Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра Інформаційних систем та технологій

ЛАБОРАТОРНА РОБОТА № 1

з курсу «Структури даних та алгоритми»

Лабораторна робота №1: Колекції. Множина TreeSet

ВИКОНАЛИ:
Студенти 1 курсу ФІОТ,
групи ІА-24
Кармазіна Анастасія,
Шкарніков Антон,
Сотніков Олексій.
ПЕРЕВІРИВ:
Колеснік В. М.

Київ - 2023

Хід роботи:
1. Ознайомитись з javadoc для наступних інтерфейсів та класів:
    - Set
    - TreeSet
    - Comparable
    - Comparator
    - SortedSet
    - NavigableSet

2. Виконати завдання попередньої лабораторної роботи, замінивши списки List (ArrayList та LinkedList) на множини Set (TreeSet). При цьому необхідно щонайменше один раз використати Comparable та щонайменше один раз – Comparator.

Результат виконання роботи:

Лістинг

Код файлу/ Comment.java

```
public class Comment {
    private String text;
    private int likes;
    private int dislikes;
    public Comment(String text) {
        this.text = text;
    }

    public void like() {
        likes++;
    }

    public void dislike() {
        dislikes++;
    }

    public String getText() {
        return text;
    }

    public void setText(String text) {
        this.text = text;
    }

    public int getLikes() {
        return likes;
    }
```

```
    public void setLikes(int likes) {
        this.likes = likes;
    }

    public int getDislikes() {
        return dislikes;
    }

    public void setDislikes(int dislikes) {
        this.dislikes = dislikes;
    }

    @Override
    public String toString() {
        return "Comment{" +
                "text='" + text + '\'' +
                ", likes=" + likes +
                ", dislikes=" + dislikes +
                '}';
    }
}
```

Код файлу /CommentComparator.java

```
import java.util.Comparator;

public class CommentComparator implements Comparator<Comment> {
    @Override
    public int compare(Comment o1, Comment o2) {
        if (o1.getText().length() == o2.getText().length()){
            return 0;
        }
        if (o1.getText().length() > o2.getText().length()){
            return -1;
        }
        return 1;
    }
}
```

Код файлу/Video.java

```
import java.util.*;

public class Video implements Comparable<Video> {
    private String name;
    private final String URL;
    private int views;
    private int likes;
    private int dislikes;
    private TreeSet<Comment> comments;
```

```
public Video(String name, CommentComparator comparator) {
    this.name = name;
    URL = "https://someblog.com/v/" + name.toLowerCase();
    comments = new TreeSet<Comment>(comparator);
}

public void view(){
    views++;
}

public void like() {
    likes++;
}

public void dislike() {
    dislikes++;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getURL() {
    return URL;
}

public int getViews() {
    return views;
}

public void setViews(int views) {
    this.views = views;
}

public int getLikes() {
    return likes;
}

public void setLikes(int likes) {
    this.likes = likes;
}

public int getDislikes() {
    return dislikes;
}

public void setDislikes(int dislikes) {
```

```
        this.dislikes = dislikes;
    }

    public void addComment(Comment newComment) {
        comments.add(newComment);
    }

    public TreeSet<Comment> getComments() {
        return comments;
    }

    public boolean hasMoreLikedComment() {
        for (Comment comment: comments) {
            if (likes < comment.getLikes()) {
                return true;
            }
        }
        return false;
    }

    @Override
    public String toString() {
        return "Video{" +
                "name='" + name + '\'' +
                ", URL='" + URL + '\'' +
                ", views=" + views +
                ", likes=" + likes +
                ", dislikes=" + dislikes +
                ",\ncomments=" + comments +
                '}';
    }

    @Override
    public int compareTo(Video o) {
        if (this.views == o.getViews()){
            return 0;
        }
        if (this.views < o.getViews()){
            return -1;
        }
        return 1;
    }
}
```

Код файлу/VideoBlog.java

```
import java.util.Iterator;
import java.util.TreeSet;

public class VideoBlog {
    private String videoBlogger;
```

```java
private TreeSet<Video> videos;

public VideoBlog(String videoBlogger) {
    this.videoBlogger = videoBlogger;
    this.videos = new TreeSet<Video>();
}

public String getVideoBlogger() {
    return videoBlogger;
}

public void setVideoBlogger(String videoBlogger) {
    this.videoBlogger = videoBlogger;
}
public void addVideo(Video newVideo) {
    this.videos.add(newVideo);
}

public void removeVideo(Video video) {
    videos.remove(video);
}

public TreeSet<Video> getVideos() {
    return videos;
}

public int getAllViews() {
    int viewsCount = 0;
    for (Iterator<Video> i = videos.iterator(); i.hasNext();){
        Video v = i.next();
        viewsCount += v.getViews();
    }
    return viewsCount;
}

public TreeSet<Video> getMostDislikedVideo() {
    TreeSet<Video> mostDislikedVideos = new TreeSet<Video>();
    int maxDislikes = 0;
    for (Iterator i = videos.iterator(); i.hasNext();) {
        Object o = i.next();
        if (o instanceof Video) {
            Video v = (Video) o;
            int dislikes = v.getDislikes();
            if (dislikes != 0 && dislikes >= maxDislikes) {
                maxDislikes = dislikes;
                mostDislikedVideos.add(v);
            }
        }
    }
    return mostDislikedVideos;
}
```

```
    @Override
    public String toString() {
        return "VideoBlog{" +
                "videoBlogger='" + videoBlogger + '\'' +
                ", videos=" + videos +
                '}';
    }
}
```

Код файлу/Main.java

```
public class Main {
    public static void main(String[] args) {
        CommentComparator commentComparator = new CommentComparator();

        //-----------------------Comment-----------------------//
        Comment comment1 = new Comment("Didn't thought that name of the
video would be so wrong...");
        comment1.setLikes(100);
        comment1.setDislikes(1);

        Comment comment2 = new Comment("L0L");
        comment2.setLikes(2);
        comment2.setDislikes(4);

        Comment comment3 = new Comment("Qwerty");
        comment2.setLikes(2);

        Comment comment4 = new Comment("Some Comment");
        comment2.setDislikes(1);

        //-----------------------Video-----------------------//
        Video video1 = new Video("FirstVideo", commentComparator);
        video1.setViews(50);
        video1.setLikes(5);
        video1.setDislikes(2);

        Video video2 = new Video("SecondVideo", commentComparator);
        video2.setViews(35);
        video2.setLikes(15);
        video2.setDislikes(2);

        Video video3 = new Video("ThirdVideo", commentComparator);
        video3.setViews(70);
        video3.setLikes(10);
        video3.setDislikes(0);

        Video video4 = new Video("CoolVideo", commentComparator);
        video4.setViews(101);
        video4.setLikes(1);
```

```
        video4.setDislikes(100);

        Video video5 = new Video("StrangeVideo", commentComparator);
        video5.setViews(60);
        video5.setLikes(50);
        video5.setDislikes(3);

        Video video6 = new Video("Should I do video?", commentComparator);
        video6.setViews(200);
        video6.setLikes(75);
        //-------------------------------------------------------//

        video4.addComment(comment1);
        video4.addComment(comment2);
        video4.addComment(comment3);
        video5.addComment(comment4);

        //----------------------VideoBlog-----------------------//
        VideoBlog videoBlog1 = new VideoBlog("CoolThings");
        VideoBlog videoBlog2 = new VideoBlog("StupidThings");
        VideoBlog videoBlog3 = new VideoBlog("SomeThings");

        videoBlog1.addVideo(video1);
        videoBlog1.addVideo(video2);
        videoBlog1.addVideo(video3);

        videoBlog2.addVideo(video4);
        videoBlog2.addVideo(video5);

        videoBlog3.addVideo(video6);

        System.out.println("Sort videoBlog" + " {" +
videoBlog1.getVideoBlogger() + "} by Views:");
        for (Video video: videoBlog1.getVideos()) {
            System.out.println("\t" + video.getName() + " " +
video.getViews());
        }

        System.out.println("\n");

        System.out.println("Sort comment" + " {" + video4.getName() + "}
by length:");
        for (Comment comment: video4.getComments()){
            System.out.println("\t" + comment.getText());
        }


System.out.println("\n==========================================\n");

        printHasMoreLikedComment(video4);
        printHasMoreLikedComment(video5);
```

```
System.out.println("\n=============================================\n");

        printAllViews(videoBlog1);
        printAllViews(videoBlog2);
        printAllViews(videoBlog3);


System.out.println("\n=============================================\n");

        printMostDislikedVideo(videoBlog1);
        printMostDislikedVideo(videoBlog2);
        printMostDislikedVideo(videoBlog3);


System.out.println("\n=============================================\n");
    }

    private static void printHasMoreLikedComment(Video video) {
        System.out.println("Video " + video.getName() +
                " at URL \"" + video.getURL() +
                "\" has more liked comment: " +
video.hasMoreLikedComment());
    }

    private static void printAllViews(VideoBlog videoBlog) {
        System.out.println("Videoblogger " + videoBlog.getVideoBlogger()
+
                " has " + videoBlog.getAllViews() +
                " views in total on his videoblog.");
    }

    private static void printMostDislikedVideo(VideoBlog videoBlog) {
        System.out.println("Most disliked videos on videoblog " +
videoBlog.getVideoBlogger() +
                ": " + videoBlog.getMostDislikedVideo());
    }

}
```

Опис принципу роботи

Інтерфейси **Comparator** та **Comparable** використовують для порівняння власних типів даних, оскільки коли потрібно порівняти два значення int чи string можна використати вбудовані методи, а для порівняння деяких об’єкті, наприклад типу Person, потрібно вказувати додаткові інструкції

Реалізація інтерфейсу **Comparable**:
- імплементуємо інтерфейс Comparable
- створюємо метод compareTo який буде повертати значення 0, 1, -1

```
public class Video implements Comparable<Video> {
    . . .
    public int compareTo(Video o) {
       if (this.views == o.getViews()){
           return 0;
       }
       if (this.views < o.getViews()){
           return -1;
       }
    return 1;
    }
    . . .
}
```

Таким чином ми явно вказали що для об'єкта Video при порівнянні потрібно використовувати поле `views`

Реалізація інтерфейсу **Comparator**:
- створюємо новий клас CommentComparator
- імплементуємо інтерфейс Comparable та реалізуємо метод **compare**

```
public class CommentComparator implements Comparator<Comment> {
    @Override
    public int compare(Comment o1, Comment o2) {
        if (o1.getText().length() == o2.getText().length()){
            return 0;
        }
        if (o1.getText().length() > o2.getText().length()){
            return -1;
        }
        return 1;
    }
}
```

Таким чином вказуємо що для порівняння об'єктів Comment потрібно використовувати метод `getText()` порівнюючи довжини коментарів

Інтерфейс **Comparable** простіший і дозволяє швидко і просто порівнювати об'єкти, а **Comparator** трохи важчий в реалізації дозволяє зробити код більш гнучкий, відділити об'єкт від "правил" його порівняння та створювати нові правила порівняння за необхідністю

```
System.out.println("Sort videoBlog" + " {" + videoBlog1.getVideoBlogger()
+ "} by Views:");
for (Video video: videoBlog1.getVideos()) {
    System.out.println("\t" + video.getName() + " " + video.getViews());
}
```

```
System.out.println("\n");

System.out.println("Sort comment" + " {" + video4.getName() + "} by
length:");
for (Comment comment: video4.getComments()){
    System.out.println("\t" + comment.getText());
}
```

```
C:\Users\Nastya\.jdks\openjdk-19\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2
Sort videoBlog {CoolThings} by Views:
    SecondVideo 35
    FirstVideo 50
    ThirdVideo 70


Sort comment {CoolVideo} by length:
    Didn't thought that name of the video would be so wrong...
    Qwerty
    L0L

================================================

Video CoolVideo at URL "https://someblog.com/v/coolvideo" has more liked comment: true
Video StrangeVideo at URL "https://someblog.com/v/strangevideo" has more liked comment: false

================================================

Videoblogger CoolThings has 155 views in total on his videoblog.
Videoblogger StupidThings has 161 views in total on his videoblog.
Videoblogger SomeThings has 200 views in total on his videoblog.

================================================

Most disliked videos on videoblog CoolThings: [Video{name='SecondVideo', URL='https://someblog.com/v
comments=[]}, Video{name='FirstVideo', URL='https://someblog.com/v/firstvideo', views=50, likes=5, d
comments=[]}]
Most disliked videos on videoblog StupidThings: [Video{name='StrangeVideo', URL='https://someblog.co
comments=[Comment{text='Some Comment', likes=0, dislikes=0}]}, Video{name='CoolVideo', URL='https://
```

Висновок:

Під час виконання даної лабораторно роботи ми використовували завдання попередньої лабораторної роботи, замінивши списки List (ArrayList та LinkedList) на множини Set (TreeSet).

Дізнались що у множинах Set кожен елемент зберігається лише одному екземплярі, а різні реалізації Set використовують різний порядок зберігання елементів. Якщо порядок зберігання важливий, використовуємо контейнер TreeSet, у якому об'єкти зберігаються відсортованими за зростанням у порядку порівняння. Закріпили отримані занання на практиці та переконалися в правильнотсі роботи програми, про що також свідчать скриншоти.