Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №2

Колекції. Множина HashSet. Асоціативні масиви Map.

Виконали
студенти групи ІА-23:
Семашко Олександр,
Ширяєв Даніїл,
Степанов Нікіта.

Перевірив:
Колеснік В. М.

Київ 2022

Хід роботи:

1. Ознайомитись з javadoc для наступних інтерфейсів,класів та методів:
    - Set
    - HashSet
    - Object.equals(), Object.hashCode()
    - Map
    - HashMap

2. Виконати завдання лабораторної роботи №10, замінивши списки List (ArrayList та LinkedList) на множини Set (HashSet). Проаналізувати предметну область та на власний розсуд додати функціональність, для реалізації якої використати Map (TreeMap або HashMap).

```
public class Main {

    public static void main(String[] args) {
        FilmsDataBase base = new FilmsDataBase();

        Actor bob = new Actor("Bob", new String[]{"Mountain", "Cave"});
        base.appendWithActor(bob);
        Actor dave = new Actor("Dave", new String[]{"Mountain", "Desert"});
        base.appendWithActor(dave);
        Actor nik = new Actor("Nik", new String[]{"Mountain", "Desert", "Cave", "Rose"});
        base.appendWithActor(nik);
        Actor actorWhoNeverPlayed = new Actor("ActorWhoNeverPlayed", new String[]{});
        base.appendWithActor(actorWhoNeverPlayed);

        Film mountain = new Film("Mountain", new String[]{"Bob", "Dave", "Nik"});
        base.appendWithFilm(mountain);
        Film cave = new Film("Cave", new String[]{"Bob", "Nik"});
        base.appendWithFilm(cave);
        Film desert = new Film("Desert", new String[]{"Dave", "Nik"});
        base.appendWithFilm(desert);
        Film rose = new Film("Rose", new String[]{"Nik"});
        base.appendWithFilm(rose);

        System.out.println(Task1.getActorWhoNeverPlayed(base));
        System.out.println(Task2.actorsGivenActorHasEverPlayedWith(bob, base));
        System.out.println(Task3.filmWithBiggestActorsNumber(base));

        Task4.renameActor(bob, "Harry", base);
        Task4.renameActor(dave, "George", base);
        System.out.println("----------------------------");
```

```
        System.out.println(Task1.getActorWhoNeverPlayed(base));
        System.out.println(Task2.actorsGivenActorHasEverPlayedWith(bob, base));
        System.out.println(Task3.filmWithBiggestActorsNumber(base));
    }
}
```

```
import ...

13 usages
public class Actor implements Cinema {
    8 usages
    final private Map<String, Object> actor;

    4 usages
    Actor(String name, String[] films) {
        actor = new HashMap<>();
        actor.put("name", name);
        actor.put("films", new HashSet<>(Set.of(films)));
    }

    1 usage
    Actor(String name, HashSet<String> films) {
        actor = new HashMap<>();
        actor.put("name", name);
        actor.put("films", films);
    }

    10 usages
    public Map<String, Object> info() { return actor; }

    @Override
    public String toString() { return actor.get("name").toString(); }
}
```

```
import java.util.Map;

8 usages  2 implementations
public interface Cinema {
    10 usages  2 implementations
    Map<String, Object> info();
}
```

```
import ...

9 usages
public class Film implements Cinema {
    5 usages
    final private Map<String, Object> film;

    4 usages
    Film(String name, String[] actors) {
        film = new HashMap<>();
        film.put("name", name);
        film.put("actors", new HashSet<>(Set.of(actors)));
    }

    10 usages
    public Map<String, Object> info() { return film; }

    @Override
    public String toString() { return film.get("name").toString(); }
}
```

```java
import ...

6 usages
public class FilmsDataBase {
    6 usages
    final private Map<String, Set<Cinema>> base;

    1 usage
    FilmsDataBase() {
        base = new HashMap<>();
        base.put("filmsBase", new HashSet<>());
        base.put("actorsBase", new HashSet<>());
    }

    4 usages
    public void appendWithFilm(Film film) { base.get("filmsBase").add(film); }

    4 usages
    public void appendWithActor(Actor actor) { base.get("actorsBase").add(actor); }

    6 usages
    public Map<String, Set<Cinema>> data() { return base; }
}
```

```java
import java.util.HashSet;

2 usages
public class Task1 {
    2 usages
    static String getActorWhoNeverPlayed(FilmsDataBase base) {
        for (Cinema actor : base.data().get("actorsBase")) {
            if (((HashSet) actor.info().get("films")).isEmpty()) {
                return actor + " never played in film";
            }
        }
        return "Every actor played in film";
    }
}
```

```java
import ...

2 usages
public class Task2 {
    2 usages
    static String actorsGivenActorHasEverPlayedWith(Actor givenActor, FilmsDataBase base) {
        Set<String> actors = new TreeSet<>();
        for (Cinema film : base.data().get("filmsBase")) {
            for (Object actor : ((HashSet) film.info().get("actors"))) {
                if (actor.toString().equals(givenActor.toString())) {
                    actors.addAll(((HashSet) film.info().get("actors")));
                }
            }
        }
        actors.remove(givenActor.toString());
        StringBuilder builder = new StringBuilder(String.format("Actors who ever played with %s: ", givenActor));
        for (String actor : actors) {
            builder.append(actor);
            builder.append(", ");
        }
        return builder.toString();
    }
}
```

```java
import java.util.HashSet;

2 usages
public class Task3 {
    2 usages
    static String filmWithBiggestActorsNumber(FilmsDataBase base) {
        int biggestActorsNumber = 0;
        StringBuilder output = new StringBuilder();
        for (Cinema film : base.data().get("filmsBase")) {
            if (((HashSet) film.info().get("actors")).size() > biggestActorsNumber) {
                biggestActorsNumber = ((HashSet) film.info().get("actors")).size();
                output.replace( start: 0, output.capacity(), film.toString());
            }
        }
        return output.insert( offset: 0,  str: "Film with biggest actors number: ").toString();
    }
}
```

```java
import java.util.HashSet;

2 usages
public class Task4 {
    2 usages
    static void renameActor(Actor oldActor, String newName, FilmsDataBase base) {
        String oldName = oldActor.info().get("name").toString();
        Actor newActor = new Actor(newName, (HashSet) oldActor.info().get("films"));

        oldActor.info().replace("name", newName);
        base.data().get("actorsBase").remove(oldActor);
        base.data().get("actorsBase").add(newActor);

        for (Cinema film : base.data().get("filmsBase")) {
            ((HashSet) film.info().get("actors")).remove(oldName);
            ((HashSet) film.info().get("actors")).add(newName);
        }
    }
}
```

```
ActorWhoNeverPlayed never played in film
Actors who ever played with Bob: Dave, Nik,
Film with biggest actors number: Mountain
----------------------------
ActorWhoNeverPlayed never played in film
Actors who ever played with Harry: George, Nik,
Film with biggest actors number: Mountain
```

Висновок: на цій лабораторній роботі ми познайомилися на навчилися працювати з класом HashSet та його методами hashCode() і equals(), асоціативними масивами Map, покращили навички кодування на Java.