package intern;  
  
import appt.meta3.Ob0;  
import appt.meta3.Ob3;  
import appt.meta3.Obb;  
import appt.meta3.Util;  
import org.apache.commons.lang3.math.NumberUtils;  
  
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.nio.charset.StandardCharsets;  
import java.util.\*;  
import java.util.function.BiConsumer;  
  
public class EldarServlet extends HttpServlet {  
 private PrintWriter out;  
 private ResourceBundle mains;  
 private HttpServletRequest request;  
 private HttpServletResponse response;  
 private final int myId = 1000360;  
 private final String redis = "rev";  
 private final String prefix = "Eldar";  
  
 private static final Map<String, String> *typeMap* = new HashMap<>();  
 private static final Map<String, String> *yesnoMap* = new HashMap<>();  
 private static final Map<String, List<String>> *pagesMap* = new TreeMap<>();  
  
 static {  
 *typeMap*.put("", "-");  
 *typeMap*.put("0", "Экскурсия");  
 *typeMap*.put("1", "Билет");  
 *typeMap*.put("2", "Спорт");  
 *typeMap*.put("3", "Прокат");  
 *typeMap*.put("4", "Услуга");  
 *typeMap*.put("5", "СПА");  
 *typeMap*.put("6", "Авиация");  
 *typeMap*.put("8", "Концерт");  
  
 *yesnoMap*.put("", "-");  
 *yesnoMap*.put("1", "Да");  
 *yesnoMap*.put("0", "Нет");  
  
 *pagesMap*.put("PostgreSQL", Arrays.*asList*("Один", "Два",  
 "Три", "Четыре", "Пять", "Шесть", "Семь", "Восемь"));  
 *pagesMap*.put("Redis", Arrays.*asList*("Один", "Два", "Три", "Четыре"));  
 }  
  
 public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException {  
 initialize(req, res);  
 String task = request.getParameter("task");  
 try {  
 if ("7".equals(task)) {  
 task7post();  
 }  
 } catch (Exception e) {  
 printException(e);  
 }  
 out.flush();  
 }  
  
 public void doGet(HttpServletRequest req, HttpServletResponse res) throws IOException {  
 initialize(req, res);  
 try {  
 int task = Integer.*parseInt*(request.getParameter("task"));  
 if(task < 1 || task > 12) {  
 throw new NumberFormatException();  
 }  
 if(task == 1) {  
 task1();  
 } else if(task == 2) {  
 task2();  
 } else if(task == 3) {  
 task3();  
 } else if(task == 4) {  
 task4();  
 } else if(task == 5) {  
 task5();  
 } else if(task == 6) {  
 task6();  
 } else if(task == 7) {  
 task7();  
 } else if(task == 8) {  
 task8();  
 } else if(task == 9) {  
 task9();  
 } else if(task == 10) {  
 task10();  
 } else if(task == 11) {  
 task11();  
 } else if(task == 12) {  
 task12();  
 }  
 } catch (NumberFormatException e) {  
 printHtml("Hello", "<h1>Привет Sirius!</h1>");  
 } catch (Exception e) {  
 printException(e);  
 }  
 out.flush();  
 }  
  
 public void initialize(HttpServletRequest req, HttpServletResponse res) throws IOException {  
 request = req;  
 response = res;  
 response.setCharacterEncoding("UTF-8");  
 request.setCharacterEncoding("UTF-8");  
 out = res.getWriter();  
 mains = ResourceBundle.*getBundle*("intern\_main", new Locale("ru", "RU"));  
 }  
  
 public void printException(Exception e) {  
 StackTraceElement[] stack = e.getStackTrace();  
 String[] stackTrace = new String[stack.length];  
 for(int i = 0; i < stack.length; ++i) {  
 stackTrace[i] = stack[i].toString();  
 }  
 printHtml("Exception!", String.*format*("<h1>%s: %s</h1><div>%s</div>",  
 e.getClass().getName(), e.getMessage(), String.*join*("<br>", stackTrace)));  
 }  
  
 public void task1() throws Exception {  
 Map<String, String> regions = getRegions("100410000050");  
 Obb filter = Ob0.*createFilter*(5);  
 String regionId = request.getParameter("region");  
 if(regionId == null || regionId.equals("")) {  
 regionId = "100518301512";  
 }  
 Ob0.*addCondition*(filter, 1005101368, Ob0.ComparisonType.*EQ*, regionId);  
 Obb[] cities = Ob0.*getSrcObs*(mains, filter, 0, 0);  
 List<String> regionIds = intern.Utils.*getKeysSortedByValue*(regions, false);  
 StringBuilder text = new StringBuilder("<form action=\"#\" method=\"get\">" +  
 "<table class=\"form\"><tr><td>Регион</td><td><select name=\"region\">");  
 for(String key : regionIds) {  
 text.append(String.*format*("<option%s value=\"%s\">%s</option>",  
 key.equals(regionId) ? " selected" : "", key, regions.get(key)));  
 }  
 text.append("</select></td></tr></table><input type=\"submit\" value=\"Выбрать\">" +  
 "<input type=\"hidden\" name=\"task\" value=\"1\"></form>");  
 Arrays.*sort*(cities, Comparator.*comparing*((Obb ob) -> ob.getAt("1000098")));  
 int i = 0;  
 text.append("<table class=\"data\"><tr><td>№</td><td>ID</td><td>Город</td></tr>");  
 for(Obb city : cities) {  
 text.append(String.*format*("<tr><td>%d</td><td>%s</td><td>%s</td></tr>",  
 ++i, city.id, Ob0.*getAt*(city, "1000098")));  
 }  
 text.append("</table>");  
 printHtml("Города России", text.toString());  
 }  
  
 public Map<String, String> getRegions(String countryId) throws Exception {  
 Obb filter = Ob0.*createFilter*(5);  
 Ob0.*addCondition*(filter, 1000004, Ob0.ComparisonType.*EQ*, countryId);  
 Ob0.*addCondition*(filter, 1005101368, Ob0.ComparisonType.*NEQ*, "");  
 Obb[] cities = Ob0.*getSrcObs*(mains, filter, 0, 0);  
 Map<String, String> regions = new TreeMap<>();  
 for(Obb city : cities) {  
 String regionId = Ob0.*getAt*(city, 1005101368);  
 if(!regions.containsKey(regionId)) {  
 regions.put(regionId, Ob0.*getZn*(mains, regionId, 1000098, 4));  
 }  
 }  
 return regions;  
 }  
  
 public void task2() throws Exception {  
 String countryId = request.getParameter("country");  
 if(countryId == null || countryId.equals("")) {  
 countryId = "100410000050";  
 }  
 Obb filter = Ob0.*createFilter*(5);  
 Ob0.*addCondition*(filter, 1000004, Ob0.ComparisonType.*EQ*, countryId);  
 Ob0.*addCondition*(filter, 1000101, Ob0.ComparisonType.*EQ*, "Да");  
 Obb[] cities = Ob0.*getSrcObs*(mains, filter, 0, 0);  
 Obb[] countries = Ob0.*getSrcObs*(mains, Ob0.*createFilter*(4), 0, 0);  
 Map<String, String> regions = getRegions(countryId);  
 Map<String, Set<String>> distribution = regionDistribute(cities);  
 List<String> regionIds = intern.Utils.*getKeysSortedByValue*(regions, false);  
 Arrays.*sort*(countries, Comparator.*comparing*((Obb ob) -> ob.getAt("1000000")));  
 StringBuilder text = new StringBuilder();  
 text.append("<form action=\"#\" method=\"get\"><table class=\"form\"><tr><td>" +  
 "Страна</td><td><select name=\"country\">");  
 for(Obb country : countries) {  
 text.append(String.*format*("<option%s value=\"%s\">%s</option>",  
 country.id.equals(countryId) ? " selected" : "",  
 country.id, country.getAt("1000000")));  
 }  
 text.append("</select></td></tr></table><input type=\"submit\" value=\"Выбрать\">" +  
 "<input type=\"hidden\" name=\"task\" value=\"2\"></form>");  
 if(regionIds.isEmpty()) {  
 text.append("<h4 style=\"text-align:center;color:grey;\">" +  
 "Данная страна не поддерживает разбиение на регионы</h4>");  
 } else {  
 text.append("<table class=\"data\"><tr><td>№</td><td>Область</td><td>Города</td></tr>");  
 int i = 0;  
 for (String key : regionIds) {  
 if (regions.containsKey(key) && distribution.containsKey(key)) {  
 text.append(String.*format*("<tr><td>%d</td><td>%s</td><td>%s</td></tr>",  
 ++i, regions.get(key),  
 String.*join*(", ", distribution.get(key))));  
 }  
 }  
 text.append("</table>");  
 }  
 printHtml("Города России", text.toString());  
 }  
  
 public Map<String, Set<String>> regionDistribute(Obb[] cities) {  
 Map<String, Set<String>> distribution = new TreeMap<>();  
 for(Obb city : cities) {  
 String regionId = Ob0.*getAt*(city, 1005101368);  
 if(!distribution.containsKey(regionId)) {  
 distribution.put(regionId, new TreeSet<>());  
 }  
 distribution.get(regionId).add(Ob0.*getAt*(city, 1000098));  
 }  
 return distribution;  
 }  
  
 public void task3() throws Exception {  
 response.setContentType("application/json; charset=UTF-8");  
 String birthday = request.getParameter("bd");  
 if(birthday == null || birthday.equals("")) {  
 birthday = "01.01.1990";  
 }  
 Obb filter = Ob0.*createFilter*(23);  
 Ob0.*addCondition*(filter, 1000152, Ob0.ComparisonType.*GT*, birthday);  
 Obb[] tourists = Ob0.*getSrcObs*(mains, filter, 0, 0);  
 Arrays.*sort*(tourists, Comparator.*comparing*(  
 (Obb ob) -> ob == null ? "" : ob.getAt("1000144")));  
  
 out.print("[");  
 int i = 0;  
 for(Obb tourist : tourists) {  
 out.printf("{\"id\":\"%s\",\"nm\":\"%s %s %s\",\"bd\":\"%s\",\"trs\":[",  
 tourist == null ? "null" : tourist.id,  
 Ob0.*getAt*(tourist, 1000144),  
 Ob0.*getAt*(tourist, 1000146),  
 Ob0.*getAt*(tourist, 1000147),  
 Ob0.*getAt*(tourist, 1000152));  
 String[] tours = Ob0.*getAt*(tourist, 1023422081).split("#");  
 int j = 0;  
 for(String tour : tours) {  
 out.printf("\"%s\"%s", tour, ++j < tours.length ? "," : "");  
 }  
 out.printf("]}%s", ++i < tourists.length ? "," : "");  
 }  
 out.print("]");  
 }  
  
 public void task4() throws Exception {  
 String agentName = request.getParameter("agent");  
 String categoryName = request.getParameter("category");  
 if(agentName == null || agentName.equals("")) {  
 agentName = "КонтрАг";  
 }  
 if(categoryName == null || categoryName.equals("")) {  
 categoryName = "Концерты";  
 }  
 Obb filter = Ob0.*createFilter*(36);  
 Ob0.*addCondition*(filter, new int[]{1036922797, 1317100000},  
 Ob0.ComparisonType.*EQ*, agentName);  
 Ob0.*addCondition*(filter, new int[]{1036900082, 1162100000},  
 Ob0.ComparisonType.*EQ*, categoryName);  
 Obb[] costs = Ob0.*getSrcObs*(mains, filter, 0, 0);  
 Obb[] agents = Ob0.*getSrcObs*(mains, Ob0.*createFilter*(317), 0, 0);  
 Obb[] categories = Ob0.*getSrcObs*(mains, Ob0.*createFilter*(162), 0, 0);  
 Arrays.*sort*(costs, (Obb left, Obb right) -> right.data\_n.compareTo(left.data\_n));  
 costs = Arrays.*copyOfRange*(costs, Math.*max*(0, costs.length - 50), costs.length);  
 Arrays.*sort*(costs, Comparator.*comparing*((Obb ob) -> ob.getAt("1036423021")));  
 Arrays.*sort*(agents, Comparator.*comparing*((Obb ob) -> ob.getAt("1317100000")));  
 StringBuilder text = new StringBuilder();  
 text.append("<form action=\"#\" method=\"get\"><table class=\"form\"><tr><td>" +  
 "Контрагент</td><td><select name=\"agent\">");  
 for(Obb agent : agents) {  
 String an = agent.getAt("1317100000");  
 text.append(String.*format*("<option%s>%s</option>",  
 an.equals(agentName) ? " selected" : "", an));  
 }  
 text.append("</select></td></tr><tr><td>Категория</td><td><select name=\"category\">");  
 for(Obb category : categories) {  
 String cn = category.getAt("1162100000");  
 text.append(String.*format*("<option%s>%s</option>",  
 cn.equals(categoryName) ? " selected" : "", cn));  
 }  
 text.append("</select></td><tr></table><input type=\"submit\" value=\"Выбрать\">" +  
 "<input type=\"hidden\" name=\"task\" value=\"4\"></form>" +  
 "<table class=\"data\"><tr><td>№</td><td>Название</td>" +  
 "<td>Категория</td><td>Адрес</td></tr>");  
 int i = 0;  
 for(Obb cost : costs) {  
 text.append(String.*format*("<tr><td>%d</td><td>%s</td><td>%s</td><td>%s</td></tr>",  
 ++i, cost.getAt("1036423021"),  
 *yesnoMap*.get(cost.getAt("1036200042")),  
 cost.getAt("1036410028")));  
 }  
 printHtml("Экскурсии", text.append("</table>").toString());  
 }  
  
 public void task5() throws Exception {  
 Obb[] costs = Ob0.*getSrcObs*(mains, Ob0.*createFilter*(36), 0, 0);  
 List<Obb> list = Arrays.*asList*(costs);  
 List<Obb> array = new ArrayList<>(list);  
 List<Obb> linked = new LinkedList<>(list);  
 StringBuilder text = new StringBuilder();  
 BiConsumer<List<Obb>, String> consumer = (List<Obb> lst, String listType) -> {  
 int i = 0;  
 text.append("<div>");  
 long time = System.*currentTimeMillis*();  
 for(Obb item : lst) {  
 text.append(String.*format*("[%d : %s]%s", ++i, item.id, i == lst.size() ? "" : ", "));  
 }  
 time = System.*currentTimeMillis*() - time;  
 text.append(String.*format*("<br>%s - %d ms</div>", listType, time));  
 };  
 consumer.accept(array, "ArrayList");  
 consumer.accept(linked, "LinkedList");  
 printHtml("Временное сравнение", text.toString());  
 }  
  
 public void task6() throws Exception {  
 String type = request.getParameter("type");  
 String code = request.getParameter("code");  
 String address = request.getParameter("address");  
 String category = request.getParameter("category");  
 String agentId = request.getParameter("agent");  
  
 type = NumberUtils.*isNumber*(type) && Integer.*parseInt*(type) >= 0  
 && Integer.*parseInt*(type) <= 8 && Integer.*parseInt*(type) != 7 ? type : "";  
 code = code == null ? "" : code;  
 address = address == null ? "" : address;  
 category = category == null || !(category.equals("1") || category.equals("0")) ? "" : category;  
 agentId = agentId == null ? "" : agentId;  
  
 Obb filter = Ob0.*createFilter*(36);  
 if(!type.equals("")) {  
 Ob0.*addCondition*(filter, 1036200042, Ob0.ComparisonType.*EQ*, type);  
 }  
 if(!code.equals("")) {  
 Ob0.*addCondition*(filter, 1036423021, Ob0.ComparisonType.*EQ*, code);  
 }  
 if(!address.equals("")) {  
 Ob0.*addCondition*(filter, 1036410028, Ob0.ComparisonType.*EQ*, address);  
 }  
 if(!category.equals("")) {  
 Ob0.*addCondition*(filter, 1162200125, Ob0.ComparisonType.*EQ*, category);  
 }  
 if(!agentId.equals("")) {  
 Ob0.*addCondition*(filter, 1036922797, Ob0.ComparisonType.*EQ*, agentId);  
 }  
 Obb[] costs = Ob0.*getSrcObs*(mains, filter, 0, 0);  
 Obb[] agents = Ob0.*getSrcObs*(mains, Ob0.*createFilter*(317), 0, 0);  
 Arrays.*sort*(agents, Comparator.*comparing*((Obb ob) -> ob.getAt("1317100000")));  
  
 Map<String, String> agentMap = new HashMap<>();  
 agentMap.put("", "-");  
 for(Obb agent : agents) {  
 agentMap.put(agent.id, agent.getAt("1317100000"));  
 }  
  
 StringBuilder text = new StringBuilder();  
 text.append("<form action=\"#\" method=\"get\"><table class=\"form\"><tr><td>" +  
 "Тип</td><td><select name=\"type\">");  
 for(String key : *typeMap*.keySet()) {  
 text.append(String.*format*("<option%s value=\"%s\">%s</option>",  
 type.equals(key) ? " selected" : "", key, *typeMap*.get(key)));  
 }  
 text.append("</select></td></tr><tr><td>Код</td>" +  
 "<td><input name=\"code\" type=\"text\" value=\"\"></td></tr>" +  
 "<tr><td>Адрес</td><td><input name=\"address\" type=\"text\" value=\"\">" +  
 "</td></tr><tr><td>Категория</td><td><select name=\"category\">");  
 for(String key : *yesnoMap*.keySet()) {  
 text.append(String.*format*("<option%s value=\"%s\">%s</option>",  
 category.equals(key) ? " selected" : "", key, *yesnoMap*.get(key)));  
 }  
 text.append("</select></td></tr><tr><td>Контрагент</td><td><select name=\"agent\">");  
 for(String key : agentMap.keySet()) {  
 text.append(String.*format*("<option%s value=\"%s\">%s</option>",  
 agentId.equals(key) ? " selected" : "", key, agentMap.get(key)));  
 }  
 text.append("</select></td></tr></table><input type=\"submit\" value=\"Выбрать\">" +  
 "<input type=\"hidden\" name=\"task\" value=\"6\"></form>" +  
 "<table class=\"data\"><tr><td>№</td><td>Название</td><td>Тип</td>" +  
 "<td>Адрес</td><td>Да/Нет</td><td>Контрагент</td></tr>");  
 Arrays.*sort*(costs, Comparator.*comparing*((Obb ob) -> ob.getAt("1036423021")));  
 int i = 0;  
 for(Obb cost : costs) {  
 text.append(String.*format*("<tr><td>%d</td><td>%s</td><td>%s</td>" +  
 "<td>%s</td><td>%s</td><td>%s</td></tr>",  
 ++i, cost.getAt("1036423021"),  
 *typeMap*.get(cost.getAt("1036200042")),  
 cost.getAt("1036410028"),  
 *yesnoMap*.get(cost.getAt("1162200125")),  
 agentMap.get(cost.getAt("1036922797"))));  
 }  
 printHtml("Экскурсии", text.append("</table>").toString());  
 }  
  
 public void task7() throws Exception {  
 Map<String, String> regions = new HashMap<>();  
 Map<String, String> partners = new HashMap<>();  
 regionsAndPartners(regions, partners);  
  
 StringBuilder text = new StringBuilder();  
 text.append("<form method=\"post\" action=\"#\"><table class=\"form\">" +  
 "<tr><td>Название</td><td><input type=\"text\" name=\"name\"></td></tr>" +  
 "<tr><td>Описание</td><td><textarea name=\"desc\"></textarea></td></tr>" +  
 "<tr><td>Регион</td><td><select name=\"region\">");  
 List<String> regionIds = intern.Utils.*getKeysSortedByValue*(regions, false);  
 for(String regionId : regionIds) {  
 text.append(String.*format*("<option value=\"%s\">%s</option>",  
 regionId, regions.get(regionId)));  
 }  
 text.append("</select></td></tr>" +  
 "<tr><td>Доп. оплата</td><td><input type=\"text\" name=\"cost\"></td></tr>" +  
 "<tr><td>Бронирование у партнера</td><td><select name=\"partner\">");  
 for(String partnerId : partners.keySet()) {  
 text.append(String.*format*("<option value=\"%s\">%s</option>",  
 partnerId, partners.get(partnerId)));  
 }  
 text.append("</select></td></tr>" +  
 "<tr><td>Тип</td><td><select name=\"type\">");  
 for(String typeId : *typeMap*.keySet()) {  
 text.append(String.*format*("<option value=\"%s\">%s</option>",  
 typeId, *typeMap*.get(typeId)));  
 }  
 text.append("</select></td></tr>" +  
 "</table><input type=\"hidden\" name=\"task\" value=\"7\">" +  
 "<input type=\"submit\" value=\"Создать\"></form>");  
 printDescs(text, regions, partners);  
 }  
  
 public void task7post() throws Exception {  
 String name = request.getParameter("name");  
 String desc = request.getParameter("desc");  
 String region = request.getParameter("region");  
 String cost = request.getParameter("cost");  
 String partner = request.getParameter("partner");  
 String type = request.getParameter("type");  
  
 name = name == null ? "" : name;  
 desc = desc == null ? "" : desc;  
 region = region == null ? "-" : region;  
 cost = !NumberUtils.*isNumber*(cost) ? "" : cost;  
 partner = partner == null ? "-" : partner;  
 type = !*typeMap*.containsKey(type) ? "-" : type;  
  
 if(!name.equals("") && !desc.equals("") && !region.equals("-") &&  
 !cost.equals("") && !partner.equals("-") && !type.equals("-")) {  
 Obb ob = new Obb(506);  
 ob.id\_user = myId;  
 Ob0.*addAt*(ob, "1506410000", name);  
 Ob0.*addAt*(ob, "1506410282", desc);  
 Ob0.*addAt*(ob, "1506923461", region);  
 Ob0.*addAt*(ob, "1506223120", cost);  
 Ob0.*addAt*(ob, "1506910189", partner);  
 Ob0.*addAt*(ob, "1506310181", type);  
 Ob0.*addOb*(mains, ob);  
 }  
 response.sendRedirect("/eldar?task=7");  
 }  
  
 public void task8() throws Exception {  
 String descId = request.getParameter("did");  
 descId = descId == null ? "" : descId;  
 if(!descId.equals("")) {  
 Ob0.*delOb*(mains, descId, myId);  
 }  
 Map<String, String> regions = new HashMap<>();  
 Map<String, String> partners = new HashMap<>();  
 regionsAndPartners(regions, partners);  
 StringBuilder text = new StringBuilder();  
 text.append("<form method=\"get\" action=\"#\">" +  
 "ID <input type=\"text\" name=\"did\">" +  
 "<input type=\"submit\" value=\"Удалить\">" +  
 "<input type=\"hidden\" name=\"task\" value=\"8\"></form>");  
 printDescs(text, regions, partners);  
 }  
  
 public void regionsAndPartners(Map<String, String> regions,  
 Map<String, String> partners) throws Exception {  
 Obb filter = Ob0.*createFilter*(5);  
 Ob0.*addCondition*(filter, 1000004, Ob0.ComparisonType.*EQ*, "100410000050");  
 Ob0.*addCondition*(filter, 1005101368, Ob0.ComparisonType.*NEQ*, "");  
 Obb[] cities = Ob0.*getSrcObs*(mains, filter, 0, 0);  
 Obb[] prtns = Ob0.*getSrcObs*(mains, Ob0.*createFilter*(158), 0, 0);  
 regions.put("", "-");  
 partners.put("", "-");  
 for(Obb city : cities) {  
 regions.put(city.id, city.getAt("1000098"));  
 }  
 for(Obb partner : prtns) {  
 partners.put(partner.id, partner.getAt("1001211"));  
 }  
 }  
  
 public void printDescs(StringBuilder text, Map<String, String> regions,  
 Map<String, String> partners) throws Exception {  
 Obb[] descs = Ob0.*getSrcObs*(mains, Ob0.*createFilter*(506), 0, 0);  
 text.append("<table class=\"data\"><tr><td>№</td><td>ID</td><td>Название</td>" +  
 "<td>Описание</td><td>Регион</td>" +  
 "<td>Доп. оплаты</td><td>Бронирование у парнера</td><td>Тип</td></tr>");  
 int i = 0;  
 for(Obb desc : descs) {  
 text.append(String.*format*("<tr><td>%d</td><td>%s</td><td>%s</td><td>%s</td>" +  
 "<td>%s</td><td>%s</td><td>%s</td><td>%s</td></tr>",  
 ++i, (desc.id\_user == myId ? "<b style=\"color:green;\">" + desc.id + "<b>" : desc.id),  
 desc.getAt("1506410000"),  
 desc.getAt("1506410282"),  
 regions.get(desc.getAt("1506923461")),  
 desc.getAt("1506223120"),  
 partners.get(desc.getAt("1506910189")),  
 *typeMap*.get(desc.getAt("1506310181"))));  
 }  
 text.append("</table>");  
 printHtml("Описания экскурсий", text.toString());  
 }  
  
 public void task9() throws Exception {  
 long time = 100000000L;  
 int expire = 180;  
 String[] types = new String[]{"С", "БНС", "НС"};  
 String costId = Util.*s2s*(request.getParameter("id"));  
 String foundOrCreated = null;  
 Obb ob = null;  
 if(!costId.equals("")) {  
 ob = Ob0.*fromBytes*(Ob3.*get*(mains, redis, redis, (prefix + costId).getBytes()));  
 if(ob == null) {  
 ob = Ob0.*getOb*(mains, costId);  
 if(ob == null) {  
 foundOrCreated = "<div style=\"text-align:center;color:red;\">" +  
 "Искомый объект не существует в базе PostgreSQL</div>";  
 } else {  
 Ob3.*puts*(mains, redis, (prefix + costId).getBytes(), Ob0.*toBytes*(ob),  
 time, false, expire, "");  
 foundOrCreated = "<div style=\"text-align:center;color:blue;\">" +  
 "Объект изъят из базы PostgreSQL и записан в REDIS</div>";  
 }  
 } else {  
 foundOrCreated = "<div style=\"text-align:center;color:green;\">Объект найден</div>";  
 }  
 }  
 String text = String.*format*("<form method=\"get\" action=\"#\">" +  
 "ID <input type=\"text\" name=\"id\" value=\"%s\"> " +  
 "<input type=\"submit\" value=\"Найти\">" +  
 "<input type=\"hidden\" name=\"task\" value=\"9\">" +  
 "</form>%s%s", costId, foundOrCreated == null ? "" : foundOrCreated, ob == null ? "" :  
 String.*format*("<table class=\"data\">" +  
 "<tr><td>Название</td><td>ID номера</td><td>Тип стоимости</td></tr>" +  
 "<tr><td>%s</td><td>%s</td><td>%s</td></tr>" +  
 "</table>",  
 ob.getAt(1000348),  
 ob.getAt(1000350),  
 types[Integer.*parseInt*(ob.getAt(1046222729))]));  
 printHtml("Redis - Один", text);  
 }  
  
 public void task10() throws Exception {  
 long time = 100000000L;  
 int expire = 1;  
 String action = request.getParameter("action");  
 action = action == null || action.equals("") ? "find" : action;  
 String key = Util.*s2s*(request.getParameter("key"));  
 String value = null;  
 if(action.equals("find") && !key.equals("")) {  
 byte[] bytes = Ob3.*get*(mains, redis, redis, (prefix + key).getBytes());  
 value = bytes == null ? "" : new String(bytes, StandardCharsets.*UTF\_8*);  
 } else if (action.equals("create")) {  
 value = request.getParameter("value");  
 Ob3.*puts*(mains, redis, (prefix + key).getBytes(), value.getBytes());  
 } else if (action.equals("delete")) {  
 *//Ob3.del(mains, redis, prefix + key, redis);*  
byte[] bytes = Ob3.*get*(mains, redis, redis, (prefix + key).getBytes());  
 value = bytes == null ? "" : new String(bytes, StandardCharsets.*UTF\_8*);  
 Ob3.*puts*(mains, redis, (prefix + key).getBytes(), "".getBytes(),  
 time, false, expire, "");  
 }  
 String text = String.*format*("<form method=\"get\" action=\"#\"><table class=\"form\">" +  
 "<tr><td>Ключ</td><td><input type=\"text\" name=\"key\"></td></tr>" +  
 "<tr><td>Значение</td><td><input type=\"text\" name=\"value\"></td></tr>" +  
 "</table>" +  
 "<input type=\"submit\" value=\"Создать\">" +  
 "<input type=\"hidden\" name=\"task\" value=\"10\">" +  
 "<input type=\"hidden\" name=\"action\" value=\"create\">" +  
 "</form>%s" +  
 "<form method=\"get\" action=\"#\">" +  
 "Ключ <input type=\"text\" name=\"key\"> " +  
 "<input type=\"submit\" value=\"Найти\">" +  
 "<input type=\"hidden\" name=\"task\" value=\"10\">" +  
 "<input type=\"hidden\" name=\"action\" value=\"find\">" +  
 "</form>%s" +  
 "<form method=\"get\" action=\"#\">" +  
 "Ключ <input type=\"text\" name=\"key\"> " +  
 "<input type=\"submit\" value=\"Удалить\">" +  
 "<input type=\"hidden\" name=\"task\" value=\"10\">" +  
 "<input type=\"hidden\" name=\"action\" value=\"delete\">" +  
 "</form>%s",  
 action.equals("create") ? "<p style=\"text-align:center;color:green\">[" +  
 key + " : " + value + "] создано</p>" : "<br>",  
 !key.equals("") && action.equals("find") ?  
 !value.equals("")  
 ? "<p style=\"text-align:center;color:green\">["  
 + key + " : " + value + "]</p>"  
 : "<p style=\"text-align:center;color:red\">Объект с ключом "  
 + key + " не найден.</p>"  
 : "<br>",  
 action.equals("delete")  
 ? !value.equals("")  
 ? "<p style=\"text-align:center;color:green\">["  
 + key + "] удален</p>"  
 : "<p style=\"text-align:center;color:red\">Объект с ключом "  
 + key + " не найден.</p>"  
 : "<br>");  
 printHtml("Redis - Два", text);  
 }  
  
 public void task11() throws Exception {  
 Obb[] obs = Ob0.*getSrcObs*(mains, Ob0.*createFilter*(36), 0, 0);  
 List<String> result = new ArrayList<>(obs.length);  
 String action = Util.*s2s*(request.getParameter("action"));  
 action = action.equals("") ? "postgre" : action;  
 long time = -1;  
 if(action.equals("postgre")) {  
 time = catchTime(() -> {  
 for(Obb ob : obs) {  
 result.add(Ob0.*getOb*(mains, ob.id).id);  
 }  
 });  
 } else if(action.equals("redis")) {  
 time = catchTime(() -> {  
 for(Obb ob : obs) {  
 Obb temp = Ob0.*fromBytes*(Ob3.*get*(mains, redis, redis, (prefix + ob.id).getBytes()));  
 if(temp != null) {  
 result.add(temp.id);  
 } else {  
 result.clear();  
 break;  
 }  
 }  
 });  
 } else if(action.equals("in\_redis")) {  
 long time0 = 1000000000L;  
 int expire = 3600;  
 for(Obb ob : obs) {  
 Ob3.*puts*(mains, redis, (prefix + ob.id).getBytes(), Ob0.*toBytes*(ob),  
 time0, false, expire, "");  
 }  
 } else {  
 throw new Exception("Плохой аргумент");  
 }  
 StringBuilder text = new StringBuilder();  
 text.append("<div style=\"text-align:center;\"><p>" +  
 "<a href=\"?task=11&action=postgre\">Выбрать из PostgreSQL</a> | " +  
 "<a href=\"?task=11&action=redis\">Выбрать из Redis</a> | " +  
 "<a href=\"?task=11&action=in\_redis\">Заполнить Redis</a></p>");  
 text.append(time == -1  
 ? String.*format*("<p style=\"color:green\">%d объектов добавлено в Redis.</p>",  
 obs.length)  
 : !result.isEmpty()  
 ? String.*format*("<p>Время выборки из %s: %d мс</p>",  
 action.equals("postgre") ? "PostgreSQL" : "Redis", time)  
 : "<h1 style=\"text-align:center;\">" +  
 "<a href=\"?task=11&action=in\_redis\">Заполните</a> Redis</h1>");  
 text.append("</div>");  
 if(!result.isEmpty()) {  
 text.append(String.*join*(", ", result));  
 }  
 printHtml("Redis - Три", text.toString());  
 }  
  
 public long catchTime(Procedure procedure) throws Exception {  
 long time = System.*currentTimeMillis*();  
 procedure.run();  
 return System.*currentTimeMillis*() - time;  
 }  
  
 public interface Procedure {  
 void run() throws Exception;  
 }  
  
 public void task12() throws Exception {  
 printHtml("Redis - Четыре",  
 "<div style=\"border:1px solid #ccc; width:900px; margin:0px auto; padding:15px;\">" +  
 "Целесообразно применять Redis в онлайн-магазинах для корзины, " +  
 "в онлайн-играх по типу шахмат для хранения ходов и состояния шахматной доски, " +  
 "в стриминговых платформах для буферизации видеопотоков..." +  
 "</div>");  
 }  
  
 public void printHtml(String title, String text) {  
 response.setContentType("text/html; charset=UTF-8");  
 String task = request.getParameter("task");  
 StringBuilder output = new StringBuilder();  
 output.append(String.*format*("<!doctype html><html><head><title>%s</title><style>" +  
 "table { margin: 0px auto; }" +  
 "table.form tr td:first-child { text-align:right; }" +  
 "table.form tr td:last-child { text-align:left; }" +  
 "table.data { border: solid 1px #ccc; border-spacing: 3px;" +  
 "border-collapse: collapse; margin-bottom:10px; }" +  
 "table.data tr:first-child { font-weight:bold; }" +  
 "table.data td { border: solid 1px #ccc; padding: 5px; }" +  
 "form { margin: 0px auto; margin-bottom:20px; text-align:center;" +  
 "padding:10px; }" +  
 "</style></head>" +  
 "<body><div style=\"text-align:center;margin-bottom:20px;\">" +  
 "<table class=\"form\">", title));  
 int tasks = 0;  
 for(String key : *pagesMap*.keySet()) {  
 output.append(String.*format*("<tr>" +  
 "<td style=\"font-weight:bold;color:#072d78;padding-right:15px;\">" +  
 "%s</td><td>", key));  
 List<String> numbers = *pagesMap*.get(key);  
 for(int i = 1, n = numbers.size(); i <= n; ++i) {  
 if(String.*valueOf*(tasks + i).equals(task)) {  
 output.append(String.*format*("<b style=\"color:grey\">%s</b>%s",  
 numbers.get(i - 1), i < n ? " | " : ""));  
 } else {  
 output.append(String.*format*("<a href=\"?task=%d\">%s</a>%s",  
 tasks + i, numbers.get(i - 1), i < n ? " | " : ""));  
 }  
 }  
 output.append("</td></tr>");  
 tasks += numbers.size();  
 }  
 output.append(String.*format*("</table></div>%s</body></html>", text));  
 out.print(output);  
 }  
}