Учреждение образования

«Белорусский государственный университет информатики и

радиоэлектроники»

ОТЧЕТ

по лабораторной работе №2

# "Технология Java TM IDL. Архитектура CORBA"

Выполнила:

студентка группы 714301

факультета КП

Забелло К.В.

Проверил:

Кунцевич А.А

Минск 2020

**Ход выполнения работы**

**Индивидуальное задание.**

1. ЖД билеты(тип, номер поезда, цена, класс)
   1. Вывод билетов с ценой меньше введённой
   2. Запись всех билетов в файл, на выбранный поезд

**Код программы:**

Класс RailwayTicket

package com.zabello.entities;  
  
import java.math.BigDecimal;  
import java.util.Objects;  
  
public class RailwayTicket {  
  
 private Long id;  
  
 private String type;  
  
 private Integer trainNumber;  
  
 private BigDecimal cost;  
  
 private String classOfService;  
  
 public RailwayTicket() {  
 }  
  
 public RailwayTicket(Long id, String type, Integer trainNumber, BigDecimal cost, String classOfService) {  
 this.id = id;  
 this.type = type;  
 this.trainNumber = trainNumber;  
 this.cost = cost;  
 this.classOfService = classOfService;  
 }  
  
 public Long getId() {  
 return id;  
 }  
  
 public void setId(Long id) {  
 this.id = id;  
 }  
  
 public String getType() {  
 return type;  
 }  
  
 public void setType(String type) {  
 this.type = type;  
 }  
  
 public Integer getTrainNumber() {  
 return trainNumber;  
 }  
  
 public void setTrainNumber(Integer trainNumber) {  
 this.trainNumber = trainNumber;  
 }  
  
 public BigDecimal getCost() {  
 return cost;  
 }  
  
 public void setCost(BigDecimal cost) {  
 this.cost = cost;  
 }  
  
 public String getClassOfService() {  
 return classOfService;  
 }  
  
 public void setClassOfService(String classOfService) {  
 this.classOfService = classOfService;  
 }  
  
 @Override  
 public boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null || getClass() != o.getClass()) return false;  
 RailwayTicket that = (RailwayTicket) o;  
 return id.equals(that.id) &&  
 type.equals(that.type) &&  
 trainNumber.equals(that.trainNumber) &&  
 cost.equals(that.cost) &&  
 classOfService.equals(that.classOfService);  
 }  
  
 @Override  
 public int hashCode() {  
 return Objects.*hash*(id, type, trainNumber, cost, classOfService);  
 }  
  
 @Override  
 public String toString() {  
 return "Id: " + id +  
 "\nType: " + type +  
 "\nTrainNumber: " + trainNumber +  
 "\nCost: " + cost +  
 "\nService class: " + classOfService+"\n";  
 }  
}

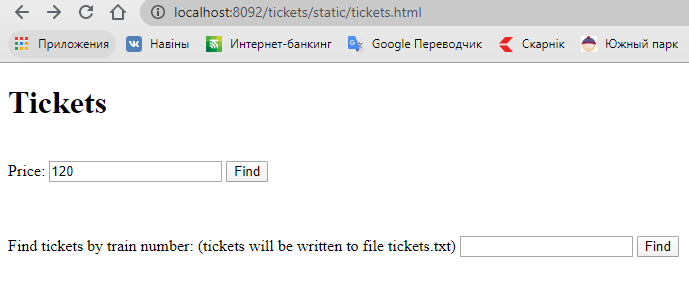
Класс Tickets

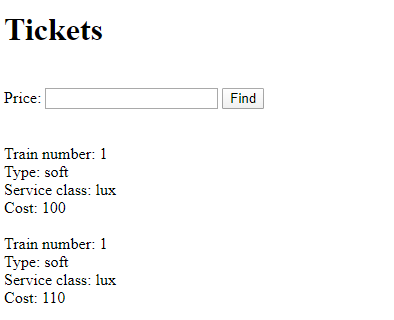
package com.zabello.sessions;  
  
import com.zabello.entities.RailwayTicket;  
  
import javax.ejb.Singleton;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.math.BigDecimal;  
import java.util.ArrayList;  
import java.util.Arrays;  
import java.util.List;  
  
@Singleton  
public class Tickets {  
  
 static ArrayList<RailwayTicket> *tickets* = new ArrayList<>(  
 Arrays.*asList*(  
 new RailwayTicket(1L, "soft", 1, new BigDecimal("100"), "lux"),  
 new RailwayTicket(2L, "soft", 1, new BigDecimal("110"), "lux"),  
 new RailwayTicket(3L, "super-soft", 2, new BigDecimal("120"), "lux"),  
 new RailwayTicket(4L, "soft", 3, new BigDecimal("110"), "lux")  
 )  
 );  
  
 public List<RailwayTicket> getTicketsByCost(BigDecimal cost) {  
 List<RailwayTicket> foundTickets = new ArrayList<>();  
 for (RailwayTicket ticket : *tickets*) {  
 if (ticket.getCost().compareTo(cost) < 0) {  
 foundTickets.add(ticket);  
 }  
 }  
 return foundTickets;  
 }  
  
 public void getTicketsByTrainNumber(Integer trainNumber) {  
 List<RailwayTicket> foundTickets = new ArrayList<>();  
 for (RailwayTicket ticket : *tickets*) {  
 if (ticket.getTrainNumber().compareTo(trainNumber) == 0) {  
 foundTickets.add(ticket);  
 }  
 }  
 writeTicketsToFile(foundTickets);  
 }  
  
  
 private void writeTicketsToFile(List<RailwayTicket> tickets) {  
 try (FileWriter writer = new FileWriter("E:/вучоба/РИС/labs/ris2/tickets.txt", false)) {  
 for (RailwayTicket ticket : tickets) {  
 writer.write(ticket.toString());  
 }  
 } catch (IOException e) {  
 e.getMessage();  
 }  
 }  
}

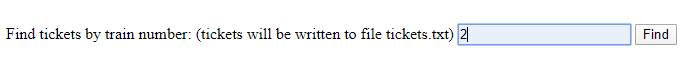
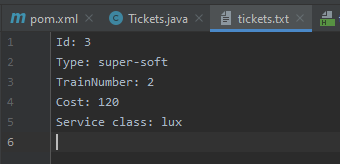
Класс TicketServlet

package com.zabello.sessions;  
  
import com.zabello.entities.RailwayTicket;  
  
import javax.ejb.EJB;  
import javax.servlet.ServletException;  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.math.BigDecimal;  
import java.util.List;  
  
@WebServlet(urlPatterns = {"/ticket-servlet"})  
public class TicketServlet extends HttpServlet {  
  
 @EJB  
 private Tickets tickets;  
  
 @Override  
 protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {  
  
 String button = req.getParameter("button");  
 if (button.equals("price\_btn")) {  
 BigDecimal cost = new BigDecimal(req.getParameter("price"));  
 List<RailwayTicket> ticketsList = tickets.getTicketsByCost(cost);  
 doResp(resp, ticketsList);  
 }  
 if (button.equals("number\_btn"))  
 tickets.getTicketsByTrainNumber(Integer.*parseInt*(req.getParameter("number")));  
 }  
  
 private void doResp(HttpServletResponse resp, List<RailwayTicket> tickets) throws ServletException, IOException {  
 PrintWriter writer = resp.getWriter();  
 writer.println("<!DOCTYPE html>\n" +  
 "<html lang=\"en\">\n" +  
 "<head>\n" +  
 " <meta charset=\"UTF-8\"/>\n" +  
 " <link rel=\"stylesheet\" href=\"style.css\"/>\n" +  
 " <title>Shop</title>\n" +  
 "</head>\n" +  
 "<body>\n" +  
 "<form action=\"TicketsServlet\" method=\"POST\">\n" +  
 " <h1>Tickets</h1>\n" +  
 " <br/>\n" +  
 " Price:\n" +  
 " <input name=\"price\"/>\n" +  
 " <button type=\"submit\" name=\"button\" value=\"price\">Find</button>\n" +  
 " <br/>\n" +  
 " <br/>\n");  
 if (!tickets.isEmpty()) {  
 for (RailwayTicket ticket : tickets) {  
 writer.println("<br>" +  
 "Train number: " + ticket.getTrainNumber()  
 + "<br> Type: " + ticket.getType()  
 + "<br> Service class: " + ticket.getClassOfService()  
 + "<br> Cost: " + ticket.getCost()  
 + "<br/>");  
 }  
  
 writer.println("<br/>\n" +  
 " <br/>\n" +  
 " Find tickets by train number:\n" +  
 " (tickets will be written to file tickets.txt)\n" +  
 " <input name=\"number\"/>\n" +  
 " <button type=\"submit\" name=\"button\" value=\"number\">Find</button>\n" +  
 " <br/>\n" +  
 " <br/>\n" +  
 " <br/>\n" +  
 "</form>\n" +  
 "</body>\n" +  
 "</html>");  
 } else {  
 writer.println("Tickets not found!" +  
 "<br/>\n" +  
 " <br/>\n" +  
 " Find tickets by train number:\n" +  
 " (tickets will be written to file tickets.txt)\n" +  
 " <input name=\"number\"/>\n" +  
 " <button type=\"submit\" name=\"button\" value=\"number\">Find</button>\n" +  
 " <br/>\n" +  
 " <br/>\n" +  
 " <br/>\n" +  
 "</form>\n" +  
 "</body>\n" +  
 "</html>");  
 }  
 }  
}

**Результаты выполнения программы:**





**Вывод.** В ходе данной лабораторной работы была изучена технология CORBA и спроектирована система на ее основе.