Звіт

Автор: Водолазський Микола Анатолійович

КІТ-118а

Лабораторна робота №15

# Колекції в java

***Мета:***

* Розширення функціональності параметризованих класів.

## **Вимоги**

1. Розробити консольну програму для реалізації завдання обробки даних згідно [прикладної області](https://oop-khpi.gitlab.io/%D0%B7%D0%B0%D0%B2%D0%B4%D0%B0%D0%BD%D0%BD%D1%8F/task07/" \l "_4).
2. Для розміщення та обробки даних використовувати контейнери (колекції) і алгоритми з [Java Collections Framework](https://docs.oracle.com/javase/8/docs/technotes/guides/collections/).
3. Забезпечити обробку колекції об'єктів: додавання, видалення, пошук, сортування згідно розділу [Прикладні задачі л.р. №10](https://oop-khpi.gitlab.io/%D0%B7%D0%B0%D0%B2%D0%B4%D0%B0%D0%BD%D0%BD%D1%8F/task10/" \l "_3).
4. Передбачити можливість довготривалого зберігання даних: 1) за допомогою стандартної серіалізації; 2) не використовуючи протокол серіалізації.
5. Продемонструвати розроблену функціональність в діалоговому та автоматичному режимах за результатом обробки параметрів командного рядка.

ПРИКЛАДНА ЗАДАЧА:

Кадрове агентство. Сортування за назвою фірми, за назвою запропонованої

спеціальності, за вказаною освітою.

ОПИС ПРОГРАМИ

***2.1 Опис змінних:***

LinkedContainer<SecondCreate> stringLinked = new LinkedContainer<>();// об’єкт параметризованого контейнера

Scanner scan = new Scanner(System.in); // змінна для активування зчитування з консолі

***2.2 Ієрархія та структура класів.***

Main class – головний клас. Містить метод main(точку входу у програму) та методи по роботі з програмою для реалізації індивідуального завдання.

interface iLinked - інтерфейс контенеру

class SecondCreate - клас прикладної задачі кадрового агенства

class linkedContainer - параметризований клас-контейнер, котрий зберігає інформацію агенства

**ТЕКСТ ПРОГРАМИ**

Main.java

import javax.xml.parsers.ParserConfigurationException;  
import javax.xml.transform.TransformerException;  
import java.io.IOException;  
  
public class Main {  
  
 public static void main(String args[]) throws ClassNotFoundException, ParserConfigurationException, TransformerException, IOException {  
  
String choose = args[0];  
  
 if(choose.equals("-auto"))  
 {  
 //automode();  
 Auto a = new Auto();  
 a.auto();  
 }else  
 {  
 Manual m = new Manual();  
 m.switchCase();  
 //manual();  
 }  
 }  
}

Manual.java

import javax.xml.parsers.ParserConfigurationException;  
import javax.xml.transform.TransformerException;  
import java.io.\*;  
import java.util.ArrayList;  
import java.util.Scanner;  
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Manual {  
  
 public void switchCase() throws IOException, ClassNotFoundException, TransformerException, ParserConfigurationException {  
 System.*out*.println("U have chosen manual mode");  
 int choose;  
  
 ArrayList<SecondCreate> arrayList = new ArrayList<SecondCreate>();  
 SecondCreate SecondCreate1 = null;  
  
 do{  
 System.*out*.println("Choose action ");  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.println("1. Create new element");  
 System.*out*.println("2. Add elem ");  
 System.*out*.println("3. Clear container ");  
 System.*out*.println("4. Convert to Array ");  
 System.*out*.println("5. Serialize ");  
 System.*out*.println("6. Deserialize ");  
 System.*out*.println("7. Xml serialize");  
 System.*out*.println("8. Xml deserialize");  
 System.*out*.println("9. Search element by criteria");  
 System.*out*.println("10. Delete element by index");  
 choose = in.nextInt();  
 switch (choose) {  
 case 1:  
 Scanner din = new Scanner(System.*in*);  
 Scanner cin = new Scanner(System.*in*);  
 System.*out*.println("Enter company name");  
 String company = din.nextLine();  
 if(company.matches("[a-zA-Z0-9]\*")==true)  
 {  
 System.*out*.println("Name is ok");  
 }else{System.*out*.println("NOT ok RENAME");company = din.nextLine();}  
 System.*out*.println("Enter specialisation");  
 String specialisation=din.nextLine();  
 if(specialisation.matches("[0-9]\*")==true)  
 {  
 System.*out*.println("ok");  
 }else{ System.*out*.println("NOT ok RENAME"); specialisation = din.nextLine();}  
 System.*out*.println("Enter working Conditions");  
 String workingConditions=din.nextLine();  
 System.*out*.println("Enter payment");  
 int payment=cin.nextInt();  
 System.*out*.println("Enter working Experience");  
 int workingExperience=cin.nextInt();  
 System.*out*.println("Enter education");  
 String education = din.nextLine();  
 System.*out*.println("Enter knowledge of English");  
 String English = din.nextLine();  
 System.*out*.println("Enter driving licence");  
 String License = din.nextLine();  
 SecondCreate1 = new SecondCreate(company,specialisation,workingConditions,payment,workingExperience,education,License,English);  
 break;  
  
 case 2:  
 System.*out*.println(arrayList.size());  
 arrayList.add(SecondCreate1);  
 System.*out*.println(arrayList.size());  
  
 for(SecondCreate tmp : arrayList)  
 {  
 System.*out*.println(tmp);  
 }  
  
 break;  
 case 3:  
 arrayList.clear();  
 System.*out*.println(arrayList.size());  
 break;  
 case 4:  
 Object []arr = arrayList.toArray();  
 for(int i=0; i<arrayList.size();i++)  
 {  
 System.*out*.println(arr[i]);  
 }  
 break;  
 case 5:  
 ObjectOutputStream objectOutputStream = new ObjectOutputStream(new FileOutputStream("store.txt"));  
 objectOutputStream.writeObject(arrayList);  
 objectOutputStream.close();  
  
 break;  
 case 6:  
 ObjectInputStream objectInputStream = new ObjectInputStream(new FileInputStream("store.txt"));  
 ArrayList<SecondCreate> newContainer = (ArrayList<SecondCreate>)objectInputStream.readObject();  
  
 for (SecondCreate t : newContainer) {  
 System.*out*.println(t);  
 }  
  
 break;  
 case 7:  
  
 XmlWrite xmlWrite = new XmlWrite();  
 xmlWrite.*write*(arrayList,"XML.xml");  
 break;  
 case 8:  
 ArrayList<SecondCreate> newXml = XmlRead.*read*("XML.xml");  
  
 for(SecondCreate t : newXml )  
 {  
 System.*out*.println(t);  
 }  
 break;  
 case 9:  
 for (SecondCreate t : arrayList)  
 {  
 Pattern p1 = Pattern.*compile*("teacher", Pattern.*CASE\_INSENSITIVE*);  
 Matcher m1 = p1.matcher(t.getSpecialisation());  
 if (m1.find()) {  
 if (t.getWorkingExperience() >= 10) {  
 Pattern p2 = Pattern.*compile*("yes", Pattern.*CASE\_INSENSITIVE*);  
 Matcher m2 = p2.matcher(t.getEnglish());  
 if (m2.find()) {  
 Pattern p3 = Pattern.*compile*("yes", Pattern.*CASE\_INSENSITIVE*);  
 Matcher m3 = p3.matcher(t.getLicence());  
 if (m3.find()) {  
 System.*out*.println(t);  
 }  
 }  
 }  
 }  
 }  
  
 break;  
 case 10:  
 Scanner sin = new Scanner(System.*in*);  
 System.*out*.println("Enter index of element u want to delete");  
 int index = sin.nextInt();  
 arrayList.remove(0);  
 break;  
 default:  
 break;  
 }}while(choose!=11);  
 }  
  
  
}

Auto.java

import java.io.File;  
import java.io.FileNotFoundException;  
import java.util.ArrayList;  
import java.util.Arrays;  
import java.util.Scanner;  
import java.util.concurrent.\*;  
  
public class Auto {  
  
 public void auto() throws FileNotFoundException {  
  
 File file = new File("file.txt");  
  
 Scanner scanner = new Scanner(file);  
  
 String line = scanner.nextLine();  
 String[] array = line.split(" ");  
  
 String company = null;  
 String specialisation = null;  
 String workingConditions = null;  
 int payment = 0;  
 int workingExperience = 0;  
 String education = null;  
 String English = null;  
 String Licence = null;  
  
  
 for (int i = 0; i < array.length; i++) {  
 company = array[0].toString();  
 specialisation = array[1].toString();  
 workingConditions = array[2].toString();  
 payment = Integer.*parseInt*(array[3]);  
 workingExperience = Integer.*parseInt*(array[4]);  
 education = array[5].toString();  
 Licence = array[6].toString();  
 English = array[7].toString();  
 }  
  
  
 SecondCreate firstWorker = new SecondCreate(company, specialisation, workingConditions, payment, workingExperience, education,Licence,English);  
 SecondCreate secondCreate = new SecondCreate("epam", "teacher", "good", 100, 1, "none","no","no");  
 SecondCreate thirdWorker = new SecondCreate("globalLogic","teacher","10.00-19.00",300,11,"magistry","yes","yes");  
  
 SecondCreate[] arr = {firstWorker, secondCreate,thirdWorker};  
  
  
 System.*out*.println("SORT BY COMPANY NAME");  
 Arrays.*sort*(arr);  
  
 for (SecondCreate tmp : arr) {  
 System.*out*.println(tmp);  
 }  
 comparatorC comparatorC = new comparatorC();  
  
  
 System.*out*.println("SORT BY Specialisation");  
 Arrays.*sort*(arr,comparatorC);  
  
 for(SecondCreate tmpss : arr)  
 {  
 System.*out*.println(tmpss);  
 }  
  
  
 System.*out*.println("SORT BY EDUCATION");  
  
 secondComparator secondComparator = new secondComparator();  
 Arrays.*sort*(arr,secondComparator);  
  
 for(SecondCreate tmpp : arr)  
 {  
 System.*out*.println(tmpp);  
 }  
  
 ArrayList arrayList = new ArrayList();  
 arrayList.add(firstWorker);  
 // linkedContainer.addLast(secondCreate);  
 // linkedContainer.addLast(thirdWorker);  
 System.*out*.println("Container size");  
 System.*out*.println(arrayList.size());  
  
 }  
  
}

**ВАРІАНТИ ВИКОРИСТАННЯ**

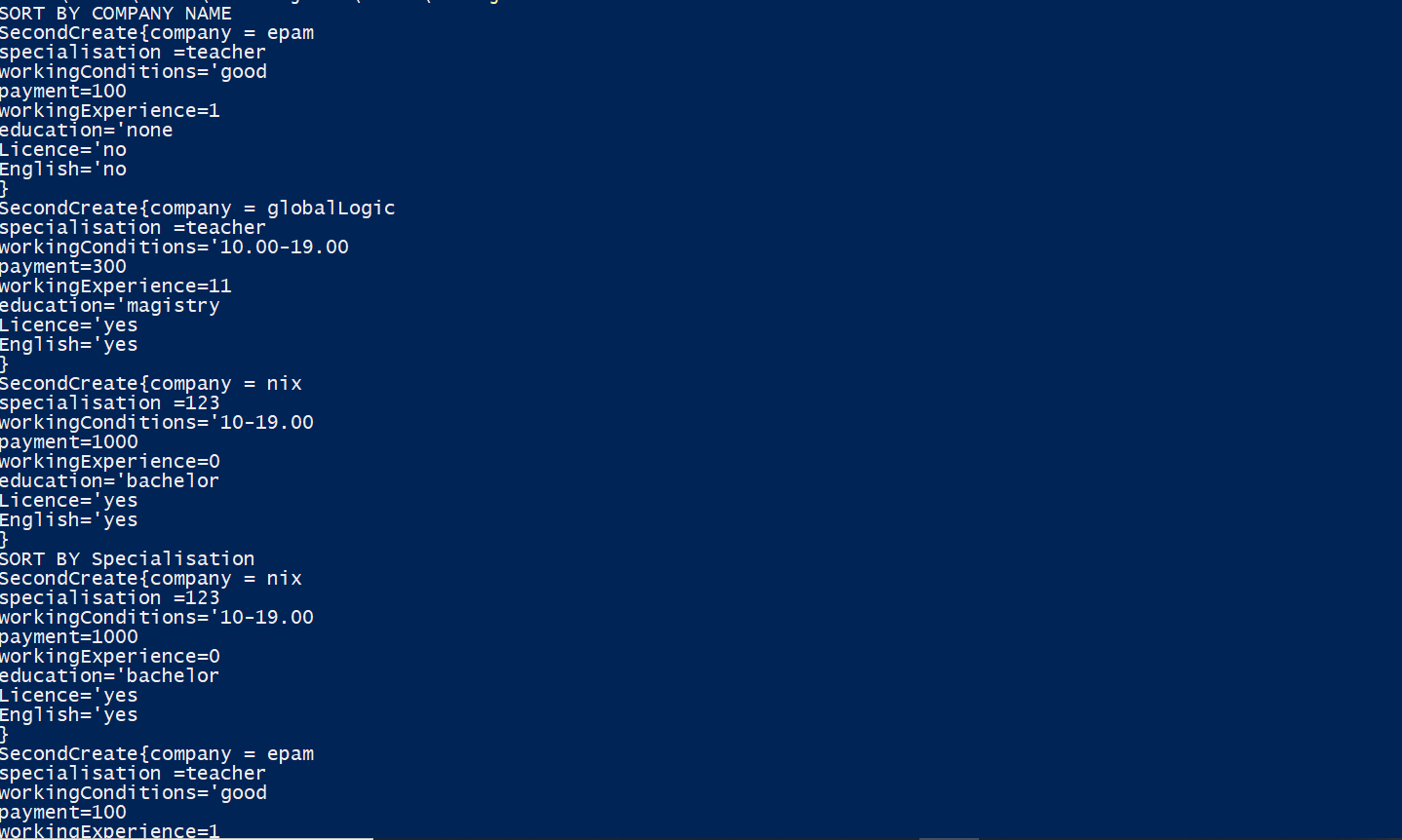


Рис. 15.1 – Результат роботи програми в автоматичному режимі.

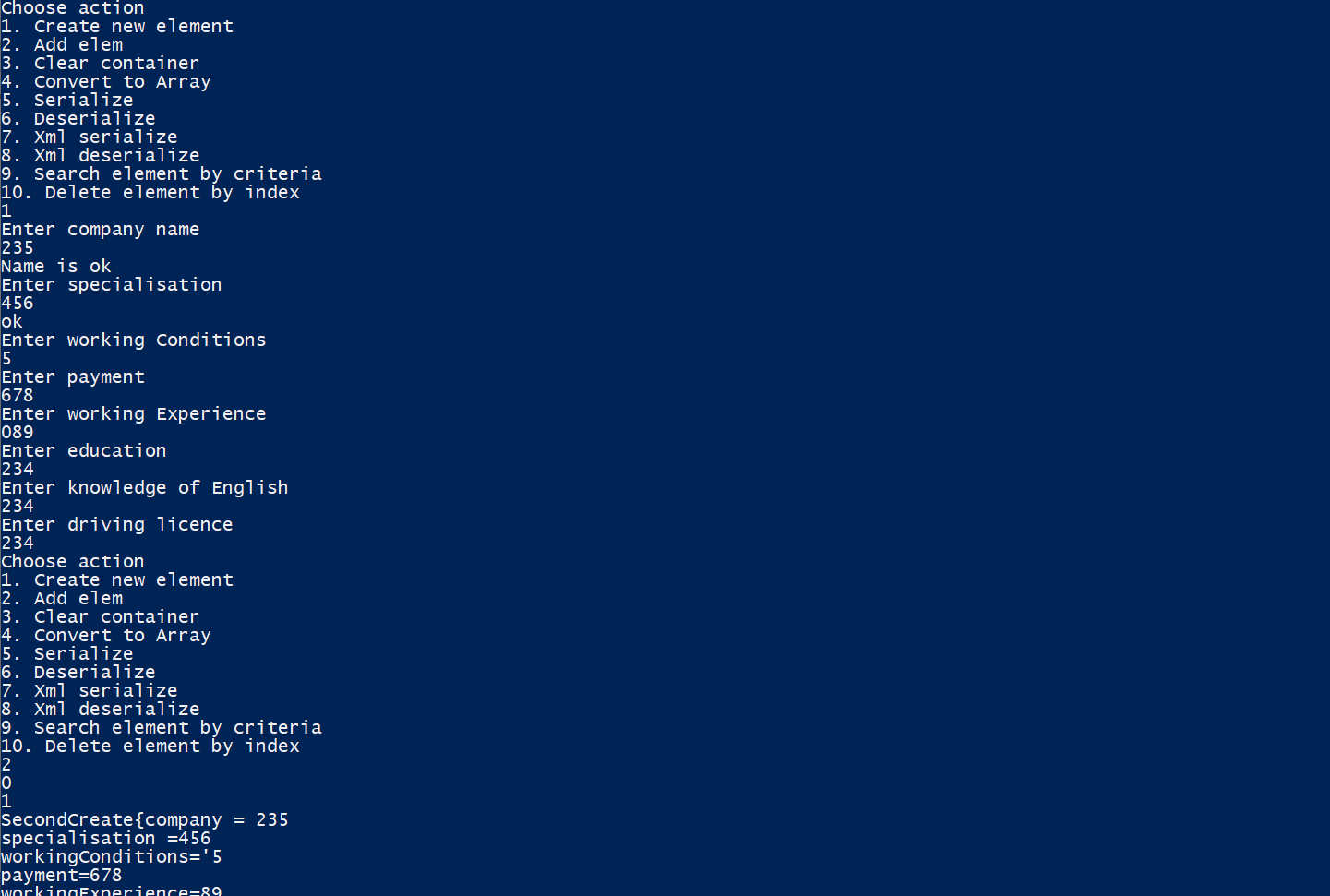


Рис. 15.2 – Результат роботи програми в ручному режимі.

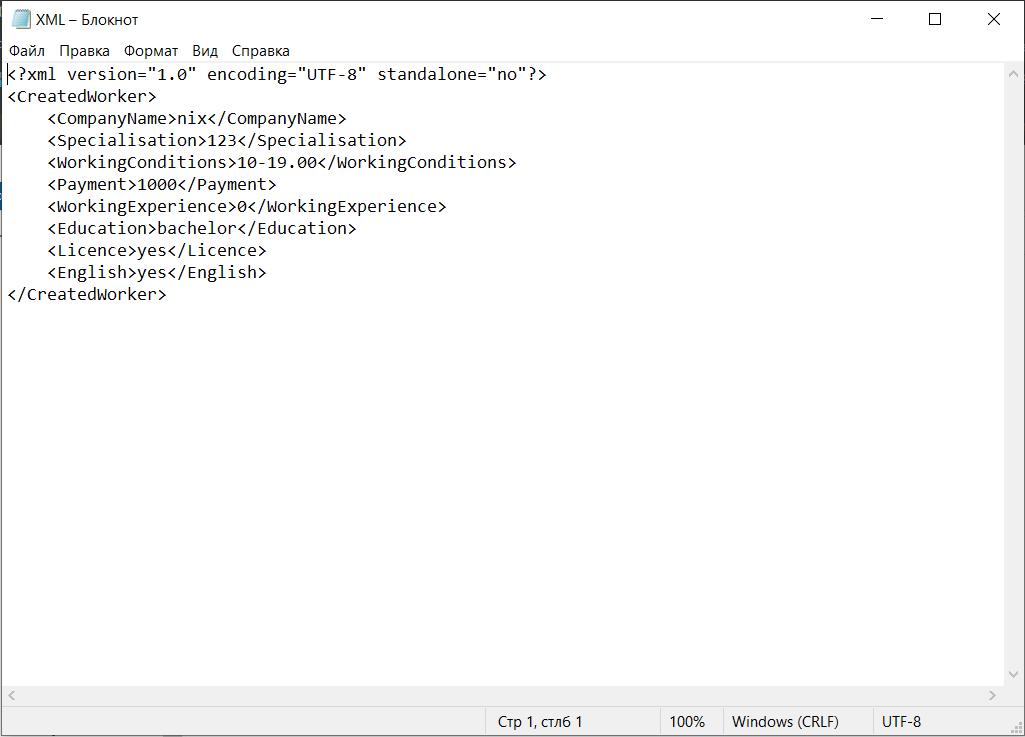


Рис. 15.3 – Результат роботи програми

Програму можна використовувати задля створення бази даних. Завдяки параметризації зв’язного списка, базу даних можна використати для будь-яких типів даних. Переважно у нашому варіанті - кадрове агенство, в якому представляються різноманітні вакансії. Також для вибору доступно багато інших можливостей. Реалізовано меню для поліпшення користування програмою.

**ВИСНОВКИ**

При виконанні лабораторної роботи набуто практичних навичок щодо розробки параметризованих класів. Завдяки цієї можливості в JAVA, можливо створювати колекції та інші класи на основі будь-яких типів. Також навчився обробляти параметризовані контейнери. Завдання виконане! Програма працює успішно!