Тема 15 Колекції в Java

Мета:

* Ознайомлення з бібліотекою колекцій *Java SE*.
* Використання колекцій для розміщення об'єктів розроблених класів.

**1 ВИМОГИ**

**1.1 Розробник**

Інформація про розробника:

- Гряник Георгій Володимирович

- КІТ-119Д;

- 6 варіант.

**1.2 Загальне завдання**

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/#_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/#_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.

2 ОПИС ПРОГРАМИ

2.1 Засоби ООП

Java Collections Framework - набір зв'язаних класів та інтерфейсів, які реалізують commonly reusable collection структур даних.

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 «СontainerList» - 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 Obshchak – загальна область пам’яті. ProcessProcesses- клас контролювання процесів: виклик процесів та завершення.

2.3 Важливі фрагменти програми **/\*\***

**\* @author <Георгiй>**

**\***

**\*/**

Console\_program\_auto()

{

start();

}

**public** **static** Serializator<ArrayList<PoliceFile>> *serializator*=**new** Serializator<ArrayList<PoliceFile>>();

**public** **void** run() ///функціє проводить координування по можливостям програм

{

///ArrayList<PoliceFile> List=new ArrayList<PoliceFile>();

**try** {

Helper.*ReadFile*();

//List=serializator.deserializtionAuto();

Helper.*show*(Obshchak.*List*);

System.***out***.println("\nКількість злочинців у базі: "+Obshchak.*List*.size());

Obshchak.*List*.sort(**new** Comparator<PoliceFile>() { **public** **int** compare(PoliceFile o1, PoliceFile o2) {**return** Helper.*comparison*(o1.getSurname(), o2.getSurname()) ; } });

System.***out***.println("\nСортування за іменем завершено");

**for** (**var** PF : Obshchak.*List*.toArray())

{

**var** PFD=(PoliceFile)PF;

PFD.show();

}

System.***out***.println("\nYв'язнені не молодше 20 років з прізвищем, що починається з голосної та містить комбінацію \"ко\". Наприклад: Архипенко, Ішкова, Єрмаков, Янковський.");

Helper.*AppliedTask*();

Obshchak.*List*.clear();

System.***out***.print("\n\nСписок очищено");

Helper.*show*(Obshchak.*List*);

System.***out***.print("\n\nРоботу завершено. Будьте здорові");

}**catch**(Exception e) { System.***out***.println(" EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR EROR ");

System.***out***.print("\n\n\nТрапилась помилка. Але тепер все добре!!\n\n");

System.***out***.println(e);}

}

**3 ВАРІАНТИ ВИКОРИСТАННЯ**

Аналог попередньої роботи з використання замість власного контейнера колекцією Java.

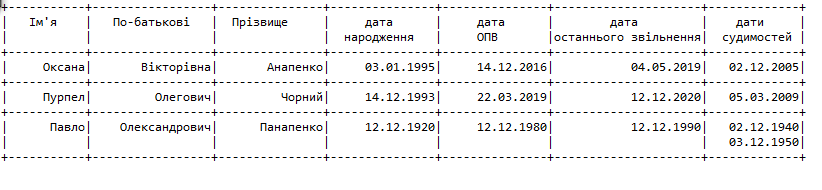


Рисунок 1. Вивід даних (частина великого списку)

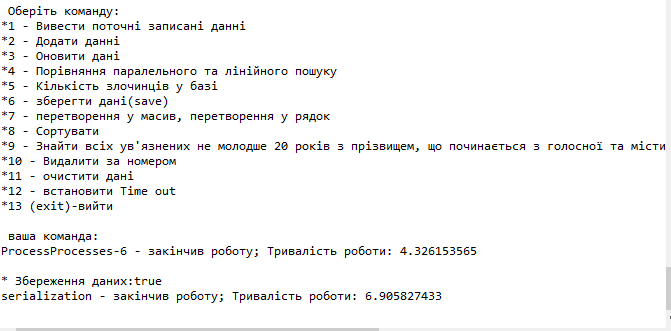


Рисунок 2. Результат роботи паралельного виконання процесів

ВИСНОВКИ

Під час виконання лабораторної роботи було набуто навички роботи з роботою Колекції в Java