|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт кибернетики | |
| Кафедра программного обеспечения систем радиоэлектронной аппаратуры | |
|  |
|  |
|  |
| **КУРСОВАЯ РАБОТА** | |

по дисциплине \_\_\_\_\_\_«Алгоритмические языки и программирование, Язык C++» \_\_\_\_\_   
 (наименование дисциплины)

**Тема курсовой работы:** Разработка игры «NetWalk»

**Студент группы** \_КМБО-02-17, Дякин Иван Павлович\_\_\_\_\_\_\_\_\_  
 (учебная группа, фамилия, имя, отчество студента) (подпись студента)

**Руководитель курсовой работы**., ,.Милонов Г. А. \_\_\_\_\_\_\_\_\_\_\_\_\_ **(**Должность, звание, ученая степень) (подпись руководителя)

**Консультант**. \_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 (Должность, звание, ученая степень) (подпись руководителя)

Работа представлена к защите «\_\_\_\_» \_\_\_\_\_\_\_\_\_20\_\_г.

Допущен к защите «\_\_\_\_» \_\_\_\_\_\_\_\_\_20\_\_г.

Москва 2018

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт кибернетики |
| Кафедра программного обеспечения систем радиоэлектронной аппаратуры |
| |  |  | | --- | --- | |  | УТВЕРЖДАЮ  Заведующий кафедрой  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Завьялов А.В..  Подпись ФИО  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 г. | |

**ЗАДАНИЕ**

**На выполнение курсового проекта (работы)** по дисциплине

**«Алгоритмические языки и программирование, Язык C++»**

Студент Дякин Иван Павлович Группа КМБО-02-17

**Тема** «Игра NetWalk»

**Перечень вопросов, подлежащих разработке, и обязательного графического материала**

Создать игру с графическим интерфейсом в космическом стиле. Должны быть реализованы взаимодействие игровых объектов, графический и звуковой функционал.

**Срок представления к защите курсового проекта (работы):** до «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2018 г.

**Задание на курсовой проект (работу) выдал** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

*Подпись руководителя ФИО руководителя*

«\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2018 г.

**Задание на курсовой проект (работу) получил** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

*Подпись обучающегося ФИО исполнителя*

«\_\_\_\_» \_\_\_\_\_\_\_\_\_ 2018 г.

Москва 2018

Содержание

[Введение 4](#_Toc533678131)

[1. Техническое задание 5](#_Toc533678132)

[1.1 Введение 5](#_Toc533678133)

[1.1.1 Наименование программы: 5](#_Toc533678134)

[1.1.2 Краткая характеристика области применения 5](#_Toc533678135)

[1.2 Основание для разработки 5](#_Toc533678136)

[1.3 Назначение разработки 5](#_Toc533678137)

[1.4 Требования, предъявляемые к программе 5](#_Toc533678138)

[1.4.1 Требования к функциональным характеристикам программы 5](#_Toc533678139)

[1.4.2 Требования к техническим средствам, используемым при работе программы 6](#_Toc533678140)

[1.4.3 Требования к языкам программы и среде разработки программы 6](#_Toc533678141)

[1.5 Требования к программной документации 6](#_Toc533678142)

[1.6 Этапы разработки 6](#_Toc533678143)

[2. Обзор способов организации данных и обоснование выбора структуры данных для эффективного выполнения операций 8](#_Toc533678144)

[3. Описание программы 8](#_Toc533678145)

[3.1 Общие сведения 8](#_Toc533678146)

[3.1.1 Наименование программы 8](#_Toc533678147)

[3.1.2 Программное обеспечение, необходимое для функционирования программы 8](#_Toc533678148)

[3.1.3 Язык программирования, на котором написана программа 8](#_Toc533678149)

[3.2 Функциональное назначение программы 8](#_Toc533678150)

[3.3 Описание логической структуры программы: 9](#_Toc533678151)

[3.3.1 Алгоритмы, используемые в программе 9](#_Toc533678152)

[3.3.2 Структура программы с описанием функций составных частей и связей между ними 10](#_Toc533678153)

[3.4 Технические средства 11](#_Toc533678154)

[3.5 Вызов программы 11](#_Toc533678155)

[Заключение 15](#_Toc533678156)

[Список использованных источников 16](#_Toc533678157)

[Приложение 17](#_Toc533678158)

[Текст программы 17](#_Toc533678159)

Введение

Задачей курсовой работы является создание проекта «Игра NetWalk». В ходе работы развиваются навыки программирования, анализа сложности алгоритмов, критического мышления и абстрагирования.

Составленное техническое задание по дисциплине “Алгоритмические языки и программирование, Язык C++” является документом к курсовой работе, который отражает все этапы разработки программного продукта, а также процесс проектирования и выявление требований, предъявляемых конечному продукту.

В ходе выполнения курсовой работы должно быть проведено освоение или углубление общекультурных и профессиональных компетенций, предусмотренных федеральным образовательным стандартом, а именно: ОПК-1, ПК-14, ПК-20.

Выполнение курсовой работы должно опираться на положения СМКО МИРЭА 7.5.1/04.И.05-16 «Инструкция по организации и проведению курсового проектирования» от 06.12.2016

1. Техническое задание

1. Введение

Данная курсовая работа предусмотрена учебным планом по специальности и выполняется в третьем семестре. Целью данной работы является закрепление практических навыков по курсу "Алгоритмические языки и программирование, Язык С++", расширение эрудиции в профессиональной области и получении и закрепление навыков самостоятельного решения инженерных задач на всех этапах.

* 1. Наименование программы:

Название данного приложения будет напрямую перекликаться с темой курсовой работы: «NetWalk». Данное название целиком и полностью отражает функционал будущего приложения.

* 1. Краткая характеристика области применения

Программа предназначена для сферы компьютерных развлечений.

1. Основание для разработки

Учебный план по направлению 01.03.02 «Прикладная математика и информатика» (бакалавры). Создание приложения, способного дать пользователю возможность играть в игру.

1. Назначение разработки

Программа предназначена для игры одного пользователя с компьютером в игру «NetWalk».

1. Требования, предъявляемые к программе
2. Требования к функциональным характеристикам программы

Данный проект является компьютерной игрой, вследствие чего предусматривается одна категория пользователей - игроки. В процессе работы приложения пользователь является непосредственным участник игрового процесса и оказывает непосредственное влияние на него

Программы должна обладать следующим функционалом:

а) графический функционал;

б) звуковой функционал;

в) графический интерфейс пользователя;

г) сцены;

1. Требования к техническим средствам, используемым при работе программы

Для нормального функционирования программы необходим IBM-совместимый компьютер под управлением операционной системы семейства Windows (желательно последней версии), Macintosh или Linux, монитор с разрешением экрана не ниже 1024х768 (рекомендуемое 1280х720), около 100 МБ свободного места на жёстком диске для установки и работы программы и хранения пользовательских файлов, манипулятор любого типа и клавиатура.

1. Требования к языкам программы и среде разработки программы

Для разработки используется язык программирования C++, в качестве среды разработки и графического интерфейса пользователя выступает фреймворк SFML.

1. Требования к программной документации
2. Пояснительная записка в соответствии со стандартами РТУ МИРЭА (СМКО МИРЭА 7.5.1/04.И.05-16);
3. Проектная документация, составленная в соответствии с ГОСТ.

В процессе создания приложения вся проделанная работа документируется, должны быть сохранены все детали разработки, а также трудности, с которыми пришлось столкнуться. Всё выше перечисленное должно быть отражено в пояснительной записке, которая прилагается проекту.

1. Этапы разработки

В «жизненный цикл» приложения включены следующие этапы:

1. Исследование концепции; 12.09.2018-14.09.2018
2. Выработка требования; 12.09.2018-14.09.2018
3. Проектирование; 12.09.2018-14.09.2018
4. Реализация компонент: написание логики программы; 14.09.2018-26.09.2018
5. Интеграция компонент: написание графического интерфейса; 14.09.2018-26.09.2018
6. Тестирование и отладка созданного приложения; 20.09.2018-25.12.2018
7. Оформление проектной документации; 14.09.2018-29.12.2018
8. Сдача проекта и защита курсовой работы. 29.12.2018

Дальнейшее сопровождение программы после сдачи и защиты курсового проекта не предусмотрено.

2**. Обзор способов организации данных и обоснование выбора структуры данных для эффективного выполнения операций**

Программа содержит основную функцию «Main», в котором проходят основные стадии программы: «Подготовка» и «Игра». На этапе «Подготовка» производится генерация уровня, загрузка медиафайлов игры. На этапе «Игра» происходит обработка взаимодйствия с пользователем. Такая реализация удобна для дальнейшего улучшения системы.

3. Описание программы

1. Общие сведения
2. Наименование программы

Название разрабатываемой программы – игра «NetWalk».

1. Программное обеспечение, необходимое для функционирования программы

Требования к программному обеспечению:

Операционная система семейства Windows XP и позднее, Macintosh или операционная система семейства Linux;

100 МБ свободного места на жестком диске.

1. Язык программирования, на котором написана программа

Программа написана на языке С++. Данный язык программирования хорошо подходит для работы с классами. С его помощью мы легко можем осуществить взаимодействие всех классов в программе. Так, для реализации курсовой работы был выбран SFML - кроссплатформенный инструментарий разработки ПО на языке программирования C++. Данный инструмент был выбран из-за удобства в использовании.

3.2 **Функциональное назначение программы**

Игра «NetWalk» стилизована под планету Земля и имеет характерный ей стиль оформления, графические объекты и музыкальное сопровождение.

Программа содержит в себе начальную сцену.

Пользователь может начать игру нажав левой кнопкой мыши на любую область окна.

После этого начнется игра. Игрок будет выступать в роли техника. Его задачей является настройка сети компьютеров планеты Земля путём отладки соединений между ними. Пользователь может поворачивать соединения для соединения компьютеров в единую сеть. Игра заканчивается, когда у всех компьютеров будет доступ в Интернет.

3.3 Описание логической структуры программы:

3.3.1 Алгоритмы, используемые в программе

Основная часть алгоритма программы находится в основном файле игры. Пользователь начинает игру нажав левой кнопкой мыши на любую область экрана. После этого непосредственно начнется игра. Процесс игры и перерисовки объектов реализуется с помощью объекта базового класса «RenderWindow». Игра заканчивается, когда пользователь настроет . После окончании игры игрок будет перенаправлен на сцену «Win».

3.3.2 Структура программы с описанием функций составных частей и связей между ними

Программа содержит 2 главных файла - это «Main» и «Pipe».

Функция «Main» представляет собой все сцены программы. Он содержит сцены «Win», «Game», «Menu». Также он содержит основной метод, начинающий игру. Фон реализован в виде картинки. Во время исполнения сцен «Menu», «Game», «Win» играет музыкальный аудиофайл. Реализация данной сцены показана на Рис. 1.

Функция «Main» содержит в себе объекты графических структуры «Pipe», «Sprite». Он инициализирует их в конструкторе. Там же он добавляет фон и аудиофайл. Поворот объектов происходит при помощи «Event» Реализация данного класса показана на Рис. 2.

Генерация уровня происходит в функции «generatePuzzle». Она позволяет каждую игру пользователя сделать интереснее, добавив элемент случайности. Алгоритм генерации был найден в сети Интернет и модернизирован.

3.4 **Технические средства**

Для нормального функционирования программы необходим IBM-совместимый компьютер под управлением операционной системы семейства Windows или Linux, монитор с разрешением экрана не ниже 1024х768, около 100 МБ свободного места на жёстком диске для установки и работы программы и хранения пользовательских файлов.

3.5 **Вызов программы**

Запуск программы осуществляется путём запуска \* файла, содержащего название программы. При вызове программы будет открыта сцена «Menu».

На Рисунке 1 представлена сцена «Menu».

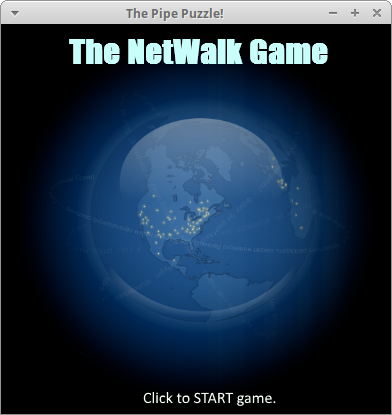


Рисунок 1 – Сцена «Menu».

После нажатия на любую область окна начнется игра. Процесс игры и объекты стукрутры «Pipe» представлены на Рисунке 2.



Рисунок 2 – Объекты структуры «Pipe» и добавленные на него графические объекты игры.

Сцена «Win» появляется, когда все компьютеры подключены в сеть Интернет. Заменяется фон и обработка действий пользователя, сцена представлена на Рисунке 3. Далее для выхода из игры можно использовать ещё одно нажатие левой кнопки мыши по любой области окна или стандартный «крест» в правом верхнем углу окна.



*Рисунок 3 –* Сцена *«Win».*

Заключение

В ходе написания курсовой работы с использованием полученных знаний была создана игра «NetWalk». В ходе проекта были задействованы навыки объектно-ориентированного программирования на языке C++ и использования кроссплатформенного фреймворка SFML, навыки работы с VS Code, отладчиком GDB, файлами сборки Makefile и взаимодействия с файлами. Готовый программный продукт полностью удовлетворяет требованиям технического задания.

В ходе выполнения курсовой работы было проведено освоение и углубление компетенций, предусмотренных федеральным образовательным стандартом высшего образования по направлению подготовки бакалавров 01.03.02 Прикладная математика и информатика.

При выполнении курсовой работы и оформлении отчета по ней были использованы положения СМКО МИРЭА 7.5.1/04.И.05-16 «Инструкция по организации и проведению курсового проектирования» от 06.12.2016.

Список использованных источников

1. C++ reference [Электронный ресурс] URL: <http://en.cppreference.com/w/>
2. SFML Documentation [Электронный ресурс] URL https://www.sfml-dev.org/documentation/
3. Р. Лафоре Объектно-ориентированное программирование в С++. 4-е изд. – СПб.: Питер, 2018г. – 928 с.

Приложение

Текст программы

Программные файлы:

Заголовочные файлы:

* pipe.h;

Файлы реализации:

* main.cpp;

**Содержимое файла «main.cpp» и «pipe.h»:**

#include <SFML/Graphics.hpp>

#include <SFML/Audio.hpp>

#include <time.h>

#include <iostream>

#include "pipe.h"

using namespace sf;

const int N = 6;

int ts = 54; //tile size

Vector2f offset(65, 55);

int sizeOn = 0;

int size = 0;

pipe grid[N][N];

pipe &cell(Vector2i v) { return grid[v.x][v.y]; }

bool isOut(Vector2i v) { return !IntRect(0, 0, N, N).contains(v); }

void generatePuzzle()

{

std::vector<Vector2i> nodes;

nodes.push\_back(Vector2i(rand() % N, rand() % N));

while (!nodes.empty())

{

int n = rand() % nodes.size();

Vector2i v = nodes[n];

Vector2i d = DIR[rand() % 4]; // Random side of pipe

if (cell(v).dirs.size() == 3)

{

nodes.erase(nodes.begin() + n);

continue;

}

if (cell(v).dirs.size() == 2)

if (rand() % 50)

continue;

bool complete = 1;

for (auto D : DIR)

if (!isOut(v + D) && cell(v + D).dirs.empty())

complete = 0;

if (complete)

{

nodes.erase(nodes.begin() + n);

continue;

}

if (isOut(v + d))

continue;

if (!cell(v + d).dirs.empty())

continue;

cell(v).dirs.push\_back(d);

cell(v + d).dirs.push\_back(-d);

nodes.push\_back(v + d);

}

}

void drop(Vector2i v)

{

if (cell(v).on)

return;

cell(v).on = true;

for (auto d : DIR)

if (!isOut(v + d))

if (cell(v).isConnect(d) && cell(v + d).isConnect(-d))

drop(v + d);

}

enum GameMode

{

START,

GAME,

WIN

};

int main()

{

srand(time(0));

GameMode state = START;

RenderWindow app(VideoMode(390, 390), "The Pipe Puzzle!");

Texture t1, t2, t3, t4;

t1.loadFromFile("images/bg\_menu.png");

t2.loadFromFile("images/comp.png");

t3.loadFromFile("images/server.png");

t4.loadFromFile("images/pipes.png");

t4.setSmooth(true);

Sprite sBackground(t1), sComp(t2), sServer(t3), sPipe(t4);

sPipe.setOrigin(27, 27);

sComp.setOrigin(18, 18);

sServer.setOrigin(20, 20);

generatePuzzle();

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

{

pipe &p = grid[j][i];

for (int n = 4; n > 0; n--) //find orientation//

{

std::string s = "";

for (auto d : DIR)

s += p.isConnect(d) ? '1' : '0';

if (s == "0011" || s == "0111" || s == "0101" || s == "0010")

p.orientation = n;

p.rotate();

}

for (int n = 0; n < rand() % 4; n++) //shuffle//

{

grid[j][i].orientation++;

grid[j][i].rotate();

}

}

Vector2i servPos;

while (cell(servPos).dirs.size() == 1)

{

servPos = Vector2i(rand() % N, rand() % N);

}

sServer.setPosition(Vector2f(servPos \* ts));

sServer.move(offset);

Music music;

music.openFromFile("music.wav");

music.play();

music.setLoop(true);

while (app.isOpen())

{

if (state == WIN)

{

t1.loadFromFile("images/bg\_win.png");

Sprite sBackground(t1);

}

Event e;

while (app.pollEvent(e))

{

if (e.type == Event::Closed)

app.close();

if (e.type == Event::MouseButtonPressed)

if (e.key.code == Mouse::Left)

{

if (state == START)

{

state = GAME;

t1.loadFromFile("images/background.png");

Sprite sBackground(t1);

continue;

}

if (state == WIN)

app.close();

Vector2i pos = Mouse::getPosition(app) + Vector2i(ts / 2, ts / 2) - Vector2i(offset);

pos /= ts;

if (isOut(pos))

continue;

cell(pos).orientation++;

cell(pos).rotate();

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

grid[j][i].on = 0;

drop(servPos);

}

}

app.clear();

app.draw(sBackground);

if (state == GAME)

{

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

{

pipe &p = grid[j][i];

int kind = p.dirs.size();

if (kind == 2 && p.dirs[0] == -p.dirs[1])

kind = 0;

p.angle += 5;

if (p.angle > p.orientation \* 90)

p.angle = p.orientation \* 90;

sPipe.setTextureRect(IntRect(ts \* kind, 0, ts, ts));

sPipe.setRotation(p.angle);

sPipe.setPosition(j \* ts, i \* ts);

sPipe.move(offset);

app.draw(sPipe);

if (kind == 1)

{

if (p.on)

sComp.setTextureRect(IntRect(53, 0, 36, 36));

else

{

size++;

sComp.setTextureRect(IntRect(0, 0, 36, 36));

}

sComp.setPosition(j \* ts, i \* ts);

sComp.move(offset);

app.draw(sComp);

}

}

if (size == 0)

state = WIN;

std::cout << size << std::endl;

size = 0;

app.draw(sServer);

}

if (state == START)

app.draw(sBackground);

app.display();

}

return 0;

}

**Содержимое файла «pipe.h»:**

#include <SFML/Graphics.hpp>

using namespace sf;

Vector2i Up(0, -1);

Vector2i Down(0, 1);

Vector2i Right(1, 0);

Vector2i Left(-1, 0);

Vector2i DIR[4] = {Up, Right, Down, Left};

struct pipe

{

std::vector<Vector2i> dirs;

int orientation;

float angle;

bool on;

pipe() { angle = 0; }

void rotate()

{

for (int i = 0; i < dirs.size(); i++)

if (dirs[i] == Up)

dirs[i] = Right;

else if (dirs[i] == Right)

dirs[i] = Down;

else if (dirs[i] == Down)

dirs[i] = Left;

else if (dirs[i] == Left)

dirs[i] = Up;

}

bool isConnect(Vector2i dir)

{

for (auto d : dirs)

if (d == dir)

return true;

return false;

}

};