; анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			}
		}

		if (выстрел && удар % 7 == 5) {
			анимация.set("удар вниз");
			if (t) {
				удар = 0;
				t = 0; анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			}
		}
	}
	if (выстрел && dy != 0) {
		анимация.set("удар в прышке");
		if (t) {
			t = 0; анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
		}
	}
	if (выстрел && анимация.кад() >= анимация.колкад() - 1) {
		выстрел = 0; t = 1; уд = 1;
	}

	//смерть
	if (Жизнь <= 0) {
		if (смер == 0) {
			анимация.set("смерть1");
			if (f) {
				f = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
```

```
                смер = 1;
            }
        }

        if (смер == 1) {
            анимация.set("смерть1");
            if (f) {
                f = 0;
                анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
                смер = 2;
            }
        }

        if (смер == 2) {
            анимация.set("смерть2");
            if (f) {
                f = 0;
                анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
                смер = 3;

            }
        }

        if (смер == 3) {
            анимация.set("смерть3");
            if (f) {
                f = 0;
                анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
                смер = 4;
            }
        }
        if (анимация.кад() >= анимация.колкад() - 1) {
            f = 1;
        }
        if (смер == 4) жив = 0;
}

//прыжок
{
        if (t4 && !выстрел && dy != 0 && прыжок == 1)
        {
            анимация.set("середина прыжка");
        }
        if (!выстрел && dy != 0 && прыжок == 0)
        {
            анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
            t4 = 0;
            прыжок = 1;
            анимация.set("начало прыжка");
        }
```

```
		if (!выстрел && dy == 0 && прыжок == 1)
		{
			анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			t4 = 0;
			прыжок = 0;
			анимация.set("приземление");
		}

		if (!t4 && анимация.кад() >= анимация.колкад() - 1) {
			t4 = 1;
		}
	}

	if (dx < 0) направление = 0; else if (dx > 0) направление = 1;
	if (!направление) анимация.зеркало();
	анимация.время(время);
}

void Халк::поведение() {

	if (t2 == x && t3 && на_земле)
	{
		t3 = 0;
		зеркало = !зеркало;
	}
	else if (на_земле)
	{
		dy = -выспрышеп;
		на_земле = 0;
		t3 = 1;
		t2 = x;
	}
}

void Нпс1::Анимация(double время)
{
	if (!выстрел && Жизнь) анимация.set("стоит");

	//удар
	if (dy == 0) {
		if (выстрел && удар % 2 == 0) {
			анимация.set("удар");
			if (t) {
				t = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				удар = 1;
			}
		}
```

```
		if (выстрел && удар % 2 == 1) {
			анимация.set("удар2");
			if (t) {
				удар = 0;
				t = 0; анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			}
		}
}
if (выстрел && анимация.кад() >= анимация.колкад() - 1) {
		выстрел = 0; t = 1;
}


//смерть
if (Жизнь <= 0) {
		if (смер == 0) {
			анимация.set("смерть1");
			if (f) {
				f = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				смер = 1;
			}
		}

		if (смер == 1) {
			анимация.set("смерть1");
			if (f) {
				f = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				смер = 2;
			}
		}

		if (смер == 2) {
			анимация.set("смерть2");
			if (f) {
				f = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				смер = 3;
			}
		}

		if (смер == 3) {
			анимация.set("смерть3");
			if (f) {
				f = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				смер = 4;
			}
		}
```

```
		if (смер == 4) {
			анимация.set("смерть4");
			if (f) {
				f = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				смер = 5;
			}
		}
		if (смер == 5) {
			анимация.set("смерть5");
			if (f) {
				f = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				смер = 6;
			}
		}
		if (смер == 6) {
			анимация.set("смерть6");
			if (f) {
				f = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				смер = 7;
			}
		}
		if (смер == 7) {
			анимация.set("смерть7");
			if (f) {
				f = 0;
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				смер = 8;
			}
		}
		if (анимация.кад() >= анимация.колкад() - 1) {
			f = 1;
		}
		if (смер == 8) жив = 0;
	}
	if (зеркало) анимация.зеркало();
	анимация.время(время);
}

void Бос::Анимация(double время)
{
	if (t4 && !выстрел && !выстрел2 && abs(dx) > 0) анимация.set("идет");

	//удар
	if (выстрел) {
		анимация.set("удар");
	}
```

```
if (выстрел2) {
	if (удар % 5 == 0) {
		анимация.set("камень1");
		if (t) {
			t = 0;
			анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			удар = 1;
		}
	}

	if (удар % 5 == 1) {
		анимация.set("камень2");
		if (t) {
			t = 0;
			анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			удар = 2;
		}
	}

	if (удар % 5 == 2) {
		анимация.set("камень3");
		if (t) {
			t = 0;
			анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			удар = 3;
		}
	}

	if (удар % 5 == 3) {
		анимация.set("камень4");
		if (t) {
			t = 0;
			анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			удар = 4;
		}
	}

	if (удар % 5 == 4) {
		анимация.set("камень5");
		if (t) {
			t = 0;
			анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
			удар = 0;
		}
	}
}
if (выстрел && анимация.кад() >= анимация.колкад() - 1) {
	выстрел = 0; t = 1; уд = 1;
}
if (выстрел2 && анимация.кад() >= анимация.колкад() - 1) {
```

```
			выстрел2 = 0; t = 1; прыжокустены = x;
		}

		//смерть
		if (Жизнь <= 0) {
			жив = 0;
		}


		//прыжок
		{
			if (t4 && !выстрел && !выстрел2 && dy != 0 && прыжок == 1)
			{
				анимация.set("середина прышка");
			}
			if (!выстрел && dy != 0 && прыжок == 0)
			{
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				t4 = 0;
				прыжок = 1;
				анимация.set("начало прышка");
			}

			if (!выстрел && !выстрел2 && dy == 0 && прыжок == 1)
			{
				анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
				t4 = 0;
				прыжок = 0;
				анимация.set("конец прышка");
			}

			if (!t4 && анимация.кад() >= анимация.колкад() - 1) {
				t4 = 1;
			}
		}

		if (dx < 0) направление = 0; else if (dx > 0) направление = 1;
		if (!направление) анимация.зеркало();
		анимация.время(время);
}

void Игрок::НачПозиц(char буква) {
		for (int i = 0; i < обьект.size(); i++)
			if (позиция.intersects(обьект[i].rect))
				if (обьект[i].name == "Player")
					позиция = обьект[i].rect;
}

void Игрок::управление() {
		SoundBuffer a; Sound ОзвВыстрела;
```

```
	a.loadFromFile("Озвучка/выстрел.ogg");
	ОзвВыстрела.setBuffer(a);
	if ((Keyboard::isKeyPressed(Keyboard::A)) || (Keyboard::isKeyPressed(Keyboard::Left)))
		if (!(выстрел && !dy)) dx = -скорасть_игрока * 2;
	if ((Keyboard::isKeyPressed(Keyboard::D)) || (Keyboard::isKeyPressed(Keyboard::Right)))
		if (!(выстрел && !dy)) dx = скорасть_игрока * 2;
	if ((Keyboard::isKeyPressed(Keyboard::A) || (Keyboard::isKeyPressed(Keyboard::Left))) &&
Keyboard::isKeyPressed(Keyboard::LShift))
		if (!(выстрел && !dy)) dx = -скорасть_игрока;
	if ((Keyboard::isKeyPressed(Keyboard::D) || (Keyboard::isKeyPressed(Keyboard::Right))) &&
Keyboard::isKeyPressed(Keyboard::LShift))
		if (!(выстрел && !dy)) dx = скорасть_игрока;
	if ((Keyboard::isKeyPressed(Keyboard::W)) || (Keyboard::isKeyPressed(Keyboard::Up)))
		if (на_земле && !выстрел)
		{
			dy = -0.6; на_земле = 0;
		}
	if (ЗапасСтрел && ((Mouse::isButtonPressed(Mouse::Left)) ||
(Keyboard::isKeyPressed(Keyboard::Space))))
		 if (!выстрел) { выстрел = 1; ОзвВыстрела.play(); }
}

void Игрок::Анимация(double time)
{
	if (!выстрел && dx == 0) анимация.set("стоит");
	if (abs(dx) > скорасть_игрока) анимация.set("бег");
	else if (!выстрел && abs(dx) > 0) анимация.set("идет");
	if (выстрел) {
		анимация.set("выстрел");
		if (t) {
			t = 0; анимация.АнимЛист[анимация.ан()].ТекКадр = 0;
		}
	}

	if (выстрел && анимация.кад() >= анимация.колкад() - 1) {
		выстрел = 0;t = 1; уд = 1;
	}

	if (dx < 0) зеркало = 1;
	if (dx > 0) зеркало = 0;
	if (зеркало) анимация.зеркало();
	анимация.время(time);
}

bool Игрок::столкновение(bool ось)
{
	int столкновение = 0;
	for (int i = 0; i < обьект.size(); i++)
		if (размер().intersects(обьект[i].rect))
		{
```

```
                        if (обьект[i].name == "solid")
                        {
                                if ((dx > 0) && (ось == 0)) {
                                        x = обьект[i].rect.left - w; столкновение = 1;
                                }
                                if ((dx < 0) && (ось == 0)) {
                                        x = обьект[i].rect.left + обьект[i].rect.width; столкновение = 1;
                                }
                                if ((dy > 0) && (ось == 1)) {
                                        y = обьект[i].rect.top - h;  dy = 0;   на_земле = 1; столкновение
= 0;
                                }
                                if ((dy < 0) && (ось == 1)) {
                                        y = обьект[i].rect.top + обьект[i].rect.height;   dy = 0;
столкновение = 0;
                                }
                        }

                }
        return столкновение;
}
```

Основная программа

```
#include "заголовок.h"

int main() {

	RenderWindow window(sf::VideoMode(1280, 720), L"Убийца гоблинов", Style::Close);
	//экран
	window.setMouseCursorVisible(0);	//нет курсора

	менеджер_анимации анимация, анимация2, анимация3, анимация4, анимация5, анимация6, анимация7, анимация8, анимация9, анимация10;
	Texture Картинка, Картинка2, Картинка3, Картинка4, Картинка5, Картинка6, Картинка7, Картинка8, Картинка9, Картинка10;

	Картинка.loadFromFile("Текстуры/персонажи/игрок.png");
	Картинка2.loadFromFile("Текстуры/персонажи/стрела.png");
	Картинка3.loadFromFile("Текстуры/персонажи/гоблин.png");
	Картинка4.loadFromFile("Текстуры/персонажи/стрелы.png");
	Картинка5.loadFromFile("Текстуры/персонажи/жизнь.png");
	Картинка6.loadFromFile("Текстуры/персонажи/чемпион_гоблин.png");
	Картинка7.loadFromFile("Текстуры/персонажи/халк.png");
	Картинка8.loadFromFile("Текстуры/персонажи/бос.png");
	Картинка9.loadFromFile("Текстуры/персонажи/камень.png");
	Картинка10.loadFromFile("Текстуры/персонажи/нпс1.png");

	анимация.ЗагрузитьАнимациюXML("Текстуры/анимация/игрок.xml", Картинка);
	анимация2.ЗагрузитьАнимациюXML("Текстуры/анимация/стрела.xml", Картинка2);
	анимация3.ЗагрузитьАнимациюXML("Текстуры/анимация/гоблин.xml", Картинка3);
	анимация4.Создать("стрелы", Картинка4, 0, 0, 75, 100, 1, 0);
	анимация5.Создать("жизнь", Картинка5, 0, 0, 50, 50, 1, 0);
	анимация6.ЗагрузитьАнимациюXML("Текстуры/анимация/чемпион_гоблин.xml", Картинка6);
	анимация7.ЗагрузитьАнимациюXML("Текстуры/анимация/халк.xml", Картинка7);
	анимация8.ЗагрузитьАнимациюXML("Текстуры/анимация/бос.xml", Картинка8);
	анимация9.ЗагрузитьАнимациюXML("Текстуры/анимация/камень.xml", Картинка9);
	анимация10.ЗагрузитьАнимациюXML("Текстуры/анимация/нпс1.xml", Картинка10);

	std::list<объект*> объекты2;
	std::list<объект*> объекты;				//лист обьектов
	std::list<объект*>::iterator it;   //итератор для листа
	Level lvl;
	lvl.LoadFromFile("карта.tmx");

	Object pl = lvl.GetObject("Player");
	Игрок игрок(анимация, lvl, pl.rect.left, pl.rect.top);

	std::vector<Object> g = lvl.GetObjects("goblin");
	for (int i = 0; i < g.size(); i++)
		объекты.push_back(new Гоблин(анимация3, lvl, g[i].rect.left, g[i].rect.top, 0.3,1));
```

```
	std::vector<Object> g1 = lvl.GetObjects("goblin1");
	for (int i = 0; i < g1.size(); i++)
		объекты.push_back(new Гоблин(анимация3, lvl, g1[i].rect.left, g1[i].rect.top, 0.4,1));

	std::vector<Object> g2 = lvl.GetObjects("goblin2");
	for (int i = 0; i < g2.size(); i++)
		объекты.push_back(new Гоблин(анимация3, lvl, g2[i].rect.left, g2[i].rect.top, 0.5,1));

	std::vector<Object> c = lvl.GetObjects("c");
	for (int i = 0; i < c.size(); i++)
		объекты.push_back(new Бонус("c", анимация4, c[i].rect.left, c[i].rect.top));

	std::vector<Object> hp = lvl.GetObjects("hp");
	for (int i = 0; i < hp.size(); i++)
		объекты.push_back(new Бонус("hp", анимация5, hp[i].rect.left, hp[i].rect.top));

	std::vector<Object> hg = lvl.GetObjects("hgoblin");
	for (int i = 0; i < hg.size(); i++)
		объекты.push_back(new Гоблин(анимация6, lvl, hg[i].rect.left, hg[i].rect.top, 0.3, 2));

	std::vector<Object> hl = lvl.GetObjects("hl");
	for (int i = 0; i < hl.size(); i++)
		объекты.push_back(new Халк(анимация7, lvl, hl[i].rect.left, hl[i].rect.top));

	std::vector<Object> npc1 = lvl.GetObjects("npc1");
	for (int i = 0; i < npc1.size(); i++)
		объекты.push_back(new Нпс1(анимация10, lvl, npc1[i].rect.left, npc1[i].rect.top));

	for (it = объекты.begin(); it != объекты.end(); it++)
	{
		объект* п = *it; объект* r = new Бонус("hp", анимация5, 1, 1);;
		if (п->пок_имя() == "c" || п->пок_имя() == "hp")
			r = new Бонус("hp", анимация5, 1, 1);
		if (п->пок_имя() == "goblin" || п->пок_имя() == "goblin1" || п->пок_имя() ==
"goblin2")
			r = new Гоблин(анимация3, lvl, 1, 1, 0.3,1);
		if (п->пок_имя() == "hl")
			r = new Халк(анимация3, lvl, 1, 1);
		if (п->пок_имя() == "npc1")
			r = new Нпс1(анимация10, lvl, 1, 1);
		*r = *п;
		объекты2.push_back(r);
	}

	Object bos = lvl.GetObject("bos");
	Бос бос(анимация8, lvl, bos.rect.left, bos.rect.top);

	//текст
	Font font;//шрифт
```

```
Text text("", font, 50);
Text text2("", font, 50);
Text text3("", font, 50);
Text text4("", font, 25);
Text text5("", font, 50);
Text text6("", font, 50);
font.loadFromFile("Текстуры/CyrilicOld.TTF");
text.setString("НАЖМИТЕ ЛЮБУЮ КЛАВИШУ");
text.setPosition(250, 600);

//фон  и мыш
Texture d, h;
Sprite мыш, фон, точка, ползунок;
h.loadFromFile("Текстуры/курсор.png");
мыш.setTexture(h);
мыш.setTextureRect(IntRect(0, 0, 20, 22));
фон.setTextureRect(IntRect(0, 0, 1280, 720));

//звуки
SoundBuffer a, b,cc,yr, cox;
Sound ОзвВыстрела, ОзвСмеха, ОзвУрона, ОзвУронаИгрока, ОзвСохронения;
a.loadFromFile("Озвучка/выстрел.ogg");
b.loadFromFile("Озвучка/смех2.ogg");
cc.loadFromFile("Озвучка/урон гоблину.ogg");
yr.loadFromFile("Озвучка/урон.ogg");
cox.loadFromFile("Озвучка/Сохронение.ogg");
ОзвВыстрела.setBuffer(a);
ОзвСмеха.setBuffer(b);
ОзвУрона.setBuffer(cc);
ОзвУронаИгрока.setBuffer(yr);
ОзвСохронения.setBuffer(cox);

//музыка
Music МузыкаЗаставка, МузыкаБос, МузыкаВИгре, победа;
МузыкаЗаставка.openFromFile("Озвучка/Ready for battle (XCOM 2 OST).ogg");
МузыкаБос.openFromFile("Озвучка/Motoi Sakuraba_-_Looking Glass Knight (Dark Souls II
OST).ogg");
МузыкаВИгре.openFromFile("Озвучка/Motoi Sakuraba_-_Bell Gargoyle (Dark Souls OST).ogg");
победа.openFromFile("Озвучка/победа.ogg");
МузыкаЗаставка.play();
МузыкаВИгре.setLoop(1);
МузыкаБос.setLoop(1);
МузыкаЗаставка.setLoop(1);


//вспомогательные переменные
int система = 0; int вссистема = 0; int ОсталосьГоблинов = 57; int запасстрел = 10;
bool пауза; bool БойСБосом = 0;
bool Нажатие = 0;
bool соб = 0;
```

```
	bool cox = 0;
	bool cox1 = 0;
	bool cox2 = 0;
	bool cox3 = 0;
	bool cox4 = 0;
	Clock clok;
	double время;
	double Жизнь = 1;
	double x = 1100;
	double y = 1548;
	int ГромкостьМузыки = 50;
	int ГромкостьЗвуков = 100;
	int п1 = 578;
	int п2 = 578;
	время = clok.getElapsedTime().asMilliseconds();

	Event event;//датчик событий
	while (window.isOpen())
	{
		//заставка
		if (система == 0) {
			d.loadFromFile("Текстуры/maxresdefault.jpg");
			фон.setTexture(d);
			while (window.pollEvent(event))//обработка событий
			{
				соб = 1;
				text.setString("СТАРТ");
				text.setPosition(560, 600);
				text2.setString("На местную деревню напали гоблины,\nпохители
девушек и припасы,\nвас попросили разобраться с ними.");
				text2.setPosition(0, 0);
				//text3.setString("По возможности \nспасите жителей\nи других
\nавантюристов");
				//text3.setPosition(0, 200);
				text5.setString("НАСТРОЙКИ");
				text5.setPosition(500, 650);
				//отслеживание мыши
				if (event.type == sf::Event::MouseMoved)
					мыш.setPosition(event.mouseMove.x, event.mouseMove.y);

				//настройки текста
				text.setFillColor(Color::Black);
				text.setStyle(sf::Text::Bold);
				text2.setFillColor(Color::Red);
				//text3.setFillColor(Color::Green);
				text5.setFillColor(Color::Black);

				//закрываем окно
				if (event.type == sf::Event::Closed)
					window.close();
```

```
//подвили курсор к кнопке
if (Mouse::getPosition(window).x > 560 &&
Mouse::getPosition(window).x < 560 + 170 && \
Mouse::getPosition(window).y < 600 + 50 &&
Mouse::getPosition(window).y > 600)
{
text.setFillColor(Color::Red);//нажали на кнопку
if (Mouse::isButtonPressed(Mouse::Left))
{
window.create(VideoMode(), L"Убийца гоблинов",
Style::Fullscreen);
window.setMouseCursorVisible(0);
система = 1; МузыкаЗаставка.stop();
МузыкаВИгре.play();//воспроизводим музыку
}
}

//подвили курсор к настройкам
if (Mouse::getPosition(window).x > 500 &&
Mouse::getPosition(window).x < 500 + 300 && \
Mouse::getPosition(window).y < 650 + 50 &&
Mouse::getPosition(window).y > 650)
{
text5.setFillColor(Color::Red);//нажали на кнопку
if (Mouse::isButtonPressed(Mouse::Left))
{
система = 3; вссистема = 0;
}
}
}

//отображаем обьекты
window.draw(фон);
window.draw(text);
window.draw(text2);
//window.draw(text3);
window.draw(text5);
if (соб) {
window.draw(мыш);
}
window.display();
clok.restart();
}
if (система == 1)
{
View view(FloatRect(0, 0, 1920, 1080));
время = clok.getElapsedTime().asMilliseconds();
время *= 2;
if (время > 100)время = 100;
```

```
                        //случайные числа
                        std::random_device random_device;
                        std::mt19937 generator(random_device());
                        std::uniform_int_distribution<> distribution(0, 5);

                        //текст
                        std::ostringstream g, k;
                        g << ОсталосьГоблинов;
                        k << игрок.пок_ЗапасСтрел();
                        text.setString("Осталось гоблинов:" + g.str() + "\nХР\nСТРЕЛЫ:" + k.str());
                        text.setFillColor(Color::Black); text.setStyle(sf::Text::Bold);
                        text.setPosition(0, 0);
                        text2.setString("Жизнь босса");
                        text2.setFillColor(Color::Black); text2.setStyle(sf::Text::Bold);
                        text2.setPosition(1600, 0);
                        text3.setString("W A D или стрелки - бегать\n+Shift - ходить\n\neps-пауза и
меню\n\nстрелять на левую кнопку\nмыши или пробел\n");
                        text3.setFillColor(Color::Black);


                        //жизнь
                        игрок.пок_Жизнь() > 1 ? игрок.изм_Жизнь(1) :
игрок.изм_Жизнь(игрок.пок_Жизнь());
                        RectangleShape жизнь(Vector2f(игрок.пок_Жизнь() * 300, 30));
                        жизнь.setFillColor(Color::Red);
                        жизнь.setPosition(75, 60);

                        //жизнь босса
                        RectangleShape Босжизнь(Vector2f(бос.пок_Жизнь() * 30, 30));
                        Босжизнь.setFillColor(Color::Red);
                        Босжизнь.setPosition(1600, 60);

                        //сохранение
                        if (   (!сох1 && ОсталосьГоблинов == 40)
                                || (!сох2 && ОсталосьГоблинов == 32)
                                || (!сох3 && ОсталосьГоблинов == 13)
                                || (!сох4 && ОсталосьГоблинов == 0))
                        {
                                if (ОсталосьГоблинов == 40)
                                        сох1 = 1;
                                if (ОсталосьГоблинов == 32)
                                        сох2 = 1;
                                if (ОсталосьГоблинов == 13)
                                        сох3 = 1;
                                if (ОсталосьГоблинов == 0)
                                        сох4 = 1;

                                for (it = объекты2.begin(); it != объекты2.end();)
                                {
```

```
                                        объект* п = *it;
                                        it = объекты2.erase(it);
                                        delete п;
                                }
                                for (it = объекты.begin(); it != объекты.end(); it++)
                                {
                                        объект* п = *it; объект* r = new Бонус("hp", анимация5, 1, 1);
                                        if (п->пок_имя() == "c" || п->пок_имя() == "hp")
                                                r = new Бонус("hp", анимация5, 1, 1);
                                        if (п->пок_имя() == "goblin" || п->пок_имя() == "goblin1" || п-
>пок_имя() == "goblin2")
                                                r = new Гоблин(анимация3, lvl, 1, 1, 0.3,1);
                                        if (п->пок_имя() == "hl")
                                                r = new Халк(анимация3, lvl, 1, 1);
                                        if(п->пок_имя() == "npc1")
                                                r = new Нпс1(анимация10, lvl, 1, 1);
                                        *r = *п;
                                        объекты2.push_back(r);
                                }
                                запасстрел = игрок.пок_ЗапасСтрел();
                                Жизнь = игрок.пок_Жизнь();
                                x = игрок.пок_x();
                                y = игрок.пок_y();
                                cox = 1;
                                ОзвСохронения.play();
                        }

                        //меню
                        if (Keyboard::isKeyPressed(Keyboard::Escape))
                        {
                                !БойСБосом ? МузыкаВИгре.pause() : МузыкаБос.pause();
МузыкаЗаставка.play();
                                система = 2; пауза = 1;
                                window.create(sf::VideoMode(1200, 628), L"Убийца гоблинов",
Style::Close);
                                window.setMouseCursorVisible(0);
                        }

                        //выход
                        while (window.pollEvent(event)) {
                                if (event.type == Event::Closed)
                                        window.close();
                        }

                        //стрелы
                        if (игрок.пок_уд() && игрок.пок_ЗапасСтрел() && игрок.пок_выстрел()) {
                                if (игрок.пок_анимация().кад() >= игрок.пок_анимация().колкад() - 2)
{
                                        объекты.push_back(new Стрела(анимация2, lvl, игрок.пок_x()
+ !игрок.пок_зеркало() * 77, игрок.пок_y() + 64, игрок.пок_зеркало()));
```

```
                                        игрок.доб_ЗапасСтрел(-1); игрок.изм_уд();
                                }
                                if (игрок.пок_анимация().кад() == 0)
                                ОзвВыстрела.play();
                        }

                        //камни
                        if (бос.пок_удар() == 3) {
                                бос.изм_t(1);
                                объекты.push_back(new Камень(анимация9, lvl, бос.пок_x(),
бос.пок_y() + бос.пок_h() / 2, бос.пок_зеркало()));
                                //игрок.ЗапасСтрел -= 1;
                        }

                        //взаимоотношения с объектами
                        ОсталосьГоблинов = 0;
                        for (it = объекты.begin(); it != объекты.end(); it++)
                        {
                                объект* Объект = *it;
                                //1. враги
                                if (Объект->пок_имя() == "goblin" || Объект->пок_имя() == "hl")
                                {
                                        if (Объект->пок_Жизнь() <= 0) continue;

                                        ОсталосьГоблинов++;

                                        //приследование
                                        //if (игрок.Жизнь > 0)
                                        Объект->изм_приследование(0);
                                        for (int j = (Объект->пок_x() / Клетка) - 5; j < (Объект->пок_x() /
Клетка) + 4; j++)
                                                if (j == (int)(игрок.пок_x() / Клетка)
                                                        && ((int)((игрок.пок_y() + игрок.пок_h()) /
Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка)
                                                        || (int)((игрок.пок_y() + игрок.пок_h()) /
Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка + 1)))
                                                {
                                                        if (Объект->пок_x() > игрок.пок_x()) Объект-
>изм_зеркало(1); else Объект->изм_зеркало(0);
                                                        Объект->изм_приследование(1);
                                                }

                                        //урон игроку
                                        if (игрок.размер().intersects(Объект->размер()))
                                        {
                                                if (Объект->пок_имя() == "goblin")
                                                if (Объект->пок_t()) {
                                                        Объект->изм_t(0); Объект-
>пок_анимация().АнимЛист[Объект->пок_анимация().ан()].ТекКадр = 0;
                                                }
```

```
                                        Объект->сдел_выстрел();
                                        if (Объект->пок_уд() && Объект-
>пок_анимация().кад() == Объект->пок_кадуд()) {
                                                Объект->изм_уд();
                                                игрок.изм_Жизнь(игрок.пок_Жизнь() -
Объект->пок_урон());
                                                ОзвУронаИгрока.play();
                                        }
                                }
                                //урон врагу
                                for (std::list<объект*>::iterator it2 = объекты.begin(); it2 !=
объекты.end(); it2++)
                                {
                                        объект* стрела = *it2;
                                        if (стрела->пок_имя() == "Bullet")
                                                if (стрела->пок_Жизнь() > 0)
                                                        if (стрела->размер().intersects(Объект-
>размер()))
                                                        {
                                                                стрела->изм_жив(0); Объект-
>изм_Жизнь(Объект->пок_Жизнь() - стрела->пок_урон()); ОзвУрона.play();
                                                                if (Объект->пок_х() > стрела-
>пок_х()) Объект->изм_зеркало(1); else Объект->изм_зеркало(0);
                                                        }
                                }

                                //выподение бонусов
                                if (Объект->пок_Жизнь() <= 0) {

                                        int СлучайноеЧисло = distribution(generator); //
Случайное число.
                                        if (СлучайноеЧисло == 1)
                                                объекты.push_back(new Бонус("c", анимация4,
Объект->пок_х(), Объект->пок_у()));
                                        else if (СлучайноеЧисло == 0)
                                                объекты.push_back(new Бонус("hp",
анимация5, Объект->пок_х(), Объект->пок_у()));
                                }

                        }
                        if (Объект->пок_имя() == "npc1")
                        {
                                Объект->изм_приследование(0);
                                for (std::list<объект*>::iterator it2 = объекты.begin(); it2 !=
объекты.end(); it2++)
                                {
                                        if ((*it2)->пок_имя() == "goblin" || (*it2)->пок_имя() ==
"hl")
                                        {
```

```
                                                for (int j = (Объект->пок_x() / Клетка) - 10; j <
(Объект->пок_x() / Клетка) + 10; j++)
                                                        if (j == (int)((*it2)->пок_x() / Клетка)
                                                                &&((int)(((*it2)->пок_y() + (*it2)-
>пок_h()) / Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка)
                                                                || (int)(((*it2)->пок_y() + (*it2)-
>пок_h()) / Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка - 1)
                                                                || (int)(((*it2)->пок_y() + (*it2)-
>пок_h()) / Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка - 2)
                                                                || (int)(((*it2)->пок_y() + (*it2)-
>пок_h()) / Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка - 3)
                                                                || (int)(((*it2)->пок_y() + (*it2)-
>пок_h()) / Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка - 4)
                                                                || (int)(((*it2)->пок_y() + (*it2)-
>пок_h()) / Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка - 5)
                                                                || (int)(((*it2)->пок_y() + (*it2)-
>пок_h()) / Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка - 6)
                                                                || (int)(((*it2)->пок_y() + (*it2)-
>пок_h()) / Клетка) == (int)((Объект->пок_y() + Объект->пок_h()) / Клетка - 7)))
                                                        {
                                                                if (Объект->пок_x() > (*it2)-
>пок_x()) Объект->изм_зеркало(1); else Объект->изм_зеркало(0);
                                                                Объект-
>изм_приследование(1);
                                                        }

                                                if ((*it2)->размер().intersects(Объект-
>размер()))
                                                {
                                                        Объект->сдел_выстрел();
                                                        (*it2)->сдел_выстрел();
                                                }
                                        }
                                        if ((*it2)->пок_имя() == "Bullet")
                                                if ((*it2)->пок_Жизнь() > 0)
                                                        if ((*it2)->размер().intersects(Объект-
>размер()))
                                                        {
                                                                (*it2)->изм_жив(0); Объект-
>изм_Жизнь(0);
                                                        }
                                }

                        }

                        //сбор стрел
                        if (Объект->пок_имя() == "c") {
                                if (игрок.размер().intersects(Объект->размер()))
                                {
                                        игрок.доб_ЗапасСтрел(10); Объект->изм_жив(0);
```

```
                                        }
                                }
                                //сбор здоровья
                                if (Объект->пок_имя() == "hp") {
                                        if (игрок.размер().intersects(Объект->размер()))
                                        {
                                                игрок.изм_Жизнь(игрок.пок_Жизнь() + 0.2); Объект->изм_жив(0);
                                        }
                                }

                        }

                        //взаимоотношения с босом
                        if (игрок.пок_x() > 45000)
                        {
                                //приследование
                                бос.изм_приследование(0);
                                for (int j = (бос.пок_x() / Клетка) - 7; j < (бос.пок_x() / Клетка) + 8; j++)
                                {

                                        if (j == (int)(игрок.пок_x() / Клетка) && ((int)((игрок.пок_y() + игрок.пок_h()) / Клетка) == (int)((бос.пок_y() + бос.пок_h()) / Клетка)
                                                || (int)((игрок.пок_y() + игрок.пок_h()) / Клетка) == (int)((бос.пок_y() + бос.пок_h()) / Клетка - 1)
                                                || (int)((игрок.пок_y() + игрок.пок_h()) / Клетка) == (int)((бос.пок_y() + бос.пок_h()) / Клетка - 2)
                                                || (int)((игрок.пок_y() + игрок.пок_h()) / Клетка) == (int)((бос.пок_y() + бос.пок_h()) / Клетка - 3)
                                                || (int)((игрок.пок_y() + игрок.пок_h()) / Клетка) == (int)((бос.пок_y() + бос.пок_h()) / Клетка + 1)))
                                        {
                                                if (бос.пок_x() > игрок.пок_x()) бос.изм_зеркало(1); else бос.изм_зеркало(0);
                                                if (бос.пок_прыжокустены() != бос.пок_x())
                                                бос.изм_приследование(1);
                                                if (!игрок.размер().intersects(бос.размер()))
                                                        бос.изм_выстрел2(1);
                                        }
                                }

                                //урон игроку
                                if (игрок.размер().intersects(бос.размер()))
                                {
                                        if (бос.пок_t()) {
                                                бос.изм_t(0);
бос.пок_анимация().АнимЛист[бос.пок_анимация().ан()].ТекКадр = 0;
                                        }
                                        бос.сдел_выстрел();
```

```
                                        if (бос.пок_уд() && бос.пок_анимация().кад() ==
бос.пок_кадуд()) {
                                                бос.изм_уд();
                                                игрок.изм_Жизнь(игрок.пок_Жизнь() -
бос.пок_урон());
                                                ОзвУронаИгрока.play();
                                        }
                                }
                                //урон игроку от камня
                                for (std::list<объект*>::iterator it2 = объекты.begin(); it2 !=
объекты.end(); it2++)
                                {
                                        объект* камень = *it2;
                                        if (камень->пок_имя() == "cam")
                                                if (камень->пок_Жизнь() > 0)
                                                        if (камень-
>размер().intersects(игрок.размер()))
                                                        {
                                                                камень->изм_жив(0);
                                                                игрок.изм_Жизнь(игрок.пок_Жизнь() -
камень->пок_урон());
                                                                ОзвУронаИгрока.play();
                                                        }
                                }
                                //урон врагу
                                for (std::list<объект*>::iterator it2 = объекты.begin(); it2 !=
объекты.end(); it2++)
                                {
                                        объект* стрела = *it2;
                                        if (стрела->пок_имя() == "Bullet")
                                                if (стрела->пок_Жизнь() > 0)
                                                        if (стрела->размер().intersects(бос.размер()))
                                                        {
                                                                стрела->изм_жив(0);
бос.изм_Жизнь(бос.пок_Жизнь() - стрела->пок_урон()); ОзвУрона.play();
                                                        }
                                }
                        }

                        //смерть игрока
                        if (игрок.пок_Жизнь() < 0)
                        {
                                игрок.изм_жив(0);
                                !БойСБосом ? МузыкаВИгре.pause() : МузыкаБос.pause();
МузыкаЗаставка.play();
                                пауза = 0;
                                ОзвСмеха.play();
                                система = 2;
                                window.create(sf::VideoMode(1200, 628), L"Убийца гоблинов",
Style::Close);
```

```
                window.setMouseCursorVisible(0);
        }

        //смена музыки
        if (!БойСБосом && игрок.пок_x() > 45000)
        {БойСБосом = 1;МузыкаВИгре.stop();МузыкаБос.play();}

        //победа
        if (бос.пок_жив() == 0) {
                система = 2;
                пауза = 0;
                window.create(sf::VideoMode(1200, 628), L"Убийца гоблинов",
Style::Close);
                window.setMouseCursorVisible(0);
                МузыкаБос.pause(); победа.play();
        }

                //удодение мёртвых
                for (it = объекты.begin(); it != объекты.end();)
                {
                        объект* п = *it;
                        if (п->пок_жив() == false) {
                                it = объекты.erase(it);
                                delete п;
                        }
                        else it++;
                }

                //обновление персонажей
                игрок.обновить(время);
                бос.обновить(время);
                for (it = объекты.begin(); it != объекты.end(); it++)
                        (*it)->обновить(время);

        clok.restart();//сброс времени

        //отображение на экран
        view.setCenter(игрок.пок_x(), игрок.пок_y());
        window.setView(view);

        text3.setPosition(view.getCenter().x + 300, view.getCenter().y - 550);
        text2.setPosition(view.getCenter().x + 660, view.getCenter().y - 550);
        text.setPosition(view.getCenter().x - 960, view.getCenter().y - 550);
        Босжизнь.setPosition(view.getCenter().x + 660, view.getCenter().y - 550 + 60);
        жизнь.setPosition(view.getCenter().x - 960 + 75, view.getCenter().y - 550 + 60);

        window.clear(Color(85, 170, 255));
        lvl.Draw(window);
        for (it = объекты.begin(); it != объекты.end(); it++)
        {
```

```
                (*it)->показать(window);
                if ((*it)->пок_имя() == "npc1" && !(*it)->пок_приследование())
                {
                        text4.setPosition((*it)->пок_x(), (*it)->пок_y() - 100);
                        text4.setString((*it)->пок_текст());
                        window.draw(text4);
                }
        }
        игрок.показать(window);
        бос.показать(window);

        if (игрок.пок_x() > 45000)window.draw(text2); window.draw(text);
        if (cox) {
                text2.setString("Сохранение");
                text2.setFillColor(Color::Green);
                text2.setPosition(view.getCenter().x + 660, view.getCenter().y - 450);
                window.draw(text2);
                if (игрок.пок_x() > x + 300 || игрок.пок_x() < x - 300)
                cox = 0;
        }
        if (игрок.пок_x() < 2400 && игрок.пок_y() < 1900)window.draw(text3);
        if (игрок.пок_x() > 45000)window.draw(Босжизнь);
        if (игрок.пок_Жизнь() > 0) window.draw(жизнь);
        window.display();
}
if (система == 2)
{
        View view(FloatRect(0, 0, 1200, 628));
        window.setView(view);

        //отслеживание мыши
        if (event.type == sf::Event::MouseMoved)
                мыш.setPosition(event.mouseMove.x, event.mouseMove.y);

        //фон
        d.loadFromFile("Текстуры/og_og_1540932814235486221.jpg");
        фон.setTexture(d);//заливаем текстуру спрайтом
        фон.setTextureRect(IntRect(0, 0, 1200, 628));

        //текст
        text.setFillColor(Color::Black);
        text2.setFillColor(Color::Black);
        text3.setFillColor(Color::Black);
        text5.setFillColor(Color::Black);
        text.setStyle(sf::Text::Bold);
        text2.setStyle(sf::Text::Bold);
        text3.setStyle(sf::Text::Bold);
        text.setString("НАЧАТЬ С НАЧАЛА");
        text2.setString("ПРОДОЛЖИТЬ ИГРУ");
        text3.setString("ПРОДОЛЖИТЬ С ПОСЛЕДНЕЙ\nСОХРАНЁННОЙ ТОЧКИ");
```

```
                        text.setPosition(10, 400);
                        text2.setPosition(10, 550);
                        text3.setPosition(10, 450);
                        text5.setPosition(850, 550);
                        std::ostringstream go;
                        go << ОсталосьГоблинов;

                        //закрытие окна
                        while (window.pollEvent(event))
                                if (event.type == sf::Event::Closed)
                                        window.close();

                        //НАЧАТЬ С НАЧАЛА
                        if (Mouse::getPosition(window).x > 10 && Mouse::getPosition(window).x < 10 +
475 &&
                                Mouse::getPosition(window).y < 400 + 50 &&
Mouse::getPosition(window).y > 400)
                        {

                                text.setFillColor(Color::Red);//красный текст
                                if (Mouse::isButtonPressed(Mouse::Left))
                                {

                                        for (it = объекты.begin(); it != объекты.end();)
                                        {
                                                объект* п = *it;
                                                it = объекты.erase(it);
                                                delete п;
                                        }

                                        for (int i = 0; i < g.size(); i++)
                                                объекты.push_back(new Гоблин(анимация3, lvl,
g[i].rect.left, g[i].rect.top, 0.3,1));

                                        for (int i = 0; i < g1.size(); i++)
                                                объекты.push_back(new Гоблин(анимация3, lvl,
g1[i].rect.left, g1[i].rect.top, 0.4,1));


                                        for (int i = 0; i < g2.size(); i++)
                                                объекты.push_back(new Гоблин(анимация3, lvl,
g2[i].rect.left, g2[i].rect.top, 0.5,1));

                                        for (int i = 0; i < c.size(); i++)
                                                объекты.push_back(new Бонус("c", анимация4,
c[i].rect.left, c[i].rect.top));

                                        for (int i = 0; i < hp.size(); i++)
                                                объекты.push_back(new Бонус("hp", анимация5,
hp[i].rect.left, hp[i].rect.top));
```

```
                                        for (int i = 0; i < hg.size(); i++)
                                                объекты.push_back(new Гоблин(анимация6, lvl,
hg[i].rect.left, hg[i].rect.top, 0.3, 2));

                                        for (int i = 0; i < hl.size(); i++)
                                                объекты.push_back(new Халк(анимация7, lvl,
hl[i].rect.left, hl[i].rect.top));

                                        for (int i = 0; i < npc1.size(); i++)
                                                объекты.push_back(new Нпс1(анимация10, lvl,
npc1[i].rect.left, npc1[i].rect.top));

                                        игрок.изм_x(pl.rect.left);
                                        игрок.изм_y(pl.rect.top);
                                        игрок.изм_Жизнь(1);
                                        игрок.изм_жив(1);

                                        бос.изм_x(bos.rect.left);
                                        бос.изм_y(bos.rect.top);
                                        бос.изм_Жизнь(10);
                                        бос.изм_жив(1);

                                        игрок.изм_ЗапасСтрел(10);

                                        //сброс параметров
                                        МузыкаЗаставка.stop(); победа.stop(); МузыкаБос.stop();
МузыкаВИгре.stop(); МузыкаВИгре.play();
                                        БойСБосом = 0;
                                        система = 1;

                                        cox = 0; cox1 = 0; cox2 = 0; cox3 = 0; cox4 = 0;
ОсталосьГоблинов = 57;

                                        //настройка окна
                                        window.create(VideoMode(), L"Убийца гоблинов",
Style::Fullscreen);
                                        window.setMouseCursorVisible(0);
                                }
                        }

                        //СОХРАНЕНИЕ
                        if (Mouse::getPosition(window).x > 10 && Mouse::getPosition(window).x < 10 +
725 &&
                                Mouse::getPosition(window).y < 450 + 100 &&
Mouse::getPosition(window).y > 450
                                && бос.пок_жив() == 1)
                        {
                                text3.setFillColor(Color::Red);//выдилить текст
                                if (Mouse::isButtonPressed(Mouse::Left))
```

```
                                {
                                        for (it = объекты.begin(); it != объекты.end();)
                                        {
                                                объект* п = *it;
                                                it = объекты.erase(it);
                                                delete п;
                                        }

                                        for (it = объекты2.begin(); it != объекты2.end(); it++)
                                        {
                                                объект* п = *it; объект* r = new Бонус("hp",
анимация5, 1, 1);
                                                if (п->пок_имя() == "c" || п->пок_имя() == "hp")
                                                        r = new Бонус("hp", анимация5, 1, 1);
                                                if (п->пок_имя() == "goblin" || п->пок_имя() ==
"goblin1" || п->пок_имя() == "goblin2")
                                                        r = new Гоблин(анимация3, lvl, 1, 1, 0.3,1);
                                                if (п->пок_имя() == "hl")
                                                        r = new Халк(анимация3, lvl, 1, 1);
                                                if (п->пок_имя() == "npc1")
                                                        r = new Нпс1(анимация10, lvl, 1, 1);
                                                /*if (п->имя == "c" || п->имя == "hp")
                                                        r = new Бос(анимация3, lvl, 1, 1);*/
                                                *r = *п;
                                                объекты.push_back(r);
                                        }

                                        игрок.изм_ЗапасСтрел(запасстрел);
                                        игрок.изм_x(x);
                                        игрок.изм_y(y);
                                        игрок.изм_жив(1);
                                        игрок.изм_Жизнь(Жизнь);
                                        бос.изм_Жизнь(10);

                                        МузыкаЗаставка.pause(); !БойСБосом ? МузыкаВИгре.play() :
МузыкаБос.play();
                                        window.create(VideoMode(), L"Убийца гоблинов",
Style::Fullscreen);
                                        window.setMouseCursorVisible(0);

                                        система = 1;
                                }
                        }

                        //ПРОДОЛЖИТЬ ИГРУ
                        if (пауза)
                        if (Mouse::getPosition(window).x > 10 && Mouse::getPosition(window).x < 10 +
500 &&
                                Mouse::getPosition(window).y < 550 + 50 &&
Mouse::getPosition(window).y > 550
```

```
                && бос.пок_жив() == 1)
            {
                text2.setFillColor(Color::Red);//выдилить текст
                if (Mouse::isButtonPressed(Mouse::Left))
                {
                    //возвращение в игру
                    МузыкаЗаставка.pause(); !БойСБосом ? МузыкаВИгре.play() :
МузыкаБос.play();
                    window.create(VideoMode(), L"Убийца гоблинов",
Style::Fullscreen);
                    window.setMouseCursorVisible(0);
                    if (!пауза && игрок.пок_ЗапасСтрел() < 5) {
игрок.изм_ЗапасСтрел(5); }
                    if (!игрок.пок_жив())
                    {
                        игрок.изм_жив(1); игрок.изм_Жизнь(1);
                    }
                    система = 1;
                }
            }

            //подвили курсор к настройкам
            if (Mouse::getPosition(window).x > 850 && Mouse::getPosition(window).x < 850
+ 300 && \
                Mouse::getPosition(window).y < 550 + 50 &&
Mouse::getPosition(window).y > 550)
            {
                text5.setFillColor(Color::Red);//нажали на кнопку
                if (Mouse::isButtonPressed(Mouse::Left))
                {
                    система = 3; вссистема = 2;
                }
            }

            //отображение обьектов
            window.draw(фон);
            window.draw(text);
            if (!пауза) {
                text.setString("ВЫ ВЫЙГРАЛИ!"); text.setPosition(800, 0);
text.setFillColor(Color::Red);
                if (бос.пок_Жизнь() > 0)
                text.setString("ВЫ ПРОИГРАЛИ"); text.setPosition(800, 0);
text.setFillColor(Color::Red);
                window.draw(text);
                text.setFillColor(Color::Green);
                if (ОсталосьГоблинов > 0) {
                    text.setPosition(700, 100);
                    text.setString("Осталось гоблинов: " + go.str() + "\nскоро они
вновь\nнападут на деревню");
                }
```

```
                window.draw(text);
        }



        window.draw(text2);
        window.draw(text3);
        window.draw(text5);
        window.draw(мыш);
        window.display();
        clok.restart();
}
if (система == 3){
        bool k;
        Texture т, п;
        Sprite точка, ползунок, точка1, ползунок1;
        d.loadFromFile("Текстуры/dark-4487690_1280.jpg");
        т.loadFromFile("Текстуры/12.png");
        п.loadFromFile("Текстуры/1.png");
        фон.setTexture(d);
        точка.setTexture(т); точка1.setTexture(т);
        ползунок.setTexture(п); ползунок1.setTexture(п);
        фон.setTextureRect(IntRect(0, 0, 1280, 720));
        text.setString("НАЗАД");
        text.setPosition(560, 550);

        text4.setString("Громкость музыки\n\n\n\nГромкость звуков");
        text4.setPosition(100, 195);
        //точка.setTextureRect(IntRect(0, 0, 30, 30));
        //ползунок.setTextureRect(IntRect(0, 0, 258, 10));
        точка.setPosition(п1, 290); точка1.setPosition(п2, 200);
        ползунок.setPosition(350, 300); ползунок1.setPosition(350, 210);

        while (window.pollEvent(event))//обработка событий
        {
                соб = 1;

                //отслеживание мыши
                if (event.type == sf::Event::MouseMoved)
                        мыш.setPosition(event.mouseMove.x, event.mouseMove.y);

                //настройки текста
                text.setFillColor(Color::Black);
                text.setStyle(sf::Text::Bold);

                //закрываем окно
                if (event.type == sf::Event::Closed)
                        window.close();

                //подвили курсор к громкости звуков
```

```
                                if (Mouse::getPosition(window).x > 350 &&
Mouse::getPosition(window).x < 578 + 30 &&
                                        Mouse::getPosition(window).y < 290 + 30 &&
Mouse::getPosition(window).y > 290)
                                        if (Mouse::isButtonPressed(Mouse::Left))
                                        {k = 1; п1 = Mouse::getPosition(window).x - 15; } else {if (k) {
ОзвВыстрела.play(); k = 0; } }
                                if (п1 < 350) п1 = 350; if (п1 > 578) п1 = 578;

                                //подвили курсор к громкости музыки
                                if (Mouse::getPosition(window).x > 350 &&
Mouse::getPosition(window).x < 578 + 30 &&
                                        Mouse::getPosition(window).y < 200 + 30 &&
Mouse::getPosition(window).y > 200)
                                        if (Mouse::isButtonPressed(Mouse::Left))
                                                п2 = Mouse::getPosition(window).x - 15;
                                if (п2 < 350) п2 = 350; if (п2 > 578) п2 = 578;

                                точка.setPosition(п1, 290); точка1.setPosition(п2, 200);

                                //подвили курсор к кнопке
                                if (Mouse::getPosition(window).x > 560 &&
Mouse::getPosition(window).x < 560 + 180 &&
                                        Mouse::getPosition(window).y < 550 + 50 &&
Mouse::getPosition(window).y > 550)
                                {
                                        text.setFillColor(Color::Red);//нажали на кнопку
                                        if (Mouse::isButtonPressed(Mouse::Left))
                                        {
                                                система = вссистема;
                                        }
                                }
                        }
                        //громкость
                        ГромкостьМузыки = ((double)(п2 - 350) / 228) * 100;
                        МузыкаВИгре.setVolume(ГромкостьМузыки);
                        МузыкаБос.setVolume(ГромкостьМузыки);
                        МузыкаЗаставка.setVolume(ГромкостьМузыки);
                        победа.setVolume(ГромкостьМузыки);

                        ГромкостьЗвуков = ((double)(п1 - 350) / 228) * 100;
                        ОзвВыстрела.setVolume(ГромкостьЗвуков);
                        ОзвСмеха.setVolume(ГромкостьЗвуков);
                        ОзвУрона.setVolume(ГромкостьЗвуков);
                        ОзвУронаИгрока.setVolume(ГромкостьЗвуков);

                        //отображаем обьекты
                        window.draw(фон);
                        window.draw(text);
                        if (соб) {
```

```
                window.draw(text4);
                window.draw(ползунок);
                window.draw(точка);
                window.draw(ползунок1);
                window.draw(точка1);
                window.draw(мыш);
            }
            window.display();
            clok.restart();
        }
    }
    return 0;
}
```

Вывод

Ожидаемый результат совпал с полученным, ошибок не обнаружено.