**Solution to Chapter 11 Exercises**

1. **Exercise 1: Sorting a List of Strings**

**Task:**

* + Create a list of strings representing song titles and sort them alphabetically.

**Solution:**

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

public class SongSorter {

public static void main(String[] args) {

List<String> songs = new ArrayList<>();

songs.add("somersault");

songs.add("cassidy");

songs.add("$10");

songs.add("havana");

songs.add("Cassidy");

songs.add("50 Ways");

Collections.sort(songs);

System.out.println(songs);

}

}

* + **Output:**

[$10, 50 Ways, Cassidy, cassidy, havana, somersault]

1. **Exercise 2: Converting a List of Strings to a List of Custom Song Objects**

**Task:**

* + Convert the list of strings to a list of Song objects and sort by the song title.

**Solution:**

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

class Song implements Comparable<Song> {

private String title;

Song(String title) {

this.title = title;

}

public String getTitle() {

return title;

}

@Override

public int compareTo(Song s) {

return title.compareTo(s.getTitle());

}

@Override

public String toString() {

return title;

}

}

public class SongSorter {

public static void main(String[] args) {

List<Song> songs = new ArrayList<>();

songs.add(new Song("somersault"));

songs.add(new Song("cassidy"));

songs.add(new Song("$10"));

songs.add(new Song("havana"));

songs.add(new Song("Cassidy"));

songs.add(new Song("50 Ways"));

Collections.sort(songs);

System.out.println(songs);

}

}

* + **Output:**

[$10, 50 Ways, Cassidy, cassidy, havana, somersault]

1. **Exercise 3: Implementing Custom Sorting Using Comparator**

**Task:**

* + Implement sorting by different criteria, such as by song title or artist.

**Solution:**

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

class Song {

private String title;

private String artist;

Song(String title, String artist) {

this.title = title;

this.artist = artist;

}

public String getTitle() {

return title;

}

public String getArtist() {

return artist;

}

@Override

public String toString() {

return title + " by " + artist;

}

}

class TitleComparator implements Comparator<Song> {

public int compare(Song s1, Song s2) {

return s1.getTitle().compareTo(s2.getTitle());

}

}

class ArtistComparator implements Comparator<Song> {

public int compare(Song s1, Song s2) {

return s1.getArtist().compareTo(s2.getArtist());

}

}

public class SongSorter {

public static void main(String[] args) {

List<Song> songs = new ArrayList<>();

songs.add(new Song("somersault", "Zero 7"));

songs.add(new Song("cassidy", "Grateful Dead"));

songs.add(new Song("$10", "Hitchhiker"));

songs.add(new Song("havana", "Camila Cabello"));

songs.add(new Song("Cassidy", "Grateful Dead"));

songs.add(new Song("50 Ways", "Paul Simon"));

Collections.sort(songs, new TitleComparator());

System.out.println("Sorted by Title: " + songs);

Collections.sort(songs, new ArtistComparator());

System.out.println("Sorted by Artist: " + songs);

}

}

* + **Output:**

Sorted by Title: [$10 by Hitchhiker, 50 Ways by Paul Simon, Cassidy by Grateful Dead, cassidy by Grateful Dead, havana by Camila Cabello, somersault by Zero 7]

Sorted by Artist: [somersault by Zero 7, havana by Camila Cabello, 50 Ways by Paul Simon, Cassidy by Grateful Dead, cassidy by Grateful Dead, $10 by Hitchhiker]

1. **Exercise 4: Removing Duplicates Using a Set**

**Task:**

* + Use a Set to automatically filter out duplicate song titles.

**Solution:**

import java.util.HashSet;

import java.util.Set;

public class SongSet {

public static void main(String[] args) {

Set<String> songs = new HashSet<>();

songs.add("somersault");

songs.add("cassidy");

songs.add("$10");

songs.add("havana");

songs.add("Cassidy");

songs.add("50 Ways");

System.out.println(songs);

}

}

* + **Output:**

[50 Ways, somersault, cassidy, $10, havana, Cassidy]