

Rajob raihan Monmoy

University of Greenwich

[Date]



Table of Contents

[1 COMP1148 Coursework Feedback sheet 3](#_Toc476222528)

[2 Introduction 5](#_Toc476222529)

[3 Design and Development 6](#_Toc476222530)

[4 Testing and Faults 7](#_Toc476222531)

[5 Conclusions, further development and reflection 8](#_Toc476222532)

[5.1 Future innovations 8](#_Toc476222533)

[5.2 Reflection 8](#_Toc476222534)

[6 Appendix A – The commented code for deliverable A 9](#_Toc476222535)

[7 Appendix B – White Box Testing 10](#_Toc476222536)

# COMP1148 Coursework Feedback sheet

Name: Overall mark: %

There are **no specific marks for each stage** and your overall mark is based on **your** **performance as a whole**. The following table indicates which Grading Band your coursework falls into and why. You may receive marks in a higher band if your tutor feels some of the work justifies it.

Remember: You cannot get marks for a particular stage unless you have made a reasonable attempt at all of the previous stages. For example, if you miss out the testing (stage 4) you cannot get a mark over 60%, regardless of how good your innovations are.

|  |  |  |
| --- | --- | --- |
| **Grading Band** | **Indicative Grading Criteria** | **Your work** |
| **71-100%** | a very good attempt at all stages, with an excellent report |  |
| **61-70%** | a good attempt at all stages, with a good report |  |
| a very good attempt at all stages, but the report is weak |  |
| a very good attempt at all stages, but the program is hard to use |  |
| a very good attempt at all stages, but the code contains minor runtime errors |  |
| **41-60%** | a good attempt at stages 1-4, with a good report |  |
| a reasonable attempt at all stages, but the report is missing a section |  |
| a reasonable attempt at all stages, but the code contains serious runtime errors |  |
| **21-40%** | an attempt at stages 1-3, with a reasonable report |  |
| a reasonable attempt at further stages, but the report is missing several sections |  |
| a reasonable attempt at further stages, but the code cannot be run by your tutor |  |
| **11-20%** | an attempt at stages 1-2 |  |
| **0-10%** | an attempt at stage 1 |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Assessment Criteria** | Achieved well | Partially achieved | Not achieved |
| **The functionality of the programme.** |  | | |
| Does the system do what it is supposed to do? |  |  |  |
| **Usability:** |  | | |
| Is your system straightforward and easy to use? |  |  |  |
| Is it obvious to the user what to do? |  |  |  |
| Are all messages clear and unambiguous? |  |  |  |
| Is the output formatted appropriately? |  |  |  |
| Is bad input data handled appropriately? |  |  |  |
| Is the system free from crashes and uncaught exceptions? |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Quality of the Java code:** | |  | | |
| Inclusion of meaningful comments (other than for Stage 1). | |  |  |  |
| Use of sensible naming standards. | |  |  |  |
| Clear code layout and formatting. | |  |  |  |
| **Quality and completeness of the report:** | |  | | |
| Is the design documentation clear and concise? | |  |  |  |
| Have you included evidence of appropriate testing? | |  |  |  |
| Have you discussed any faults or failures? | |  |  |  |
| Have you reflected on the development process? | |  |  |  |
| **Feedback Comments** | | | | |
| **Interim Deliverables** | (Brief feedback on your work as the term progressed) | | | |
| **Final Deliverable** | (Feedback on your work as a whole – typically: a summary; an evaluation of your report; a suggestion of what could be improved in the program; and an indication of the best features of your coursework) | | | |

# Introduction

Please explain what you see the program should achieve.

# Design and Development

Please provide a description of how you designed and developed the final code with suitable screen shots of the program in operation.

# Testing and Faults

Please provide a summary of the white box testing (the actual table and results will be listed in appendix B) and a discussion of any faults and failures, including those that you managed to correct, and those which are still unresolved.

# Conclusions, further development and reflection

## Conclusions

Please give a summary of the program, i.e. what it does. Imagine this as a brief description that you might send to a prospective publisher.

## Future innovations

What you would do if you had another three months to work on the program?

## Reflection

Please write at least 400 words, answer either (a) **or** (b) from the following:

1. What did I actually achieve with this element of learning? Which were the most difficult parts, and why were they difficult for me? Which were the most straightforward parts, and why did I find these easy?
2. What have I got out of doing this element of the course? How have I developed my knowledge and skills? How do I see this element of the course helping me in the longer term?

**Option A**

**What did I actually achieve with this element of learning?**

**Which were the most difficult parts, and why were they difficult for me?**

**Which were the most straightforward parts, and why did I find these easy?**

**Option B**

**What have I got out of doing this element of the course?**

**How have I developed my knowledge and skills?**

**How do I see this element of the course helping me in the longer term?**

# Appendix A – The commented code for deliverable A

The commented version of the code that you that you submitted for interim deliverable A.

**Code**: CheckVideo.java

package coursework;

// import the classes of the abstract windows toolkit and

// swing to enable the use of components such as buttons

// and import event to make use of events such as actionPerformed event

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

//Extending Jframe class for using the functionality of the class and

//implementing ActionListener interface

public class CheckVideos extends JFrame implements ActionListener {

//initializing components for the GUI

JTextField trackNo = new JTextField(2);

TextArea information = new TextArea(6, 50);

JButton list = new JButton("List All Videos");

JButton check = new JButton("Check Video");

public CheckVideos() {

//set the layout to border layout

setLayout(new BorderLayout());

//set the jframe size

setBounds(100, 100, 400, 200);

//set the name for the jframe

setTitle("Check Videos");

//This is for disposing jframe but clicking the X button

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

//Create a Jpanel and add the components for top bar

JPanel top = new JPanel();

top.add(new JLabel("Enter Video Number:"));

top.add(trackNo);

top.add(check);

top.add(list);

//adding action listener to buttons

list.addActionListener(this);

check.addActionListener(this);

//adding panel to the north

add("North", top);

//create a panel and then add previously initialized textarea

//information. Then add the panel to the middle of jframe

JPanel middle = new JPanel();

information.setText(VideoData.listAll());

middle.add(information);

add("Center", middle);

//This is used so that user can not resize the jframe

setResizable(false);

//This makes the jframe visible

setVisible(true);

}

//this function is invoked when list or check button is clicked

public void actionPerformed(ActionEvent e) {

//if the clicked button is list button then call lisAll function

//from VideoData class

if (e.getSource() == list) {

information.setText(VideoData.listAll());

}

//if the clicked button is check button

else {

//take the data from trackNo textfield

String key = trackNo.getText();

//get the name from VideoData class using getName(Key) class

String name = VideoData.getName(key);

//getName(Key) class in VideoData returns null if no name found

if (name == null) {

information.setText("No such video number");

} else {

//if name found fill up the information textfield using

//the data gathered from VideoData class

information.setText(name + " - " + VideoData.getDirector(key));

//passes the rating data found from VideoData to the stars function

information.append("\nRating: "

+ stars(VideoData.getRating(key)));

information.append("\nPlay count: " + VideoData.getPlayCount(key));

}

}

}

//Produces stars for each rating. Ex: 5 rating then \*\*\*\*\* will be produced

private String stars(int rating) {

String stars = "";

for (int i = 0; i < rating; ++i) {

stars += "\*";

}

return stars;

}

}

# Appendix B – White Box Testing

White box test **table** and results – this should be updated from the version you submitted for interim deliverable C to cover any changes you have made to the code since then.

**Class**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | Clicked | Method | Output Expected | Output Obtained |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Class**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | Clicked | Method | Output Expected | Output Obtained |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Class**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | Clicked | Method | Output Expected | Output Obtained |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Class**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | Clicked | Method | Output Expected | Output Obtained |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Class**:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | Clicked | Method | Output Expected | Output Obtained |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |