COM1028 Software Engineering
Testing Report Part 2

<<Stefanos Chatzakis
6481123, sc01396@surrey.ac.uk>>

1. Introduction 1
2. Requirements Testing 1
2.1. Requirements Test Plan 1
2.2. Evidence of the Testing 2

1. Introduction

This report is a test plan of how I carried a test for the user's actions. Moreover, I have only tested the functional requirements.

2. Requirements Testing

This section and its sub sections give the requirements testing for the project. In my project the following were wish list requirements F3, F6,F7 and F8 and these requirements were not implemented and therefore not included in the following test plan.

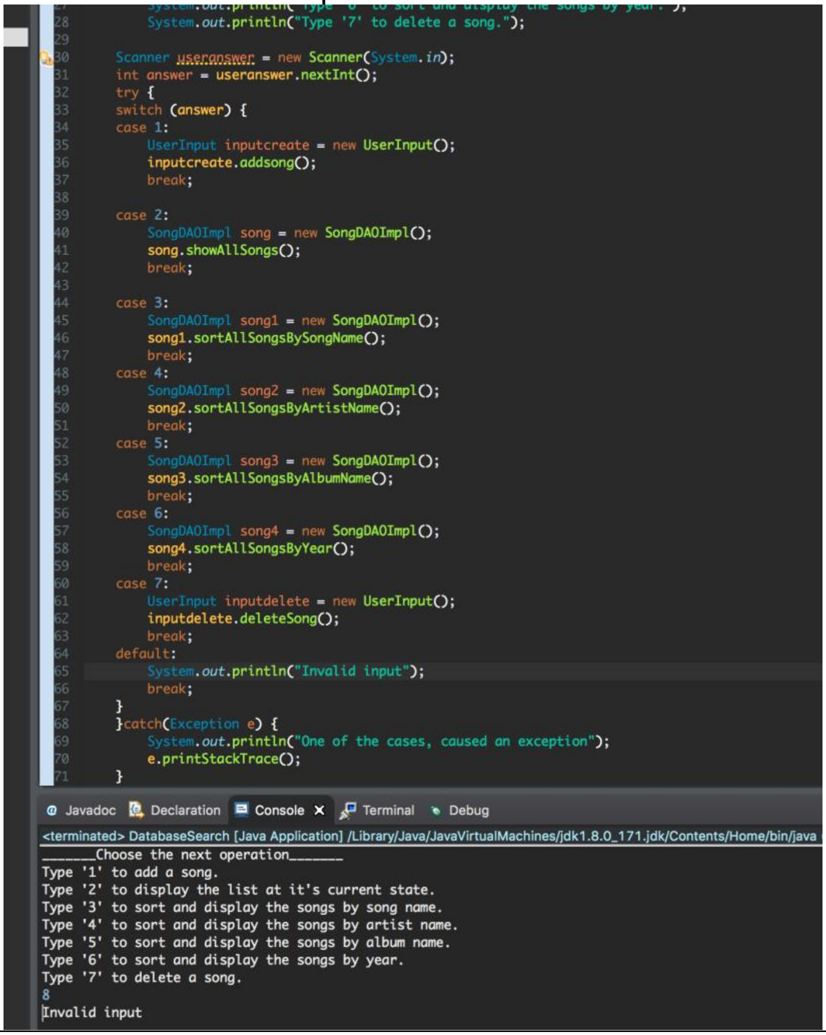
2.1. Requirements Test Plan

<<Explanation of this: see page 53 of IEEE Test Standard. It provides a Level Test Case outline. The following is the example table I would suggest using. The row in grey are the guidelines and should not be included in your submission. Please note there may be more than one test per requirement to cover different possible input combinations. You need to include a table for all your requirements here>>

Test Number	Requirement No	Preconditions/ Dependencies	Expected Inputs	Expected Results	Test Evaluation
1	F1	Preconditions: Existing elements to sort Dependencies: User's selection	Previously added song objects.	The expected result is to sort and display the list of music, depending on what the user's selection was.	The validation occurs once the list displays the correct the order that the user selected and also if the sorting is correct.
2	F2	Dependencies: Usaer's input	songName, artistName, albumName, year	Add new object to the list.	By, seeing the new Object added to the list with the attributes the user inputed.

3	F4	Dependencies: User's selection	Sort the list by artist	Set the list sort by artist	The validation occurs once the list displays songs alphabetically by artist
4	F5	Dependencies: User's selection	Sort the list by album	Set the list sort by album	The validation occurs once the list displays songs alphabetically by album

2.2. Evidence of the Testing

Test Number	Actual Output	Test Status
1	Invalid Argument, which means that the try/catch Test has passed. This also occurs if Test number 2,3 and 4 do not have the correct input. 	passed

```java
        System.out.println("Type '7' to delete a song.");

    Scanner useranswer = new Scanner(System.in);
    int answer = useranswer.nextInt();
    try {
    switch (answer) {
    case 1:
        UserInput inputcreate = new UserInput();
        inputcreate.addsong();
        break;

    case 2:
        SongDAOImpl song = new SongDAOImpl();
        song.showAllSongs();
        break;

    case 3:
        SongDAOImpl song1 = new SongDAOImpl();
        song1.sortAllSongsBySongName();
        break;
    case 4:
        SongDAOImpl song2 = new SongDAOImpl();
        song2.sortAllSongsByArtistName();
        break;
    case 5:
        SongDAOImpl song3 = new SongDAOImpl();
        song3.sortAllSongsByAlbumName();
        break;
    case 6:
        SongDAOImpl song4 = new SongDAOImpl();
        song4.sortAllSongsByYear();
        break;
    case 7:
        UserInput inputdelete = new UserInput();
        inputdelete.deleteSong();
        break;
    default:
        System.out.println("Invalid input");
        break;
    }
    }catch(Exception e) {
        System.out.println("One of the cases, caused an exception");
        e.printStackTrace();
    }
```

```
<terminated> DatabaseSearch [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_171.jdk/Contents/Home/bin/java
_______Choose the next operation_______
Type '1' to add a song.
Type '2' to display the list at it's current state.
Type '3' to sort and display the songs by song name.
Type '4' to sort and display the songs by artist name.
Type '5' to sort and display the songs by album name.
Type '6' to sort and display the songs by year.
Type '7' to delete a song.
8
Invalid input
```