

Санкт-Петербургский Политехнический Университет Петра Великого  
Институт компьютерных наук и технологий  
Кафедра компьютерных систем и программных технологий

# Программирование

Отчет по лабораторной работе  
Симулятор звездной системы

**Работу выполнил:**

Ильин А. А.

Группа: 13501/4

**Преподаватель:**

Вылегжанина К.Д.

Санкт-Петербург  
2016

# Содержание

<b>1</b>	<b>Навигатор по лабиринту</b>	<b>2</b>
<b>2</b>	<b>Проектирование приложения</b>	<b>2</b>
<b>3</b>	<b>Реализация навигатора по лабиринту</b>	<b>3</b>
3.1	Работа приложения в консоли . . . . .	3
<b>4</b>	<b>Примеры работы приложения</b>	<b>3</b>
<b>5</b>	<b>Выводы</b>	<b>5</b>
<b>6</b>	<b>Приложение 1</b>	<b>5</b>

# 1 Навигатор по лабиринту

Навигатор по лабиринту - полноценная игра, цель которой - найти выход из лабиринта.

## Задание

Необходимо написать программу, которая сможет перемещать игрока по лабиринту.

## Концепция

Успешная программа будет загружать из файла лабиринт и выводить его на экран с возможностью прохождения.

## Минимально работоспособный продукт (MVP)

Консольное приложение, способное вывести на экран лабиринт, а так же перемещать игрока

## UML-модели

На диаграмме показано, что можно осуществить пользователю во время запуска приложения.

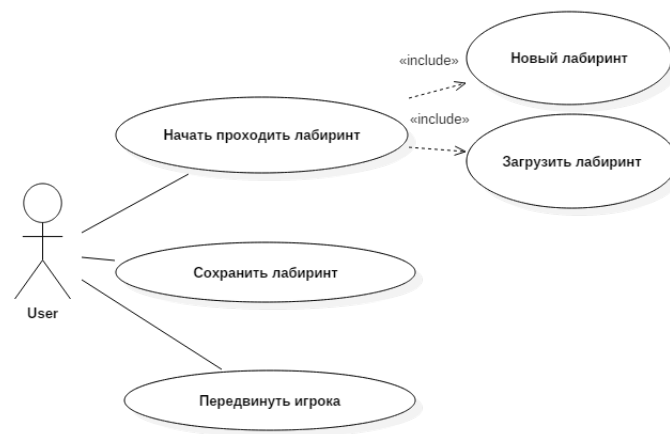


Рис. 1: Диаграмма прецендентов использования для интерфейса

# 2 Проектирование приложения

Для реализации проекта использовались 2 подпроекта:

- **Lib** - ядро, в котором реализованы функции, для работы с лабиринтом(является библиотекой для под проектов, отвечающих за взаимодействие с пользователем).
- **App** - данный подпроект был реализован с целью взаимодействия с пользователем через консоль.

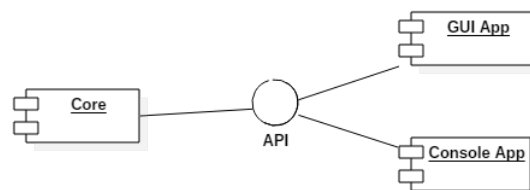


Рис. 2: Диаграмма компонентов

Библиотека предоставляет собой следующую функциональность:

- **void move(std::string command)** перемещение игрока по лабиринту
- **void fread()** чтение лабиринта из файла
- **void findplayer()** поиск игрока в лабиринте

**Вывод:** Использование под проекта, как библиотеки очень полезно и удобно.

### 3 Реализация навигатора по лабиринту

Операционная система Debian(32-bit). Для создания проекта использовались Qt Creator 3.5.0 (opensource) и GCC.

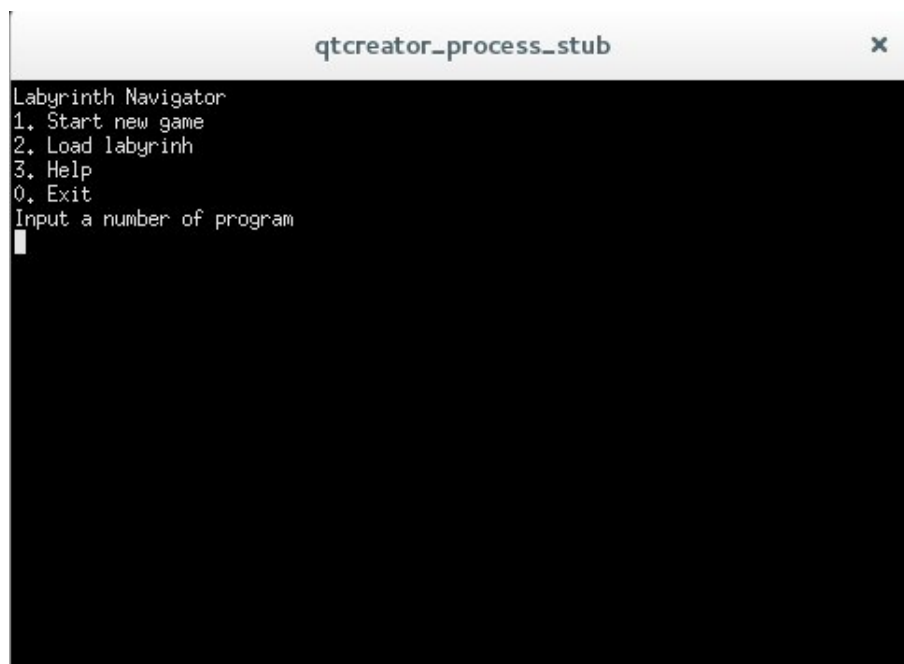
#### 3.1 Работа приложения в консоли

Было решено выделить один класс **App** для реализации взаимодействия с пользователем по средству консоли.

### 4 Примеры работы приложения

Ниже предоставлены снимки экрана, демонстрирующие основную функциональность консольного приложения:

При открытии консольного приложения перед пользователем возникает следующее.



```
qtcreeator_process_stub
Labyrinth Navigator
1. Start new game
2. Load labyrinth
3. Help
0. Exit
Input a number of program
█
```

Рис. 3: Главное меню в консоли

Для примера, был выбран пункт "1. Start new game". Далее пользователь может ввести команды для манипуляции над игроком

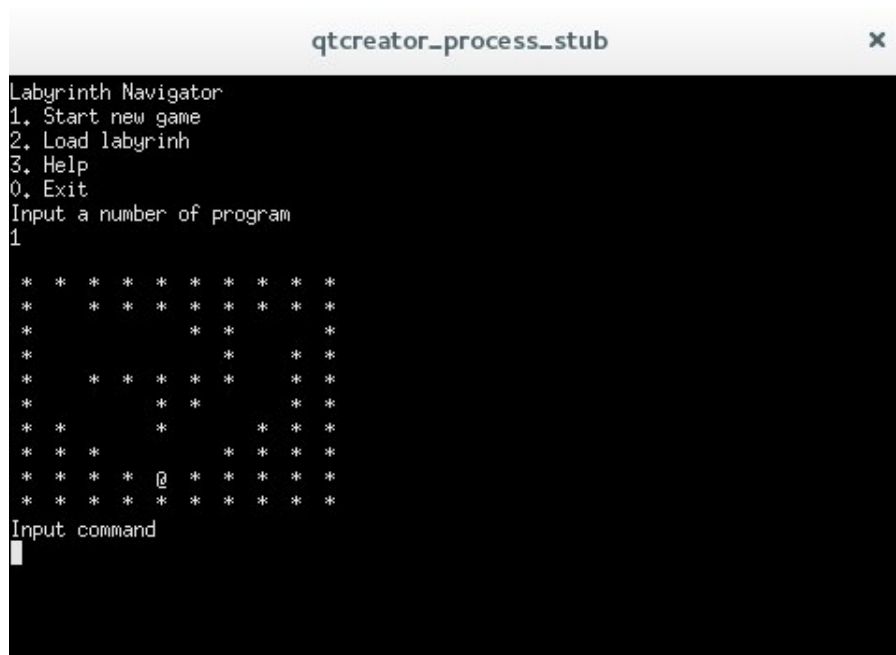


Рис. 4: Новая игра

Показан пример действия команды "up".

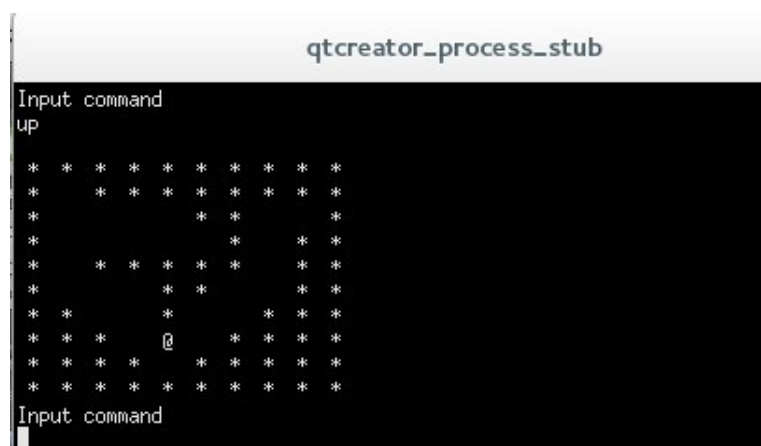


Рис. 5: Game command

Показан пример действия пункта "3. Help" для предоставления помощи.

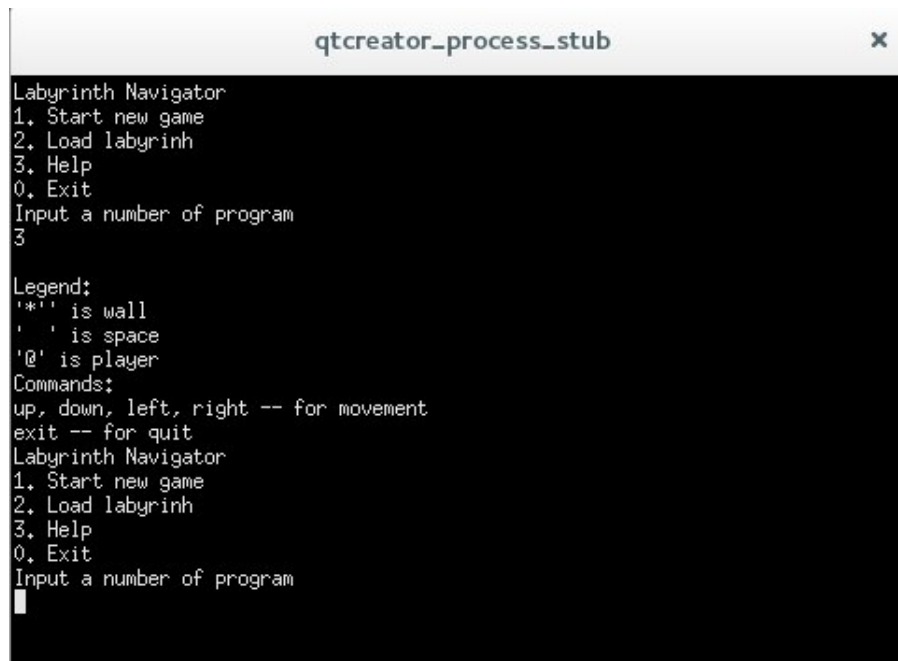


Рис. 6: Help Menu

## 5 Выводы

Был написан проект "Навигатор по лабиринту" который перемещает игрока по лабиринту. Была проделана работа по созданию нескольких классов для выполнения задачи. В итоге был реализован MVP.

## 6 Приложение 1

### Листинги

main.cpp

```
1 #include <stdio.h>
2 #include "app.h"
3
4 int main(void)
5 {
6     app application;
7     application.start();
8     return 0;
9 }
```

app.cpp

```
1 #include "app.h"
2
3
4
5 void app::printlabyrinth()
6 {
7
8     for (int i = 0; i < 10; i++)
9     {
10         for (int j = 0; j < 10; j++)
11             if (labyrinth.maze[i][j] == 1)
12                 std::cout << " * ";
13             else if (labyrinth.maze[i][j] == 2) std::cout << " @ ";
14             else std::cout << "   ";
15         std::cout << "\n";
16     }
17 }
18
19 void app::printMenu()
```

```

20 {
21     std::cout << "Labyrinth_Navigator" << "\n";
22     std::cout << "1._Start_new_game" << "\n";
23     std::cout << "2._Load_labyrinth" << "\n";
24     std::cout << "3._Help" << "\n";
25     std::cout << "0._Exit" << "\n";
26 }
27
28 void app::start()
29 {
30     while (true)
31     {
32         labyrinth.fread();
33         printMenu();
34
35         std::cout << "Input_a_number_of_program" << std::endl;
36         std::cin >> choice;
37         std::cout << std::endl;
38
39         menu_choice();
40     }
41 }
42
43 void app::menu_choice()
44 {
45     switch (choice){
46
47     case 1:
48         game();
49         break;
50
51     case 2:
52         std::cout << "Dunno_wtf_is_it" << std::endl;
53         break;
54     case 3:
55         help();
56         break;
57     case 0:
58         exit(0);
59         break;
60
61     default:
62     {
63         std::cout << "Wrong_number._Input_another_number" << std::endl << std::endl;
64     }
65 }
66
67 void app::help()
68 {
69     std::cout << "Legend:" << std::endl;
70     std::cout << "'*' is wall\n' ' is space\n'@' is player" << std::endl;
71     std::cout << "Commands:" << std::endl;
72     std::cout << "up,down,left,right _ _ for movement" << std::endl;
73     std::cout << "exit _ _ for quit" << std::endl;
74 }
75
76
77
78 void app::game()
79 {
80     while (true)
81     {
82         printlabyrinth();
83         std::cout << "Input_command" << std::endl;
84         std::cin >> command;
85         std::cout << std::endl;
86
87         if(command == "exit") exit(0);
88         if(command == "up" || command == "down" || command == "left" || command == "right"
89         → ) labyrinth.move(command);
90         else{ std::cout << "Wrong_command._Input_another_command" << std::endl << std::
91         → endl;};
92     }
93 }

```

## app.h

```
1 #ifndef APP_H
2 #define APP_H
3 #include "lib.h"
4
5 #include <fstream>
6 #include <string>
7 #include <iostream>
8
9 class app
10 {
11 public:
12     Lib labyrinth;
13     std::string command;
14     int choice;
15     void printlabyrinth();
16     void printMenu();
17     void start();
18     void menu_choice();
19     void game();
20     void help();
21 };
22
23 #endif // APP_H
```

## lib.h

```
1 #ifndef LIB_H
2 #define LIB_H
3
4 #include <string>
5 #include <fstream>
6 #include <iostream>
7 class Lib
8 {
9
10 public:
11     Lib();
12     int x; int y;
13     int** maze;
14     void move(std::string command);
15     void fread();
16     void findplayer();
17     ~Lib();
18 };
19
20 #endif // LIB_H
```

## lib.cpp

```
1 #include "lib.h"
2
3
4 Lib::Lib()
5 {
6     maze = new int* [10];
7     for (int i = 0; i < 10; i++)
8         maze[i] = new int [10];
9 }
10 void Lib::move(std::string command)
11 {findplayer();
12     if(command == "up" && maze[x-1][y] == 0) {maze[x][y] = 0; maze[x-1][y] = 2;} else
13     if(command == "down" && maze[x+1][y] == 0) {maze[x][y] = 0; maze[x+1][y] = 2;} else
14     if(command == "left" && maze[x][y-1] == 0) {maze[x][y] = 0; maze[x][y-1] = 2;} else
15     if(command == "right" && maze[x][y+1] == 0) {maze[x][y] = 0; maze[x][y+1] = 2;}
16     else {std::cout << "You can't do it" << std::endl << std::endl;}
17 }
18 void Lib::fread()
19 {
20     std::ifstream lab("file.txt");
21     for (int i = 0; i < 10; ++i)
22         for (int j = 0; j < 10; ++j)
23             lab>>maze[i][j];
24     lab.close();
25 }
```



```
25 | }
26 |
27 | void Lib::findplayer()
28 | {
29 |     for (int i = 0; i < 10; ++i)
30 |         for (int j = 0; j < 10; ++j)
31 |             if (maze[i][j] == 2) {x = i;
32 |                 y = j;
33 |             };
34 | }
35 |
36 | Lib::~~Lib()
37 | {
38 |     for (int i = 0; i < 10; i++) {
39 |         delete [] maze[i];
40 |     }
41 |     delete [] maze;
42 | }
```