Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

«Уральский федеральный университет имени первого Президента России Б.Н. Ельцина»

Кафедра интеллектуальных информационных технологий

Оценка работы \_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель от УрФУ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Игра​ «Portal2D»​

Отчет по проекту

Студенты ​*Костюченко А.А., Наговицин М.Д., Меньшиков Ю.А., Жуков В.В.*

Специальность (направление подготовки) ​*Программная инженерия*

Группа ​*ФО-160001*

Екатеринбург 2017

Оглавление

**1 Постановка задачи**

**1.1 Цель**

Создать консольное приложение, ознакомиться с основными алгоритмами и структурами данных.

**1.2 Ограничения**

* по срокам выполнения работы
* по использованию языков программирования
* по использованию объектно-ориентированных средств языка программирования

**1.3 Задачи**

* изучить литературу, касающуюся языка программирования C++
* написать программу, реализующую игру «Portal2D»
* закомментировать код
* написать отчет по написанной программе
* научиться работать в команде

**2 Проект решения**

**2.1 Общее описание решения**

В качестве приложения была выбрана игра «Portal2D», в качестве средства разработки – интегрированная среда разработки Microsoft Visual Studio 2015.

**3 Описание решения**

**3.1 Модуль Map**

Состоит из одной функции

MapShell​\*\* ​createMap​(​char​\* ​levelName​, ​Hero​\* hero​ ​ , Aim​ \* ​ aim​ , ​ RedPortal​ \*​ redPortal​, BluePotal​ ​\* ​bluePortal​);

Функция отвечает за создание карты (считывая уровень из текстового файла, функция заполняет структурный массив необходимой информацией) Также функция заполняет поля в структурных переменных, хранящих координаты объектов на карте.

Также модуль содержит описания структур: MapShell, Hero, Aim, RedPortal, BluePortal.

Каждая из структур содержит поля для хранения различной информации: координат, очков здоровья (в случае Hero) и информации о содержимом ячейки на карте (MapShell)

**3.2 Модуль Instruments**

Состоит из двух функций:

void​ ​drawFrame​(​MapShell​\*\* ​map​);

Данная функция, принимая на вход массив ячеек карты, выводит на экран содержимое каждой ячейки в зависимости от содержимого.

void​ ​clearScreen​();

Данная функция отвечает за очистку экрана, отключает мерцание экрана

**3.3 Модуль Gameplay**

Модуль отвечает за процесс игры, за взаимодействие объектов на карте.

Запуск уровня осуществляется с помощью функции:

double game::startLevel(char\* levelName)

Функция возвращает количество очков, набранное игроком. В качестве аргумента получает имя файла (в котором хранится уровень). Из функции осуществляются вызовы следующих функций: ​

* Создание карты (считывание из файла):

​MapCell\*\* createMap(char\* levelName, GameInfo\* gameInfo)

* Отрисовка кадра:

​void drawFrame(MapCell\*\* map, GameInfo\* gameInfo)

* Запуск главной функции модуля: ​

​void performAnAction(GameInfo\* gameInfo, MapCell\*\* map)

Модуль содержит в себе 14 функций, главной функцией является:

void performAnAction(GameInfo\* gameInfo, MapCell\*\* map)

Функция принимает на вход структуру, в которой содержится информация о динамическом объекте карты (персонаж, портал и т.д.), и массив ячеек карты.

В данной функции запущен цикл, зависящий от параметра gameIsRunning, в котором программа “слушает” нажатия пользователя. В зависимости от нажатой клавиши, происходит перемещение игрока, прицела, а также высчитывается прошедшее время, отрисовываются кадры и контролируется расположение игрока в воздухе (гравитация).

Перемещение осуществляется при помощи пяти функций:

void jump(GameInfo\* gameInfo, game::MapCell\*\* map)

void moveLeft(char type, GameInfo\* gameInfo, MapCell\*\* map)

void moveRight(char type, GameInfo\* gameInfo, MapCell\*\* map)

void moveUp(char type, GameInfo\* gameInfo, MapCell\*\* map)

void moveDown(char type, GameInfo\* gameInfo, MapCell\*\* map)

Функции принимают на вход текстуру, структуру, содержащую в себе информацию о динамическом объекте игры, и массив ячеек карты.

Данные функции переносят модель и координаты​ объекта в соответствующую направлению ячейку карты.

А также есть функция, отвечающая за гравитацию:

void gravity(MapCell\*\* map, GameInfo\* gameInfo)

Если под персонажем​ отсутствует непроходимый блок, то его модель и координаты перемещаются на ячейку карты вниз.

В игре есть возможность устанавливать порталы, осуществляется это

посредством функции:

void setPortal(char type, GameInfo\* gameInfo, game::MapCell\*\* map)

Функция принимает на вход модель одного из порталов, структуру, содержащую в себе информацию о динамическом объекте игры, и массив ячеек карты.

Считываются​ координаты прицела, по которым устанавливается один из порталов.

Для входа в установленный портал используется функция:

void enterThePortal(char type, GameInfo\* gameInfo, MapCell\*\* map)

Функция принимает на вход модель игрока, структуру, содержащую в себе информацию о динамическом объекте игры, и массив ячеек карты.

При входе в портал персонаж перемещается по координатам второго портала.

В игре есть непроходимые стены, которые отключаются посредством кнопок. Активация кнопки осуществлена с помощью функции:

void activateTheButton(GameInfo\* gameInfo, game::MapCell\*\* map)

Функция принимает на вход структуру, содержащую в себе информацию о динамическом объекте игры, и массив ячеек карты.

При нажатии на кнопку непроходимая стена отключается, и ячейка по координатам непроходимой стены становится проходимой.

Уровень завершается, если координаты героя и выхода совпадают, данное условие проверяет функция:

bool checkGameOverConditions(GameInfo\* gameInfo, MapCell\*\* map)

Функция принимает структуру, содержащую информацию о динамических объектах, и массив ячеек карты. Функция меняет значение переменной gameIsRunning типа bool с true на false и цикл прекращается.

**3.4 Модуль Menu**

Модуль отвечает за работу много вкладочного меню.

Все функции находятся в пространстве имен menu.

Прорисовка каждого пункта меню осуществляется функциями

printPointRecord(int key),

printPointRecordSearch(int key),

printMenu(int key),

printPointStart(int key)

Каждая из них принимает номер пункта на котором остановился пользователь и в зависимости от этого выводит нужный вариант.

За воспроизведение вкладок отвечают данные функции:

Функция, вызываемая в функции main является стартовым (главным) меню

menu(queue::Queue<int> \*queue, bool flag)

Остальные функции воспроизводящие пункты меню имеют префикс doPoint, следом за ним написано название вкладки:

doPointStart(queue::Queue<int> \*queue, bool flag),

doPointRecords(),

Если после doPoint написано два названия с большой буквы, следовательно, это пункт предыдущего подпункта меню

doPointRecordSearch()

Перемещение по меню осуществляет:

controlMenu(ParametersForMenu parametersForMenu)

Она принимает структурную переменную *parametersForMenu,* содержащую в себе значения верхней и нижней границы определенной вкладки меню, а также функцию прорисовки этого же пункта меню.

В конце работы возвращает значения пункта в который захотел зайти пользователь, или же при нажатии ESCAPE выходит из текущего пункта (если находится в стартовом меню, то выходит из игры)

**3.5 Records.cpp**

Модуль отвечает за всю работу с рекордами в игре.

Состоит из 7 функций в пространстве имён “records”, которые объявлены в Records.h. Также пространство “records” содержит структурный тип DataAboutTheChampion.

**3.6 Queue.h**

Модуль отвечает за работу с двусвязной очередью.

Реализация функций находится в header файле из-за невозможности объявления в cpp файле, по причине использования template. На данный момент состоит из 3 функций в пространстве имён “queue”, которое также содержит 2 структурных типа Node и Queue.

**3.7 SortingMethods.h**

Модуль отвечает за работу с алгоритмами сортировки.

Реализация функций находится в header файле из-за невозможности объявления в cpp файле, по причине использования template. На данный момент состоит из 4 функций в пространстве имён “sorting”.

**3.8 List.cpp**

Модуль отвечает за работу с односвязным списком.

Состоит из 4 функций в пространстве имён “list”, которые объявлены в Lists.h. Также пространство “list” содержит структурный тип List.