Міністерство освіти і науки України

Національний університет «Львівська політехніка»



**Лабораторна робота №12**

на тему:

«Основи програмування Java»

з курсу:

«ООР»

Виконав:

гр. КН-110

Шарко Іван

Прийняв:

Гасько Р.Т.

Львів 2018

*Лабораторна робота № 12*

***Мета роботи:*** розширити вміння працювати з мовою Java, розглянути основні функції мови Java.

**Програмний код**

**AutoStation.java**

**import** java.util.Scanner;

**public** **class** AutoStation {

**private** **static** Scanner *scanner* = **new** Scanner(System.***in***);

**public** **static** **void** main(String[] args){

Data information = **new** Data();

**boolean** flag, secondFlag = **false**;

String numberInput;

System.***out***.println(" <>-----------------<> WELCOME TO THE 'LVIV HEART' AUTO STATION! <>----------------<> \n");

//Terminal "Home Screen"

**while**(**true**) {

System.***out***.println("Please, input the city to see more information: 'Kyiv', 'Rivne', 'Lublin', 'Paris' \n" +

" [tickets] - Your tickets \n [time] - When your transport will dispatch \n " +

" [sell] - sell all tickets and leave \n [exit] - Exit program");

Data.*direction* = *scanner*.next();

Data.*cityMatcher*(Data.*direction*);

System.***out***.println("<>---------------<> " + Data.*direction* + " <>---------------<>");

**if**(Data.*direction*.equals("tickets")){

information.iterator();

**continue**;

} **else** **if**(Data.*direction*.equals("time")) {

**double** minutesLeft = 10 - (information.counter / 60);

**if**(minutesLeft <= 1){

System.***out***.println("hurry up, there's only " + minutesLeft \* 60 + " seconds left! \n");

} **else** **if**(minutesLeft < 10){

System.***out***.println("Dispatching in " + minutesLeft

+ " minutes. \n");

}

**continue**;

} **else** **if**(Data.*direction*.equals("exit")){

**break**;

} **else** **if**(Data.*direction*.equals("No city chosen")){

**continue**;

} **else** **if**(Data.*direction*.equals("sell")) {

//cancel after sometime

information.sell();

information.cancelTimer();

System.***out***.println("TimerTask cancelled");

**continue**;

}

flag = **true**;

**while** (flag) {

System.***out***.println(" [0] - Seats left \n [1] - What's the price \n " +

"[2] - Confirm the city and buy tickets \n [3] - Back \n");

numberInput = *scanner*.next();

**switch** (numberInput) {

**case** "0":

System.***out***.println("There are/is " + information.freeSpace(Data.*direction*) + " seat/seats left.");

**break**;

**case** "1":

System.***out***.println("The price is " + information.seePrice(Data.*direction*) + " UAH.");

**break**;

**case** "2":

System.***out***.println("You are heading to " + information.whereToGo(Data.*direction*) + ".");

secondFlag = **true**;

**break**;

**case** "3":

flag = **false**;

**break**;

**default**:

System.***out***.println("Wrong input, retry:");

**break**;

}

//Buying process:

**while** (secondFlag) {

**int** money;

System.***out***.println("How many tickets do you want? (" + information.freeSpace(Data.*direction*) + " left) \n");

**do**{

information.seats = *scanner*.nextInt();

**if** (information.seats > information.freeSpace(Data.*direction*)) {

System.***out***.println("There are not so many seats left, sorry.");

} **else** **if** (information.seats == -1) {

secondFlag = **false**;

**break**;

}

} **while** (information.seats > information.freeSpace(Data.*direction*));

**if**(!secondFlag){

**break**;

}

**int** localPrice = information.priceCounter(information.price, information.seats);

System.***out***.println("Pay " + localPrice + " UAH for the ticket: ");

money = *scanner*.nextInt();

**if** (money == -10) {

secondFlag = **false**;

**break**;

}

**if** (information.buyTicket(money) >= 0) {

**if**(information.buyTicket(money) == 0) {

System.***out***.println("Thank you and good luck!");

} **else** {

System.***out***.println("Payback of " + information.buyTicket(money) + " UAH. Thank you and good luck!");

}

**switch** (Data.*direction*){

**case** "Kyiv":

information.kyivSeats = information.kyivSeats - information.seats;

**break**;

**case** "Rivne":

information.rivneSeats = information.rivneSeats - information.seats;

**break**;

**case** "Lublin":

information.lublinSeats = information.lublinSeats - information.seats;

**break**;

**case** "Paris":

information.parisSeats = information.parisSeats - information.seats;

**break**;

}

Papers ticket = **new** Papers(Data.*direction*, information.seats);

information.ticketDataBase.add(ticket);

information.seats = 0;

//Starting timer

**if**(information.timerLaunched) {

information.cancelTimer();

information.timerLaunched = **false**;

}

information.cancelTimer();

information.startTimer();

System.***out***.println("Hurry up, 10 minutes left before Your departure! \n");

secondFlag = **false**;

flag = **false**;

**break**;

} **else** {

System.***out***.println("Not enough money. input -10 to Exit");

}

}

}

}

}

}

**Data.java**

**import** java.util.\*;

**import** java.util.regex.Matcher;

**import** java.util.regex.Pattern;

**public** **class** Data{

**public** **double** counter;

**public** **boolean** timerLaunched = **false**;

**private** Timer dispatchTimer = **new** Timer();

**private** TimerTask timerTask = **new** TimerTask() {

@Override

**public** **void** run() {

counter++;

}

};

**public** **int** price = 100000000, seats = 100000000;

**public** **static** String *direction*;

**public** **int** kyivSeats = 3, rivneSeats = 50, lublinSeats = 30, parisSeats = 5;

//creating ArrayList for storing Tickets.

**public** ArrayList <Papers> ticketDataBase = **new** ArrayList <>();

//iterator method

**public** **void** iterator(){

Iterator itr = ticketDataBase.iterator();

//traverse elements of ArrayList object

**while**(itr.hasNext()){

Papers t = (Papers) itr.next();

System.***out***.println(t.destination + ": " + t.seatsNumber + " tickets");

}

}

**public** **void** startTimer(){

dispatchTimer.scheduleAtFixedRate(timerTask, 0, 1000);

timerLaunched = **true**;

}

**public** **void** cancelTimer(){

dispatchTimer.cancel();

dispatchTimer = **new** Timer();

counter = 0;

timerTask = **new** TimerTask() {

@Override

**public** **void** run() {

counter++;

}

};

}

**public** **int** priceCounter(**int** money, **int** seats){

price = seats \* money;

**return** price;

}

**public** String whereToGo(String direction){

**switch**(direction){

**case** "Kyiv":

price = 250;

**return** direction;

**case** "Rivne":

price = 150;

**return** direction;

**case** "Lublin":

price = 1000;

**return** direction;

**case** "Paris":

price = 2000;

**return** direction;

**default**:

**break**;

}

**return** "Unknown";

}

**public** **int** seePrice(String direction){

**int** price;

**switch** (direction){

**case** "Kyiv":

price = 250;

**return** price;

**case** "Rivne":

price = 150;

**return** price;

**case** "Lublin":

price = 1000;

**return** price;

**case** "Paris":

price = 2000;

**return** price;

**default**:

**break**;

}

**return** 0;

}

**public** **int** buyTicket(**int** money){

**int** payback;

**if** (money >= price) {

payback = money - price;

**return** payback;

} **else** {

**return** -1;

}

}

**public** **void** sell(){

ticketDataBase.clear();

}

**public** **int** timeToGo(String direction){

**int** time;

**switch**(direction){

**case** "Kyiv":

time = 20;

**return** time;

**case** "Rivne":

time = 10;

**return** time;

**case** "Lublin":

time = 120;

**return** time;

**case** "Paris":

time = 60;

**return** time;

**default**:

**return** 0;

}

}

**public** **int** freeSpace(String direction){

**switch**(direction){

**case** "Kyiv":

**return** kyivSeats;

**case** "Rivne":

**return** rivneSeats;

**case** "Lublin":

**return** lublinSeats;

**case** "Paris":

**return** parisSeats;

**default**:

**return** 0;

}

}

//to locate mistakes in spelling and recognize the city

**public** **static** **void** cityMatcher(String input){

String kyiv = "\\b(?:Kyiv|kyiv|kuiv|Kuiv|kiev|Kiev)\\b";

String rivne = "\\b(?:Rivne|rivne|Rovno|rovno)\\b";

String lublin = "\\b(?:Lublin|lublin|lyblin|Lyblin|Liblin|liblin|lablin|Lablin)\\b";

String paris = "\\bParis|paris|parizh|Parizh\\b";

Pattern patternKyiv = Pattern.*compile*(kyiv);

Pattern patternRivne = Pattern.*compile*(rivne);

Pattern patternLublin = Pattern.*compile*(lublin);

Pattern patternParis = Pattern.*compile*(paris);

Matcher matcherKyiv = patternKyiv.matcher(input);

Matcher matcherRivne = patternRivne.matcher(input);

Matcher matcherLublin = patternLublin.matcher(input);

Matcher matcherParis = patternParis.matcher(input);

**if**(matcherKyiv.find()){

*direction* = "Kyiv";

} **else** **if**(matcherLublin.find()){

*direction* = "Lublin";

} **else** **if**(matcherParis.find()) {

*direction* = "Paris";

} **else** **if**(matcherRivne.find()) {

*direction* = "Rivne";

}

}

}

**Papers.java**

**public** **class** Papers{

String destination;

Integer seatsNumber;

Papers(String destination, Integer seatsNumber){

**this**.destination = destination;

**this**.seatsNumber = seatsNumber;

}

}