Министерство науки и высшего образования

ПЕТРОЗАВОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Физико-технический институт

Индивидуальный отчет по курсовому проекту
по дисциплине «Технология программирования»

Выполнили:

студент гр. 21312

Зудов А.С.

Преподаватель:

канд. физ.-мат. наук

Бульба А.В.

Петрозаводск 2020

Цель работы: разработать игру на языке C++ с использованием библиотеки SFML.

Программная реализация:

Среда разработки: Visual Studio Express 2013;

Язык: C++;

Заголовочные файлы:

Entity.h – Содержит класс сущность. Данный класс является предком класса Игрок и Враг.

Объявляет координаты и размеры сущности, а также логические переменные для управления стартом игры

Player.h - Содержит класс Игрока (PacMan). Данный класс содержит поле очков, и выбранное направление. А также объявляет методы управления игроком, проверки на препятствия и обновления состояния.

Enemy.h - Содержит класс Врага (Призрака). Объявляет буферные поля для памяти предыдущих координат, поле направления, методы проверки на препятствие и обновления состояния.

Interface.h - Содержит класс самого игрового поля. Объявляет поля для текстур, спрайтов, шрифтов, таймеров, карты и список врагов, а также метод взаимодействия с игрой.

Map.h - Содержит класс карты. Объявляет поле карты.

Файлы реализации (.cpp):

Entity.cpp – Содержит конструктор, инициализирующий поля данного класса.

Player.cpp – Содержит реализацию методов получения полей, управления игроком, проверкой на препятствия и обновлением, так же содержит конструктор и деструктор

Enemy.cpp - Содержит реализацию конструктора, методов проверки на препятствия и обновления.

Interface.cpp – Содержит конструктор и деструктор, а так же реализацию метода взаимодействия с полем

Map.cpp – содержит в себе реализацию карты.

Процесс разработки:

Краткое словесное описание сюжета

К нам обратился владелец клуба со старыми игровыми аппаратами. Заказчик просит создать игру, похожую на классическую PacMan. В игре должно быть реализовано:
игровое поле в виде лабиринта, в которых стенки являются препятствиями;
точки, которые поедает главный персонаж и тем самым зарабатывает очки;
3 противника в виде приведений, при прикосновении с которыми заканчивается игра.
Цель игры собрать все точки и не попасться призракам. Игра должна отображать текущий счет очков и возможность на подготовку к запуску игры и выход из нее по завершению»

Список классов

- Экран – Данный класс будет содержать реализацию самой игры. Он будет отвечать за содержание игрового поля.
- Игрок – Данный класс будет содержать в методы и атрибуты игрока. Он будет отвечать за поведение игрока и всё, что с ним происходит во время игры.

Код заголовочных файлов:

Enemy.h:

```
#ifndef __ENEMY_H__
#define __ENEMY_H__
#include "stdafx.h"
#include "Entity.h"
class Enemy :public Entity{  //дочерний класс врага, призрака от класса сущности
private:
	float yy, xx;
	int direction;//Направление врага
public:
	Enemy(Image &image, float X, float Y, int W, int H);
	void checkCollisionWithMap(float Dx, float Dy); //метод проверки на столкновение
	void update(float time);//метод обновления поведения врага
};

#endif;
```

Map.h

```
#ifndef __MAP_H__
#define __MAP_H__
#pragma once
#include "stdafx.h"
//Класс карты
class Map{
public:
	static std::string TileMap[];
};
```

#endif

Код исходных файлов:

Enemy.cpp:

```
#pragma once
#include "stdafx.h"
#include "Enemy.h"
using namespace sf;

Enemy::Enemy(Image &image, float X, float Y, int W, int H) :Entity(image, X, Y, W, H)
{
	sprite.setTextureRect(IntRect(0, 0, w, h));
	direction = rand() % (3);   //случайное направление
	speed = 0.1; //скорость
	dx = speed;
}

void Enemy::checkCollisionWithMap(float Dx, float Dy)//проверка на препятствие
{
	for (int i = y / 32; i < (y + h) / 32; i++)//цикл проверки следующих элементов
	for (int j = x / 32; j<(x + w) / 32; j++)
	{
		int k = y;
		int l = x;
		if ((k % 32 == 0) && (l % 32 == 0) && ((float)((int)x) == x) && ((float)((int)y) == y)){
			if ((mp.TileMap[i + 1][j] == ' ') || (mp.TileMap[i + 1][j] == 's')) direction = rand()
% (4);
			if ((mp.TileMap[i - 1][j] == ' ') || (mp.TileMap[i - 1][j] == 's')) direction = rand()
% (4);
		}
		if (mp.TileMap[i][j] == '0')//если впереди препятствие
		{//в соответствие с выбранным направлением перемещает на предыдущий элемент
			if (Dy > 0) {
				y = i * 32 - h;  dy = -0.1;
				direction = rand() % (3); //случайное направление
			};//по Y
			if (Dy < 0) {
				y = i * 32 + 32; dy = 0.1;
				direction = rand() % (3);
			}
			if (Dx > 0) {
				x = j * 32 - w; dx = -0.1;
				direction = rand() % (3);
			}
			if (Dx < 0) {
				x = j * 32 + 32; dx = 0.1;
				direction = rand() % (3);
			}

		}
	}
}
```

```
void Enemy::update(float time)
{
	if (Keyboard::isKeyPressed(Keyboard::Enter)) life = true;

		switch (direction) //выбор направления
		{
		case 0:
		{//вправо
				dx = speed; //скорость по x
				dy = 0; //скорость по y
				CurrentFrame += 0.005*time; //просчтет кадров
				if (CurrentFrame > 3) CurrentFrame -= 3;//если достигнет 3, то
рисовать спрайт заново
				sprite.setTextureRect(IntRect(32 * int(CurrentFrame), 0, 32, 32));
				break;
		}
		case 1:
		{//влево
				dx = -speed;
				dy = 0;
				CurrentFrame += 0.005*time;
				if (CurrentFrame > 3) CurrentFrame -= 3;
				sprite.setTextureRect(IntRect(32 * int(CurrentFrame), 0, 32, 32));
				break;
		}
		case 2:
		{//вверх
				dy = -speed;
				dx = 0;
				CurrentFrame += 0.005*time;
				if (CurrentFrame > 3) CurrentFrame -= 3;
				sprite.setTextureRect(IntRect(32 * int(CurrentFrame), 0, 32, 32));
				break;
		}
		case 3:
		{//вниз
				dy = speed;
				dx = 0;
				CurrentFrame += 0.005*time;
				if (CurrentFrame > 3) CurrentFrame -= 3;
				sprite.setTextureRect(IntRect(32 * int(CurrentFrame), 0, 32, 32));
				break;
		}
		}

		if (dx > 0){
			xx = x;
			x += dx*time; //движение по “X”
			if (xx < floor(x)) x = floor(x);
		}

		if (dx < 0){
			xx = x;
			x += dx*time; //движение по “X”
```

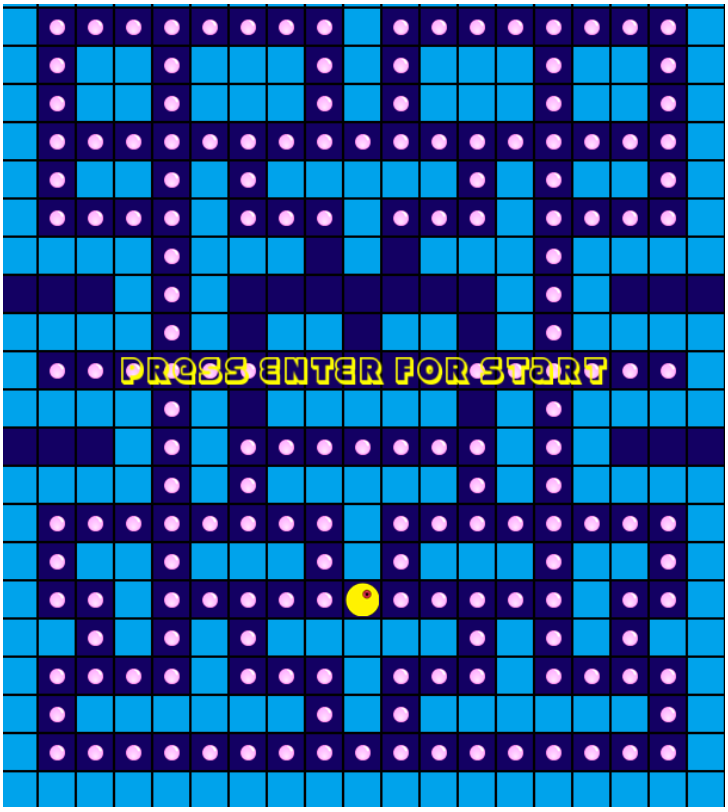
```
            if (x<floor(xx - 0.01)) x = round(x);
        }

        if (dy < 0){
            yy = y;
            y += dy*time; //движение по “Y”
            if (yy < floor(y)) y = floor(y);
        }

        if (dy > 0){
            yy = y;
            y += dy*time; //движение по “Y”
            if (y<floor(yy - 0.01)) y = round(y);
        }

        checkCollisionWithMap(dx, 0);//обрабатываем столкновение по X
        checkCollisionWithMap(0, dy);//обрабатываем столкновение по Y
        sprite.setPosition(x, y);

}
```

Map.cpp:

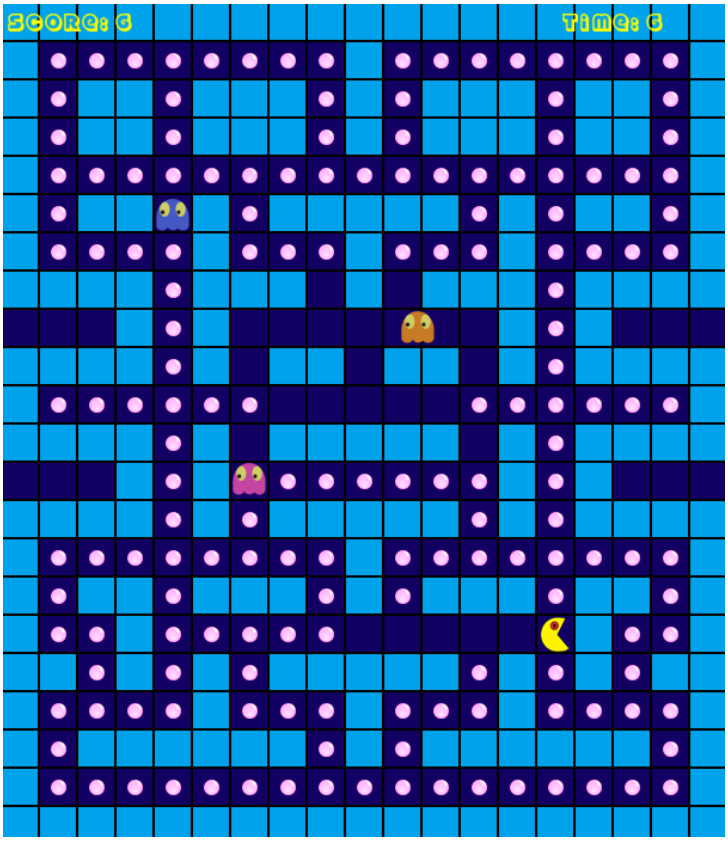
```
#include "map.h"

std::string Map::TileMap[] = {  //массив строк карты
        "0000000000000000000",
        "0ssssssss0ssssssss0",
        "0s00s000s0s000s00s0",
        "0s00s000s0s000s00s0",
        "0sssssssssssssssss0",
        "0s00s0s00000s0s00s0",
        "0ssss0sss0sss0ssss0",
        "0000s000 0 000s0000",
        "   0s0       0s0   ",
        "0000s0 00 00 0s0000",
        "0ssssss     ssssss0",
        "0000s0 00000 0s0000",
        "   0s0sssssss0s0   ",
        "0000s0s00000s0s0000",
        "0ssssssss0ssssssss0",
        "0s00s000s0s000s00s0",
        "0ss0sssssssssss0ss0",
        "00s0s0s00000s0s0s00",
        "0ssss0sss0sss0ssss0",
        "0s000000s0s000000s0",
        "0sssssssssssssssss0",
        "0000000000000000000",
};
```

Руководство пользователя

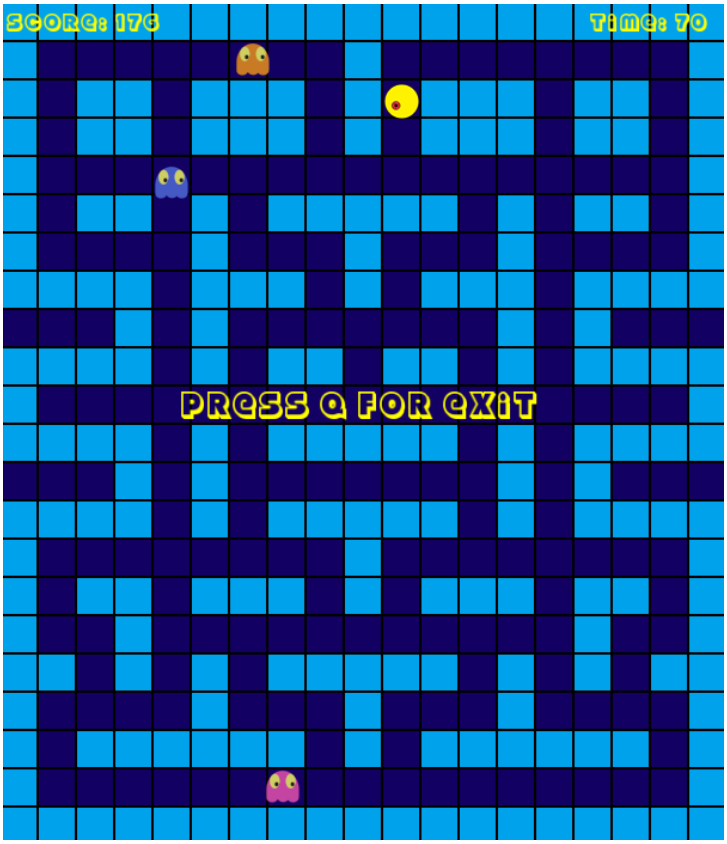
При запуске программы, первое что видит игрок это сообщение об запуске программы с помощью нажатия клавиши Enter:



После нажатия клавиши запускается игра и на экране обновляются действия героя и призраков:



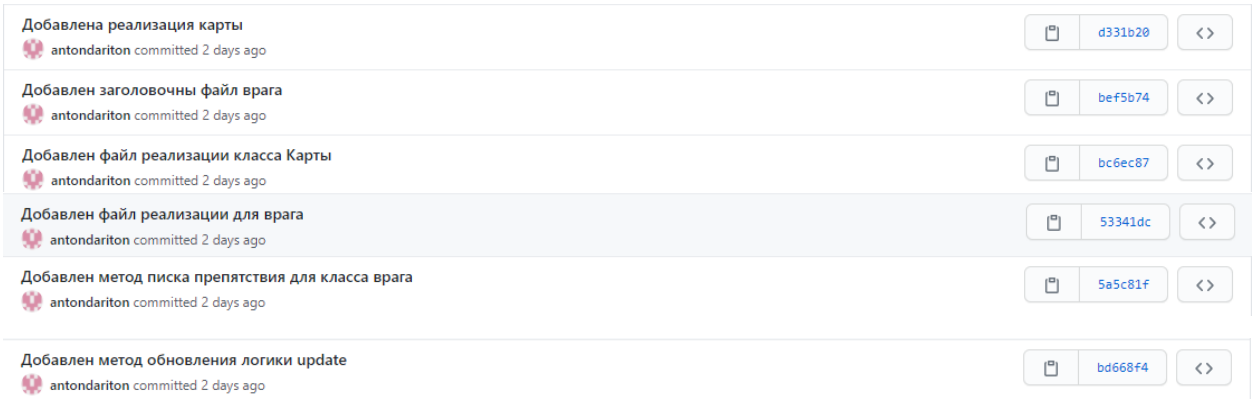
Собрав все точки либо погибнув от призрака, игра выдаст сообщения об выходе программы через кнопку Q:



История проекта на GitHub.

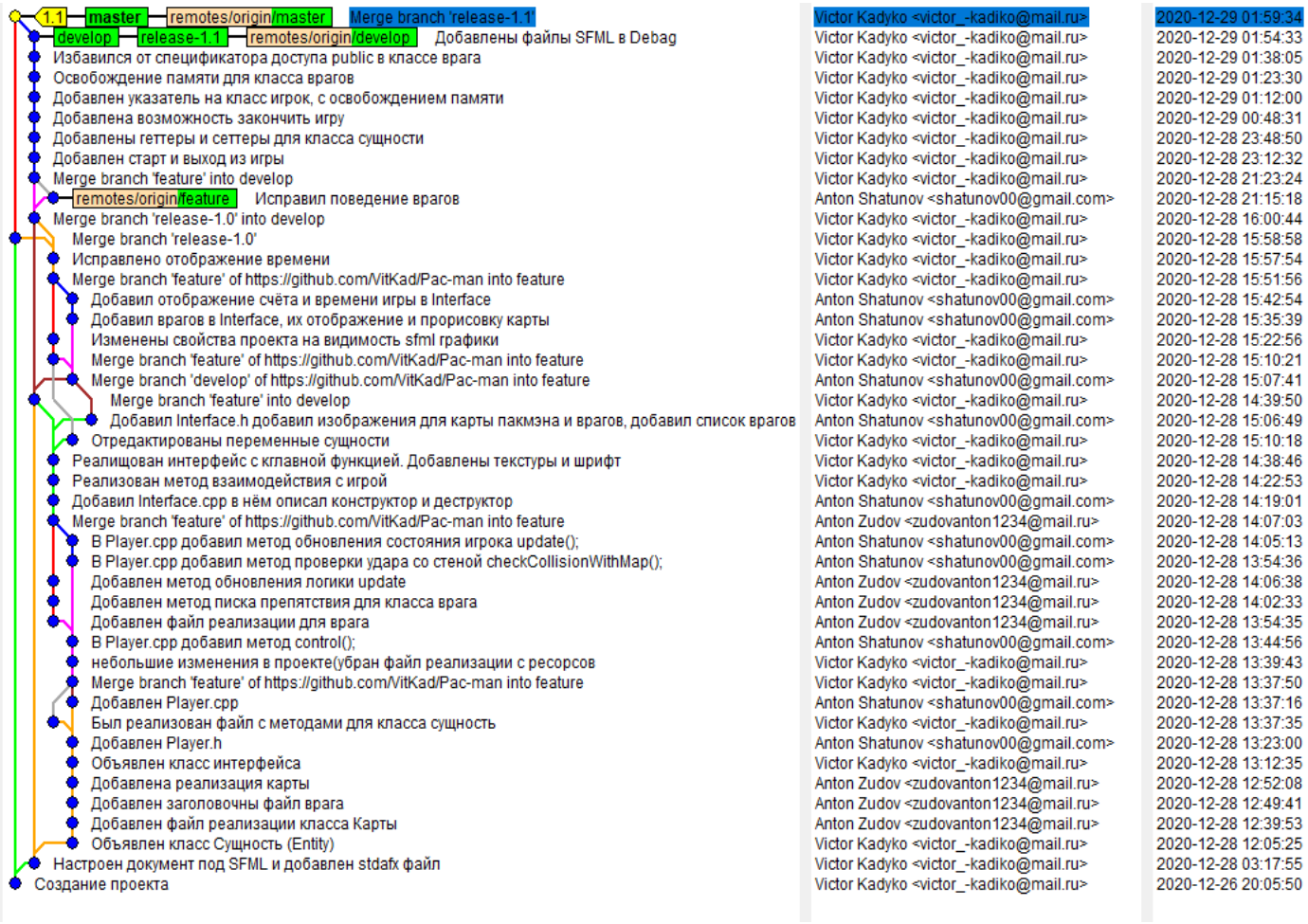
Адрес репозитория: https://github.com/VitKad/Pac-man.git

Зудов Антон– antondariton



В коммите bc6ec87 был добавлен файл реализации класса
В коммите bef5b74 был добавлен заголовочный файл врага
В коммите d331b20 была добавлена реализация карты
В коммите 53341dc был добавлен файл реализации для врага
В коммите 5a5c81f был добавлен метод поиска препятствия для класса врага
В коммите bd668f4 был добавлен метод обновления логики update

Графическое отображение:



1.1 master remotes/origin/master Merge branch 'release-1.1'	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-29 01:59:34
develop release-1.1 remotes/origin/develop Добавлены файлы SFML в Debag	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-29 01:54:33
Избавился от спецификатора доступа public в классе врага	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-29 01:38:05
Освобождение памяти для класса врагов	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-29 01:23:30
Добавлен указатель на класс игрок, с освобождением памяти	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-29 01:12:00
Добавлена возможность закончить игру	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-29 00:48:31
Добавлены геттеры и сеттеры для класса сущности	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 23:48:50
Добавлен старт и выход из игры	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 23:12:32
Merge branch 'feature' into develop	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 21:23:24
remotes/origin/feature Исправил поведение врагов	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 21:15:18
Merge branch 'release-1.0' into develop	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 16:00:44
Merge branch 'release-1.0'	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 15:58:58
Исправлено отображение времени	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 15:57:54
Merge branch 'feature' of https://github.com/VitKad/Pac-man into feature	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 15:51:56
Добавил отображение счёта и времени игры в Interface	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 15:42:54
Добавил врагов в Interface, их отображение и прорисовку карты	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 15:35:39
Изменены свойства проекта на видимость sfml графики	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 15:22:56
Merge branch 'feature' of https://github.com/VitKad/Pac-man into feature	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 15:10:21
Merge branch 'develop' of https://github.com/VitKad/Pac-man into feature	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 15:07:41
Merge branch 'feature' into develop	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 14:39:50
Добавил Interface.h добавил изображения для карты пакмэна и врагов, добавил список врагов	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 15:06:49
Отредактированы переменные сущности	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 15:10:18
Реалищован интерфейс с клавной функцией. Добавлены текстуры и шрифт	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 14:38:46
Реализован метод взаимодействия с игрой	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 14:22:53
Добавил Interface.cpp в нём описал конструктор и деструктор	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 14:19:01
Merge branch 'feature' of https://github.com/VitKad/Pac-man into feature	Anton Zudov <zudovanton1234@mail.ru>	2020-12-28 14:07:03
В Player.cpp добавил метод обновления состояния игрока update();	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 14:05:13
В Player.cpp добавил метод проверки удара со стеной checkCollisionWithMap();	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 13:54:36
Добавлен метод обновления логики update	Anton Zudov <zudovanton1234@mail.ru>	2020-12-28 14:06:38
Добавлен метод писка препятствия для класса врага	Anton Zudov <zudovanton1234@mail.ru>	2020-12-28 14:02:33
Добавлен файл реализации для врага	Anton Zudov <zudovanton1234@mail.ru>	2020-12-28 13:54:35
В Player.cpp добавил метод control();	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 13:44:56
небольшие изменения в проекте(убран файл реализации с ресорсов	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 13:39:43
Merge branch 'feature' of https://github.com/VitKad/Pac-man into feature	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 13:37:50
Добавлен Player.cpp	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 13:37:16
Был реализован файл с методами для класса сущность	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 13:37:35
Добавлен Player.h	Anton Shatunov <shatunov00@gmail.com>	2020-12-28 13:23:00
Объявлен класс интерфейса	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 13:12:35
Добавлена реализация карты	Anton Zudov <zudovanton1234@mail.ru>	2020-12-28 12:52:08
Добавлен заголовочны файл врага	Anton Zudov <zudovanton1234@mail.ru>	2020-12-28 12:49:41
Добавлен файл реализации класса Карты	Anton Zudov <zudovanton1234@mail.ru>	2020-12-28 12:39:53
Объявлен класс Сущность (Entity)	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 12:05:25
Настроен документ под SFML и добавлен stdafx файл	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-28 03:17:55
Создание проекта	Victor Kadyko <victor_-kadiko@mail.ru>	2020-12-26 20:05:50

Заключение.

Нашей командой была разработана игра, которая соответствует требованиям заказчика. Выбор среды разработки и языка программирования остался прежним. Была использована система контроля версия Git. Она применялась для совместной разработки программы. Сбоев и зависаний не наблюдается. Был использован принцип раздельной компиляции. Все классы разделены на отдельные заголовочные файлы, имеющие свою реализацию в соответствующих .cpp файлах. В программе реализована очистка динамической памяти. Неиспользованных переменных и избыточных алгоритмов не наблюдается. В отчете приведены диаграмма вариантов использования и диаграмма классов. Цель, поставленная заказчиком, выполнена.