|  |
| --- |
| МИНОБРНАУКИ РОССИИ  Федеральное государственное  бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

Институт информационных технологий

Кафедра корпоративных информационных систем

|  |  |  |
| --- | --- | --- |
| **ОТЧЕТ**  **по лабораторной работе №3** | | |
| **по дисциплине** | | |
| **«Объектно-ориентированное программирование»**  **Тема лабораторной работы: «**Классы. Диаграммы классов в языке UML.**»** | | |
| Студент группы | ИКБО-07-18 | Бабух А.А. |
| Принял | ассистент кафедры | Колесников С.Н. |
|  |  |  |
| Выполнено | «\_\_\_» \_\_\_\_\_\_ 201\_\_ г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* |
|  |  | *(подпись студента)* |
| Зачтено | «\_\_\_» \_\_\_\_\_\_ 201\_\_ г. | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* |
|  |  | *(подпись преподавателя)* |

1. **Постановка задачи**
   1. **Задание №1**
      * 1. Реализовать классы в соответствии с предложенной диаграммой.
        2. В каждый класс включите конструкторы, необходимые для заполнения атрибутов класса (реализуйте проверку переданных в конструктор параметров).
        3. Рекомендации по доработке функциональности программы:
   2. В базовой версии программы все объекты, предусмотренные в ней (пользователи, заявки, вопросы тестов и т.д.) могут только создаваться, но не редактироваться. Однако реальная программа, конечно, должна содержать возможность просматривать и изменять параметры существующих объектов (например, изменить пароль пользователя, поменять заявку до того, как она была обработана и т.д.). Рекомендуется реализовать соответствующую возможность. Для этого нужно разработать интерфейс, позволяющий выбрать один из существующих объектов и изменить его, а также добавить в нужный класс методы для изменения соответствующих атрибутов.
   3. Классы, необходимые для построения пользовательского интерфейса, на предложенных диаграммах отсутствуют. В целом предложенные диаграммы не являются догмой и могут быть усовершенствованы (или вообще изменены).
   4. В каждой программе есть пользователи двух типов: администратор и клиенты. Предполагается, что администратор в программе один. Он должен быть добавлен сразу после первого запуска программы.

Вариант №2: Написать программу для составления оптимального маршрута разносчика пиццы. Пользователи, желающие заказать пиццу, оставляют в системе заявку, указывая район своего проживания. Кроме того система постоянно накапливает информацию о способах передвижения между соседними районами города. Когда поступает заказ, программа должна найти короткий путь от текущего местонахождения вагончика с пиццей до района проживания заказчика.

Основные алгоритмы: нахождение кратчайшего маршрута.

Интерфейсы:

* Интерфейс для создания «карты» (на самом деле это просто граф) города.
* Интерфейс для входа в систему.
* Интерфейс для регистрации пользователей.
* Интерфейс для заполнения заявки.

Интерфейс для обработки результатов.

Таблицы данных:

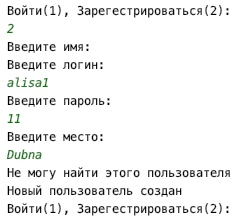
* Пользователи
* Карта города (районы и связи между ними)

1. **Описание используемых классов**

* Main.java: находится интерфейс программы;
* Admin.java: находятся поля – имя, логин и пароль админа;
* Client.java: находятся поля – имя, логин и пароль клиента, метод ввода местности;
* Map.java: метод нахождения кратчайшего пути на местности;
* Path.java: находятся поля – время и узлы;
* User.java: метод заполнения полей юзера – имя, логин и пароль.

1. **Пользовательский интерфейс**

Пользователь имеет возможность регистрации или вхождения в программу, а затем уже выполнения дальнейших действий. (Смотреть Рисунок №1)



**Рисунок 1 –** Пользовательский интерфейс

1. **Листинг программы**

**Admin.java**

package com.company;

public class Admin extends User {

public Admin(){}

public Admin(String name, String login, String password){

this.name = name;

this.login = login;

this.password = password;

}

}

**Client.java**

package com.company;

public class Client extends User

{

protected String place;

public Client(){}

public Client(String name, String login, String password)

{

this.name = name;

this.login = login;

this.password = password;

}

public void setPlace(String place)

{

this.place=place;

}

public String getPlace()

{

return place;

}

}

**Map.java**

package com.company;

import java.util.ArrayList;

import java.util.List;

public class Map

{

List<Path> paths = new ArrayList<>();

public void addPath(String place1, String place2, int time)

{

paths.add(new Path(new String[]{place1, place2},time));

}

public List<String> getAllPlaces()

{

List<String> places = new ArrayList<>();

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

{

String place1 = paths.get(i).points[0];

String place2 = paths.get(i).points[1];

if(!places.contains(place1)){places.add(place1);}

if(!places.contains(place2)){places.add(place2);}

}

return places;

}

public void fill()

{

paths.add(new Path(new String[]{"1", "2"},1));

paths.add(new Path(new String[]{"2", "5"},6));

paths.add(new Path(new String[]{"2", "3"},1));

paths.add(new Path(new String[]{"1", "3"},3));

paths.add(new Path(new String[]{"3", "4"},2));

paths.add(new Path(new String[]{"4", "5"},2));

}

int time = 0;

int min\_time =Integer.MAX\_VALUE;

List<String> shortest\_way = new ArrayList<>();

List<String> curr\_way = new ArrayList<>();

public List<String> get\_shortest\_way(String place1, String place2)

{

generateShortWay(place1, place2,0);

return shortest\_way;

}

private void fill\_shortest(List<String> list)

{

shortest\_way.clear();

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

{

shortest\_way.add(list.get(i));

}

}

public void generateShortWay(String place1, String place2, int last\_time)

{

curr\_way.add(place1);

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

{

String start\_place = paths.get(i).points[0];

String end\_place = paths.get(i).points[1];

if(paths.get(i).points[0].equals(place1))

{

int new\_time = paths.get(i).time;

if(end\_place.equals(place2))

{

time += new\_time;

curr\_way.add(end\_place);

if(time < min\_time)

{

min\_time = time;

fill\_shortest(curr\_way);

}

time -= new\_time;

curr\_way.remove(curr\_way.size()-1);

}

else

{

time += new\_time;

if (time >= min\_time)

{

time -= last\_time;

curr\_way.remove(curr\_way.size() - 1);

return;

}

generateShortWay(end\_place, place2, new\_time);

}

}

}

time -= last\_time;

curr\_way.remove(curr\_way.size() - 1);

return;

}

}

**Path.java**

package com.company;

public class Path

{

public String[] points;

public int time;

public Path(String[] points, int time)

{

this.points=points;

this.time = time;

}

}

**Main.java**

package com.company;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

import java.lang.String;

import static java.lang.Integer.parseInt;

public class PizzaSystem

{

private List<User> users = new ArrayList<>();

private Map map = new Map();

private User currentUser;

private String currentLocation;

private Scanner in = new Scanner(System.in);

private boolean end = false;

public static void main(String[] args)

{

PizzaSystem p = new PizzaSystem();

p.users.add(new Admin("Admin", "admin", "1"));

p.currentUser = p.users.get(0);

while(!p.end)

{

p.mainMenu();

if (p.currentUser.getClass() == Client.class)

{

Client client = (Client)p.currentUser;

p.clientMenu(client);

}

else if (p.currentUser.getClass() == Admin.class)

p.adminMenu();

}

}

private void mainMenu(){

boolean exit=false;

while (!exit) {

System.out.println("Войти(1), Зарегестрироваться(2):");

int choice = in.nextInt();

while (choice==1) {

System.out.println("Если хотите выйти введите exit:");

String goBack = in.nextLine();

if(goBack.equals("exit")){

break;

}

else {

System.out.println("Введите логин: ");

String curLogin = in.nextLine();

System.out.println("Введите пароль: ");

String curPassword = in.nextLine();

User CurUser = findUser(curLogin, curPassword);

if (CurUser != null) {

currentUser = CurUser;

choice = 0;//exit

exit=true;

} else {

System.out.println("Попробуйте еще.");

}

}

}

if (choice==2){

System.out.println("Введите имя: ");

String curName = in.nextLine();

System.out.println("Введите логин: ");

String curLogin = in.nextLine();

System.out.println("Введите пароль: ");

String curPassword = in.nextLine();

System.out.println("Введите место: ");

String curPlace = in.nextLine();

addUser(curName,curLogin,curPassword,curPlace);

}

}

}

private void clientMenu(Client client){

boolean exit = false;

while (!exit) {

System.out.println("Введите команду (add, exit)");

String cm = in.nextLine();

switch (cm) {

case "add":

System.out.println("Введите место: ");

String place1 = in.nextLine();

System.out.println("Введите время : ");

String time= in.nextLine();

map.addPath(place1,client.getPlace(), parseInt(time));

break;

case "exit":

exit=true;

break;

default:

System.out.println("Неизвестная команда");

break;

}

}

}

private void adminMenu(){

boolean exit = false;

while (!exit) {

System.out.println("Введите команду (way, exit)");

String cm = in.nextLine();

switch (cm) {

case "way":

map.fill();

System.out.println("точка А : ");

String A= in.nextLine();

System.out.println("точка В: ");

String B= in.nextLine();

List<String> results = map.get\_shortest\_way(A,B);

end = true;

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

{

System.out.println((i+1)+". " + results.get(i));

}

case "exit":

exit=true;

break;

default:

System.out.println("Неизвестная команда");

break;

}

}

}

private void addUser(String name, String login, String password,String curPlace)

{

User user = findUser(login,password);

if(user==null){Client client = new Client(name,login,password); client.setPlace(curPlace); users.add(client); System.out.println("Новый пользователь создан"); }

else {System.out.println("Уже зарегестрирован");}

}

private User findUser(String login, String password){

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

{

if(users.get(i).enter(login,password)){return users.get(i);}

}

System.out.println("Не могу найти этого пользователя");

return null;

}

}

**User.java**

package com.company;

public class User

{

protected String name;

protected String login;

protected String password;

public User(){}

public User(String name, String login, String password){

this.name = name;

this.login = login;

this.password = password;

}

boolean enter(String login, String password) {

if (this.login.equals(login) && this.password.equals(password))

{

return true;

}

else return false;

}

String getName()

{

return name;

}

}