Тема 10. Обробка параметризованих контейнерів∗

Мета:

- Розширення функціональності параметризованих класів.

1 ВИМОГИ

1.1 Розробник

Інформація про розробника:
- Гряник Георгій Володимиович
- КІТ-119Д;
- 6 варіант.

1.2 Загальне завдання

Використовуючи програму рішення завдання лабораторної роботи №9:

1. Розробити параметризовані методи (Generic Methods) для обробки колекцій об'єктів згідно прикладної задачі.
2. Продемонструвати розроблену функціональність (створення, управління та обробку власних контейнерів) в діалоговому та автоматичному режимах.
   - Автоматичний режим виконання програми задається параметром командного рядка **-auto**. Наприклад, java ClassName -auto.
   - В автоматичному режимі діалог з користувачем відсутній, необхідні данні генеруються, або зчитуються з файлу.
3. Забороняється використання алгоритмів з Java Collections Framework.

1.3 Задача

Поліцейська картотека

Поліцейська картотека. Сортування за прізвищем, за датою народження, за датою останнього позбавлення волі, за датою останнього звільнення.

2 ОПИС ПРОГРАМИ

2.1 Засоби ООП

Розробити параметризовані методи (Generic Methods) для обробки колекцій , автоматизація процесу за бажанням користувача

2.2 Ієрархія та структура класів

Клас “PoliceFile ” – описує поліцейську картотеку з можливістю додавати та виводити дані класу. Клас описує дані про злочинця відповідно до завданн. Клас «Date” – опису формат часу :день, місяць, рік . Створений для ергономічного запису дат відомостей про злочинця. Клас « Console_program» - клас керування програми , створений щоб надавати користувачеві можливість керувати програмою. Клас «ContainerList» - клас-контейнер створений для зберігання даних у список. Реалізовано додавання, видалення та інші можливості для керування даними. Клас «Serializator» - клас розроблений для збереження даних контейнеру у файл. При цьому зберігання прохоже у звичайний файл та файл типу .xml. При цьому в класі реалізовано методи для відновлення даних як із звичайного файлу

так із .xml файлу. Клас Console_File розроблений для роботи із файлами розміщені в директоріях. Цей клас забезпечує можливість користувачеві обирати файл та перемінятися між директоріями. Клас «Helper» - реалізація допоміжних методів які реалізують допоміжні дії в основній програмі. Клас «Console_program_auto» - клас розроблений для автоматизації програмию.

2.3 Важливі фрагменти програми

```
/**
 * @author <Георгій>
 *
 */
public class Helper {

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

	public static ContainerList<PoliceFile> WaysToAddData(ContainerList<PoliceFile> A)
/// Спосб додавання елементів
	{

		int  a;
		System.out.print("\n 1: З клавіатури\n 2: З Файлу\n Ваш вібір:   ");
		a=in.nextInt();
		switch (a)
		{
		case 1:
			add_from_keyboard(A);
			break;
		case 2:
			A=Console_program.serializator.deserializtionXML();
			break;

		}
		return A;
	}
	//////////////////////////////////////////////////////
    private static ContainerList<PoliceFile>  add_from_keyboard(ContainerList<PoliceFile>
A )
		{

	int  a;

	System.out.print("\n Скільки ви плануєте додати в'язнів: ");
		int n=in.nextInt();
		System.out.print("\n Додати:\n 1: В початок \n 2: В кінець \n 3: ? \nВаш
вібір:   ");
		a=in.nextInt();
		for(int i=0;i<n;i++)
		{
			PoliceFile  add_from_keyboard=new PoliceFile();
			add_from_keyboard.EL(add_from_keyboard,false);
		  switch (a)
		  {
		case 1:
			A.addOnBack(add_from_keyboard);
			break;
		case 2:
			A.add(add_from_keyboard);
			break;
```

```
          }System.out.print("\n!записано!\n");
        } System.out.print("\nВаші дані успішно записані\n");
        return A;
  }
////////////////////////////////////////////////////
public static void  show(ContainerList<PoliceFile> List)
        {
               PoliceFile PF=new PoliceFile();
              PF.printTableHead();

              for (var L:List) {
              L.show();
        }
  }
//////////////////////////////////////////////////////////////////////
public static void posuk(ContainerList<PoliceFile> List)////  пошуку
   {
         if (List.size()==0) {
               System.out.print("Список пустий");
               return ;
         }
         String text=new String();
         System.out.print("Шукати:\n"+
                          "  1. Імя\n"+
                            "  2. по-Батькові\n"+
                          "  3. Прізвище\n"+
                            "  4. Index\n"+
                            "in: ");
         int a=in.nextInt();
         int i=0;
              switch(a)
              {
              case 1:
                      System.out.print("Введіть Імя : ");
                            text=in.next();

               for (var L:List)
               {
                  if (L.getName().equals(text))
                       {
                       System.out.print("Позиція вашаго елемента: "+i+"\n");
                       L.printTableHead();
                       L.show();
                       return ;
                       }i++;
              }
                System.out.print("Ваш елемент незнайдено");
               break;
              case 2:
                      System.out.print("Введіть по-Батькові : ");
                      text=in.next();

               for (var L:List)
               {
                        if (L.getSurname().equals(text))
                             {
                             System.out.print("Позиція вашаго елемента: "+i+"\n");
                             L.printTableHead();
                             L.show();
                             return ;
                             }i++;
                  }
```

```
                    System.out.print("Ваш елемент незнайдено");
                   break;
                 case 3:
                        System.out.print("Введіть прізвище : ");
                        text=in.next();

                        for (var L:List)
                        {
                           if (L.getLastname().equals(text))
                                {
                                System.out.print("Позиція вашаго елемента: "+i+"\n");
                                L.printTableHead();
                                L.show();
                                return ;
                                }i++;
                        }
                    System.out.print("Ваш елемент незнайдено");
                   break;
                 case 4:
                        System.out.print("Введіть index : ");
                        a=in.nextInt();
                   try
                    {
                        System.out.print("\n\n");
                     List.getNode(a).show();
                    }
                        catch(Exception e)     {   System.out.print("Ваш елемент
незнайдено");}
                   break;
                 }

        }

    ////////////////////////////////
       public static void UpdateData(ContainerList<PoliceFile> List ){
              int size=List.size();
              if (size==0)
              {
              System.out.print("\n\n Список пустий\n\n");
              return;
              }

              System.out.print("Введть індекс: ");
              int index=in.nextInt();
              if(index<1||index>size)
              {
                     System.out.print("\nТакого злочинця незнайдено\n");
                     return;
              }
              PoliceFile PF=List.getNode(index);
              PF.show();
               List.Substitute(PF.EL(PF,true), index);

        }
       public static void romove(ContainerList<PoliceFile> List) {
              if (List.size()==0)
              {
              System.out.print("\n\n Список пустий\n\n");
              return;
              }
             System.out.print("\nВведіть номер планує видалити?: ");
             int n=in.nextInt();
```

```
          List.removeNode(n);

       }

////////////////////////////////////////////////////////////////////////
    public static void Sort(ContainerList<PoliceFile> List)//// помилку пошуку
        {
              int size=List.size();
              if (size==0) {
                    System.out.print("Список пустий");
                    return ;
              }
              System.out.print("Сортувати:\n"+
                               "  1. Ім`я\n"+
                                 "  2. по-Батькові\n"+
                               "  3. Прізвище\n"+
                                 "  4. датою народження\n"+
                               "  5. датою останнього позбавлення волі\n"+
                                 "  6. датою останнього звільнення\n"+
                                  "in: ");

              int a=in.nextInt();
              switch(a)
                 {
                 case 1:
                    List.sort(new Comparator<PoliceFile>() {
                         @Override
                         public int compare(PoliceFile o1, PoliceFile o2) {
                            return  comparison (o1.getName(), o2.getName());
                         }
                     });
                    break;
                 case 2:
                    List.sort(new Comparator<PoliceFile>() {
                         @Override
                         public int compare(PoliceFile o1, PoliceFile o2) {

                            return  comparison(o1.getSurname(), o2.getSurname()) ;

                         }
                     });
              break;

                 case 3:
                    List.sort(new Comparator<PoliceFile>() {
                         @Override
                         public int compare(PoliceFile o1, PoliceFile o2) {
                               return comparison(o1.getLastname(), o2.getLastname());
                         }
                     });
              break;
                 case 4:
                    List.sort(new Comparator<PoliceFile>() {
                         @Override
                         public int compare(PoliceFile o1, PoliceFile o2) {
                               return comparison(o1.getDateOfBirth(),
o2.getDateOfBirth());
                         }
                     });
              break;
                 case 5:
                    List.sort(new Comparator<PoliceFile>() {
```

```
                        @Override
                        public int compare(PoliceFile o1, PoliceFile o2) {
                              return comparison(o1.getDateOfLastImprisonment(),
o2.getDateOfLastImprisonment());
                        }
                    });
              break;
                 case 6:
                    List.sort(new Comparator<PoliceFile>() {
                        @Override
                        public int compare(PoliceFile o1, PoliceFile o2) {
                              return comparison(o1.getDateOfLastreLease(),
o2.getDateOfLastreLease());
                        }
                    });
              break;
                 }


                    System.out.print(" *сортування завершено ");
       }
      ////////////////////////////////////////////////////////////////
      public static int comparison(String a,String b)
          {
            int len=0;
            if(a.length()<b.length())len=a.length();
            else len=b.length();
            for (int i=0;i<len;i++)
            {
                  if (a.charAt(i)>b.charAt(i)) return 1;
                  if (a.charAt(i)<b.charAt(i)) return -1;
            }
            if(a.length()<b.length())return -1;
            else if (a.length()>b.length()) return 1;
            else return 0;
          }

      public static int comparison(Date a,Date b)   //0 a=b  1 a>b 2 a<b
    {

      if(a.getYear()==b.getYear())
      {
       if(a.getMoon()==b.getMoon())
       {
             if(a.getDay()==a.getDay()) return 0;
             else if( a.getDay()>b.getDay()) return 1;
          else return -1;
       }
             else if( a.getMoon()>b.getMoon()) return 1;
          else return -1;
      }
      else if( a.getYear()>b.getYear()) return 1;
      else return -1;

    }
} ////////////////////////////////////////////////////////////////
public class ContainerList<T  extends  Serializable >
            implements List<T>,Iterable<T>   {
      private static final long serialVersionUID = 1L;

      public Element<T> head = null ;//head
      public Element<T> current = null;
```

```
        public Element<T> last = null;
        public Element<T> previous = null;



////////////////////////////////////////////////////
          public ContainerList() {  }
////////////////////////////////////////////////
          public ContainerList(T tab[]) {
              for (T a : tab) {

                  this.add(a);
              }
          }
//////////////////////////////////////////////////

……..

        public void sort (Comparator<PoliceFile> c    )
        {
              ArrayList<T> list=new ArrayList<>();

               current = head;
               while (current != null) {
                    list.add( current.num);
                   current = current.next;
               }

       list.sort((Comparator<? super T>) c);
         clear();

         for(var a:list)
         {
               add(a);
         }

        }
……..
} ////////////////////////////////////
```

3 ВАРІАНТИ ВИКОРИСТАННЯ

Оновлена попередню програма.

Додано можливість сортувати елементи списку та можливість автоматичного режиму програми.

На початку роботи користувач повинен подати команду. Якщо передає команду «-auto»-то програма працює в автоматичному режимі. У такому режимі програма відкриває заготовлений файли та зчитує дані. Далі просто програма виконує вивід даних сортування та інше.

В іншому випадку програма працю в стандартному режимі із користувачем.

Сортування виконане для всіх полів.

```
-----------------------------------------------------------------------------------------------------------------

Кількість злочинців у базі: 7

Сортування за іменем завершено
|11111111111|        11111111|      1111111|     11.11.1900|     11.11.1900|          11.11.1900|     1.11.1900|
-----------------------------------------------------------------------------------------------------------------
|11111111111|        11111111|      1111111|     11.11.1900|     11.11.1900|          11.11.1900|     1.11.1900|
-----------------------------------------------------------------------------------------------------------------
|22222222222|        22222222|      2222222|     22.12.2222|     22.12.2222|          22.12.2222|     2.12.2222|
-----------------------------------------------------------------------------------------------------------------
|22222222222|        22222222|      2222222|     22.12.2222|     22.12.2222|          22.12.2222|     2.12.2222|
-----------------------------------------------------------------------------------------------------------------
|  333333333|           33333| 333333333333|     31.12.3333|     22.12.2222|          31.12.3333|     3.12.3333|
-----------------------------------------------------------------------------------------------------------------
| 3333333333|        33333333|      3333333|     31.12.3333|     31.12.3333|          31.12.3333|     3.12.3333|
|           |                |             |               |               |                    |     3.12.3333|
-----------------------------------------------------------------------------------------------------------------
|       йцук|          йцуукк|     йццйуцйц|     31.12.4444|     31.12.7777|          31.12.8888|     5.12.5555|
-----------------------------------------------------------------------------------------------------------------


Список очищено

-----------------------------------------------------------------------------------------------------------------
|   Ім'я    |   По-батькові   |  Прізвище   |     дата      |     дата      |        дата          |     дати     |
|           |                 |             |  народження   |     ОПВ       |останнього звільнення|   судимостей |
-----------------------------------------------------------------------------------------------------------------


Роботу завершено. Будьте здорові
```

Рисунок 1. Виконання роботи програми автоматичному режимі

```
-----------------------------------------------------------------------------------------------------------------
|   Ім'я    |   По-батькові   |  Прізвище   |     дата      |     дата      |        дата          |     дати     |
|           |                 |             |  народження   |     ОПВ       |останнього звільнення|   судимостей |
-----------------------------------------------------------------------------------------------------------------
|11111111111|        11111111|      1111111|     11.11.1900|     11.11.1900|          11.11.1900|     1.11.1900|
-----------------------------------------------------------------------------------------------------------------
|11111111111|        11111111|      1111111|     11.11.1900|     11.11.1900|          11.11.1900|     1.11.1900|
-----------------------------------------------------------------------------------------------------------------
|22222222222|        22222222|      2222222|     22.12.2222|     22.12.2222|          22.12.2222|     2.12.2222|
-----------------------------------------------------------------------------------------------------------------
|22222222222|        22222222|      2222222|     22.12.2222|     22.12.2222|          22.12.2222|     2.12.2222|
-----------------------------------------------------------------------------------------------------------------
| 3333333333|        33333333|      3333333|     31.12.3333|     31.12.3333|          31.12.3333|     3.12.3333|
|           |                |             |               |               |                    |     3.12.3333|
-----------------------------------------------------------------------------------------------------------------
|  333333333|           33333| 333333333333|     31.12.3333|     22.12.2222|          31.12.3333|     3.12.3333|
-----------------------------------------------------------------------------------------------------------------
|       йцук|          йцуукк|     йццйуцйц|     31.12.4444|     31.12.7777|          31.12.8888|     5.12.5555|
-----------------------------------------------------------------------------------------------------------------
```

Рисунок 2 – Сортування списку за датою народження

ВИСНОВКИ

Під час виконання лабораторної роботи було набуто навички роботи створення сортування контейнера списку та демонстрування роботи програми в автоматичному режимі.