~~RGU: School of Computing CMM024 Coursework~~

CMM024: Object-Oriented Programming 2021-2022 **Coursework** **Specification**

*This* *Coursework* *is* *an* *individual* *piece* *of* *work.* *All* *Coursework* *will* *be* *screened* *for* *evidence* *of* *Academic* *Misconduct* *(e.g.* *Collusion* *and* *Plagiarism).*

|  |  |
| --- | --- |
| **Hand-in** **Dates** | A zipped file of your NetBeans Java project by 09/12/2021 |
| **Learning** **Outcomes** **Assessed** | This coursework assesses:  • Analyse a set of system requirements and design, develop and test a solution, which satisfies those requirements using Java.  • Interpret and describe class behaviour and inter-class relationships in an object-oriented design using provided UML diagrams.  • Demonstrate knowledge and understanding of object-oriented design concepts and inter-class behaviour using appropriate notation; including class composition, association, and inheritance.  • Apply the principles of class composition, association and inheritance, to construct hierarchies of new classes including components required for graphical interfaces.  • Implement interactive object-oriented programs that incorporate a graphical user interface and hierarchies of classes, using an event handling model to handle user interaction with components. |
| **Deliverables** | A zipped file of your NetBeans Java project Details are discussed further on the last page. |
| **Software** **to** **be** **used** | *Apache* *NetBeans* *IDE* *version* *12* |
| **Submission** **Method** | The **Submission** **Dropbox** on the module CampusMoodle page, you should submit a zipped file of your NetBeans Java project  **Submission** **deadline** **is** **4:00PM** – a 30-minute grace period has been added on the Dropbox in the event you have any technical issues that prevent you from submitting at the deadline time. |
| **Feedback** **Arrangements** | Feedback will be available after 20 days from the submission deadline provided via the module CampusMoodle page |
| **Coursework** **Grade** | Provisional grades for this coursework will be available after 20 days from the submission deadline on the module CampusMoodle page.  Your grade for each element of the coursework can be found on Page 6: *Overview* *of* *Tasks* *for* *the* *Coursework*  Element grades will be added to give the grade for this coursework using the grid. |
| **Overall** **Module** **Grade** | This coursework contributes 100% of the overall module grade. |
| **Formative** |  |

1

~~RGU: School of Computing CMM024 Coursework~~

|  |  |
| --- | --- |
| **Feedback** **Opportunities** |  |
| **Questions?** **Contact** | Module Coordinator details – Jean Claude Golovine j.r.r.golovine1@rgu.ac.uk |
| **Moderation** **completed** **by** | Internal External |
| **Coursework** **Setter** | Jean Claude Golovine |
| **Notes** **on** **Penalties,** **Extensions** **and** **Deferrals** | Coursework submitted late without prior approval will be recorded as a Non-Submission (NS).  If you, for genuine reasons, are unable to meet the submission date/time please note the following procedures, as per the university’s Fit to Sit Policy, found at: <http://www.rgu.ac.uk/about/academic-affairs/quality-assurance-and-regulations/academic-regulations-student-forms/academic-regulations-student-forms/>Please ensure to follow the instructions at the web address above.  Coursework Extension requests must be submitted 24 hours before the submission date and time.  Deferral Requests can be submitted in advance of the deadline and up to 5 working days after the deadline date/time.  Either request must be sent to the following email address:  StudentRequestSoC@rgu.ac.uk |

Please read the entire Coursework specification carefully before starting the Coursework. If any aspect of what you are being asked to do is not clear, seek advice and assistance from the Module Coordinator.

***Instructions to Candidates***

• **SAVE YOUR WORK REGULARLY.** We are unable to give credit for work you may have lost due to not saving it properly.

• The assessment is divided into 15 tasks, described in later sections of this paper. After completing the coding Tasks 1 to 14 you should submit your solution as a zipped copy of your NetBeans project to the **Resit** **Practical** **Exam** **Submission** **Dropbox** Link on the CMM024 Moodle page. Please remember that you are responsible for the submission of all of your work before the deadline.

• Details of the submission requirements on Task 15.

2

~~RGU: School of Computing CMM024 Coursework~~

3

~~RGU: School of Computing CMM024 Coursework~~

**Table** **of** **Contents**

**Contribution** **to** **the** **Module** **Grade...................................................................................................5** **Aim** **of** **the** **coursework** **for** **CMM024................................................................................................5** **Objectives** **of** **the** **coursework** **..........................................................................................................5** **Background........................................................................................................................................5** First phase........................................................................................................................................5 Second phase...................................................................................................................................6 **Overview** **of** **Tasks** **for** **the** **Coursework...........................................................................................7** *Task* *1*...............................................................................................................................................9

Create a new Netbeans Project.................................................................................................................. 9 Requirements for the enumerated typeenum................................................................................10 Requirements for the Abstract Item Class...............................................................................................11

*Task* *2*.............................................................................................................................................13

Requirements for the TextileItem............................................................................................................13 Requirements for the ObjectItem Class...................................................................................................14 Requirements for the IRecordsStorage Interface....................................................................................15 Requirements for RecordsStorage Class..................................................................................................16 Requirements for the code implementation of the RecordsStorage Class .............................................18 Demonstrate RecordsStorage by implementing a TestStorage class ......................................................20

Task 3.............................................................................................................................................21

Create a ObjectRecordPanel GUI to create Object Items........................................................................21 Create a TextileRecordPanel GUI to create Textile Items........................................................................22

Task 4.............................................................................................................................................25

Create a DisplayPanel GUI........................................................................................................................25 Create a MainFrame GUI..........................................................................................................................27

Uploading your work.......................................................................................................................30 **Indicative** **Grading............................................................................................................................31** **Grading** **Grid.....................................................................................................................................32** **Appendix...........................................................................................................................................36** Six enumerated types ....................................................................................................................36 Item Class.......................................................................................................................................37 TextileItem Class............................................................................................................................38 ObjectItem Class............................................................................................................................39 IRecordsStorage interface .............................................................................................................40 RecordsStorage Class...................................................................................................................41 ObjectRecordPanel Class..............................................................................................................42 TextileRecordPanel Class..............................................................................................................43 DisplayPanel Class........................................................................................................................44 MainFrame Class...........................................................................................................................45

4

~~RGU: School of Computing CMM024 Coursework~~

**Contribution** **to** **the** **Module** **Grade**

This coursework is worth 100% of your module grade.

**Aim** **of** **the** **coursework** **for** **CMM024**

To produce a hierarchy of Java classes fulfilling specified requirements. And then to implement a Graphical User Interface that interacts efficiently with that hierarchy of Java classes.

**Objectives** **of** **the** **coursework**

This coursework will evaluate the Learning Outcomes (1-5) found in the Module Descriptor.

• Analyse a set of system requirements and design, develop and test a solution which satisfies those requirements using Java.

• Interpret and describe class behaviour and inter-class relationships in an object-oriented design using provided UML diagrams.

• Demonstrate knowledge and understanding of object-oriented design concepts and inter-class behaviour using appropriate notation; including class composition, association, and inheritance.

• Apply the principles of class composition, association and inheritance, to construct hierarchies of new classes including components required for graphical interfaces.

• Implement interactive object-oriented programs that incorporate a graphical user interface and hierarchies of classes, using an event-handling model to handle user interaction with components.

**Background**

You have been asked to design software to store cloth and household items. They want you to create a prototype system in Java in two phases. In the first phase you will create a set of interacting Java classes for representing the

• Textile Item (clothing) and Other Item (household or tools etc.) • RecordsStorage

• And related basic interfaces and enums

• In the second phase will create a GUI interface.

**First** **phase**

During this stage, you will create and test the following Java classes.

• A class named **Item** that will encapsulate information relating to any items that need to be stored.

• A class named **TextileItem** that will encapsulate information relating specifically to textile items such as cloth etc.

• A class named **ObjectItem** that will encapsulate information relating specifically to any other household items that can be stored.

5

~~RGU: School of Computing CMM024 Coursework~~

• Six enumerated type called **Condition**, **ObjectType**, **PanelVisibility**, **TextileSize** and **TextileType** to indicate specific information related specifically to items to store

• An enumerated type called **RecordType** to indicate what type of record to store i.e. clothing items, any other household objects or all items.

• A class called **RecordsStorage** that will encapsulate the functionality of storing any items. o Add an item to the store.

o Delete an item from the store.

o Retrieve details of a stored items give their type i.e. clothing or household items. o Calculate the value given the required type

**Second** **phase**

In the second stage you will design and implement a GUI i.e. Panels and the main GUI frame that makes use of the hierarchy of interacting classes described above The GUI should incorporate functionality that allows the user to:

• Allow users to select a type of item whether it is a clothing or household item, to store and allow specific details to be entered for this items.

• Add this item to the store

• List all items in the store in a selective manner i.e. by specifying which type of item to display or all.

A detailed list of requirements is given in the remainder of this coursework specification.

6

~~RGU: School of Computing CMM024 Coursework~~

**Overview** **of** **Tasks** **for** **the** **Coursework**

There are total of **100** **marks** available in this coursework. The indictive breakdown of the marks is indicated below.

|  |  |  |
| --- | --- | --- |
| **Task** | **Task** **Description** | |
| Task 1  1 subgrade A-F | • Create a Netbeans Java project  Create a Netbeans Java Project and create a new project called MyStoreApp with a package named backend. In the backend package, create files as specified in the full description of Task 1. Create another package named gui. In the gui package, create the GUI files.  • Implement enumerated types  Implement enumerated types, Condition, ObjectType, PanelVisibility, RecordType, TextileSize, TextileType  • Implement the abstract class Item. |  |
| Task 2  2 subgrades A-F | • Extend the Itemclass to create a subclass called TextileItem.  • Extend the Itemclass to create a subclass called ObjectItem.  • Implement the interface IRecordsStorageto add and retrieve information to display  • Implement the RecordsStorageclass that implements *IRecordsStorage*  • Add the functionality to the RecordsStorageclass  To allows for the addition of items in the store and retrieve information for selected types of items  • Create a test class called TestBackend  To demonstrate the backend code is implemented properly. Sample output provided later. |
| Task 3  1 subgrade A-F | • Implement a GUI user interface ObjectRecordPanel(for data entry of ObjectItemdetails)  • Implement a GUI user interface TextileRecordPanel(for data entry of TextileItemdetails)  You will have to add buttons and combo boxes to satisfy the requirements |
| Task 4  1 subgrade A-F | • Implement a GUI user interface DisplayPanel(to display information stored in the storage)  • Implement a GUI user interface MainFrame(Main frame window) |
| When finished upload your work according to the instructions by the coursework deadline.  In addition to specific marks for functionality listed above, marks are available for good coding style, GUI layout and use of code comments. Code comments should include at a minimum a header comment in each class file describing its purpose; it’s author, and date of creation. | |

Each of the tasks 1 to 14 above is specified in detail in the remaining sections of this paper. **Note:** **All** **UML** **diagrams** **present** **in** **the** **tasks** **section** **have** **bigger** **version** **in** **the** **Appendix**

7

~~RGU: School of Computing CMM024 Coursework~~

8

~~RGU: School of Computing CMM024 Coursework~~

***Task*** ***1***

***Create a new Netbeans Project***

Work with Netbeans and create a new Java Application project via Java with Ant.

Use Project Name: MyStoreApp. Create a package named **backend** and in it you should create class files called:

• Condition.java, • Item.java,

• ObjectType.java • PanelVisible.java • RecordType.java

• RecordsStorage.java

• RecordsStorageInterface.java • TextileItem.java

• TextileSize.java • TextileType.java

• For GUI tasks you will need to add one or more files that extends JFrame and JPanel. • ObjectRecordPanel.java (JPanel)

• TextileRecordPanel.java (JPanel) • DisplayPanel.java (JPanel)

• MainFrame.java (JPanel)

9

~~RGU: School of Computing CMM024 Coursework~~

***Requirements for the enumerated type enum***

This task is about the implementation of the six enumerated types: • Condition

• ObjectType

• PanelVisibility • RecordType

• TextileSize • TextileType

Figure 1: The six enumerated types to implement

All the Java enuntypes should have the possible values as shown in the UML diagram above

10

~~RGU: School of Computing CMM024 Coursework~~

***Requirements for the Abstract Item Class***

**Task** **3** is to implement a **Item**class (in the class file Item.java). This class is intended to encapsulate key information relating to a base item. The **Item**class is specified by the following UML class diagram, and in the further comments below:

The Itemclass should have the following object-level properties

• A Stringobject variable called descriptionrepresenting a brief item description of the item.

• A Doubleobject variable object called estimatedValuerepresenting the perceived value of the item

• A Conditionobject variable called conditionrepresenting the condition of the item

• A RecordTypeobject variable called recordTyperepresenting the type of the item

The Itemclass should have one public constructor.

• The constructor that takes four input parameters of type String, Double, Condition and RecordType. This constructor should set the description**,** estimatedValue, the conditionand recordTypeproperties using the parameters.

The Itemclass should have public object-level methods with the standard behaviours indicated by the method name.

• getItemDescription • getPrice

• toString

The **Item**class should also have the object-level methods.

11

~~RGU: School of Computing CMM024 Coursework~~

• itemMainDetails • recordKind

Note that both itemMainDetails and recordKind are abstract methods taking no parameter and returning a String. These methods will be implemented by the concrete subclasses of Itemclass i.e. TextileItemand ObjectItem.

12

~~RGU: School of Computing CMM024 Coursework~~

***Task*** ***2***

***Requirements for the TextileItem***

**Task** **4** is to implement a TextileItemclass (in the class file TextileItem.java). This class is intended to encapsulate key information relating to a specific textile Item i.e. clothing. It inherits from the Itemclass. The TextileItemclass is specified by the following UML class diagram, and in the further comments below:

The TextileItemclass should have the following object-level properties

• A TextileTypeobject variable called textileTypedenoting if an item is a dress, coat or a t-shirt for example as an enumerated type.

• A Stringobject variable object called colourDescriptionrepresenting the a description of the overall colour of the piece of clothing

• A Stringobject variable called brandrepresenting the brand of the item

• A TextileSizeobject variable called textileSizerepresenting enumerated size of the clothing i.e. small, big for example.

The TextileItemclass should have one public constructor.

• The constructor that takes seven input parameters of type String, String, Double, String, TextileType, TextileSizeand Condition.

• This constructor should set the textileType, colourDescription,brand and textileSize of the TextileItem properties using the parameters.

• The other parameters i.e. description, estimatedValue, condition in addition to a RecordType that is passed as a static value, are used to initialise the super class (base class) Item

The TextileItemclass should have public object-level methods with the standard behaviours indicated by the method name.

• itemMainDetails • recordKind

• ToString

13

~~RGU: School of Computing CMM024 Coursework~~

Note that both itemMainDetails and recordKind are the concrete implementation for the abstract methods declared in the superclass (base class) Itemclass.

***Requirements for the ObjectItem Class***

**Task** **5** is to implement a ObjectItemclass (in the class file ObjectItem.java). This class is intended to encapsulate key information relating to a specific Item that is not clothing. It inherits from the Itemclass. The ObjectItemclass is specified by the following UML class diagram, and in the further comments below:

The ObjectItemclass should have the following object-level properties

• A ObjectTypeobject variable called objectTypedenoting if an item household, tool or any other item for example as an enumerated type.

The ObjectItemclass should have one public constructor.

• The constructor that takes four input parameters of type String, Double, Condition and ObjectType.

• This constructor should set the ObjectType properties using the parameters.

• The other parameters i.e. description, estimatedValue, condition in addition to a RecordType that is passed as a static value, are used to initialise the super class (base class) Item

The ObjectItemclass should have public object-level methods with the standard behaviours indicated by the method name.

• itemMainDetails • recordKind

• toString

Note that both itemMainDetails and recordKind are the concrete implementation for the abstract methods declared in the superclass (base class) Itemclass.

14

~~RGU: School of Computing CMM024 Coursework~~

***Requirements for the IRecordsStorage Interface***

**Task** **6** is to implement an IRecordsStorageinterface (in the class file IRecordsStorage .java). This interface is intended to specify the functionality, which will be visible to other classes. All the methods must be abstract. The IRecordsStorageinterface is specified by the following UML class diagram, and in the further comments below:

The IRecordsStorageclass should have two public object-level abstract methods. • Add

• retrieveRecordsToDisplay

The abstract Addmethod should have one parameter of type Item. The abstract retrieveRecordsToDisplaymethod should have one parameter of type RecordType.

15

~~RGU: School of Computing CMM024 Coursework~~

***Requirements for RecordsStorage Class***

**Task** **7** is to implement the skeleton of the RecordsStorageclass (in the class file RecordsStorage. java). This class is intended to specify the functionality for storing i.e. adding, and retrieving stored items for displaying in the GUI and implements the IRecordsStorageinterface. The RecordsStorageclass is specified by the following UML class diagram, and in the further comments below:

The RecordsStorageclass should have the following object-level properties

• A Mapobject variable called recordsStorageimplementing a collection of items • A Listobject variable called sortedRecordsrepresenting a list of sorted items.

• An Itemobject variable called aRecord, which will be used as temporary storage for an item during the process of sorting.

• A doubleobject variable called totalValue, which will hold the total value of the items retrieved for display.

The RecordsStorageclass should have one public constructor. • The constructor that takes no parameters.

• This constructor should create the recordsStoragemap that will hold the items.

The RecordsStorageclass should have public object-level methods with the standard behaviours indicated by the method name.

• find • add

• delete

• retreiveRecordsToDisplay • getsortedItems

16

~~RGU: School of Computing CMM024 Coursework~~

The findmethod should have a Stringparameter, which will hold an item description use to find an item in storage. Its return type is an Item.

The addmethod should have an Itemparameter to be stored in the storage. Its return type is a boolean.

The deletemethod should have an Stringparameter, which will hold an item description use to find the item to delete. Its return type is a boolean.

The retreiveRecordsToDisplaymethod should have a RecordTypeparameter, which provide the information to which type of record to select for display. Its return type is a String.

The getSortedItemsmethod should have a RecordTypeparameter, which provide the information to which type of record to sort for display. Its return type is a List.

17

~~RGU: School of Computing CMM024 Coursework~~

***Requirements for the code implementation of the RecordsStorage Class***

**Task** **8** is to implement the functionality of the RecordsStorageclass. Refer to UML screenshot in Task 7 for more details. Pseudo code shown below is provided help coding the required methods.

find(String description)

Get item whose description = parameter record Description

add (Item record)

If record is not null then

Find if record exists using the description of the Item record

If record is not in storage then

Store Item into storage

Return true

Else

Return false

delete (String description)

Find if record exists using the description on the Item

If record is null then

Return false

Else

Remove item form storage

Return true

getSortedItems (RecordType recordType)

Reset total value to zero

Create new list for sorted items

Set iterator to iterate the items stored in the storage

While there more items in storage

Get map entry for the record

Set aRecord to Item from that entry

If retrieved record recordType is equal to specified record type parameter

Increment total value by item retrieved value

Add item to sortedRecords

Else if retrieved record recordType is equal to ALL

Increment total value by item retrieved value

Add item to sortedRecords

End while loop

Return sorted records

retreiveRecordsToDisplay (RecordType recordType)

Set record list to getSortedItems (recordType)

Set iterator to iterate the items stored in the list

Reset output string

18

~~RGU: School of Computing CMM024 Coursework~~

While there are items available

Get item

Append output string with details for that item

End While loop

Append value stores for total value to output string

Return string output

19

~~RGU: School of Computing CMM024 Coursework~~

***Demonstrate RecordsStorage by implementing a TestStorage class***

**Task** **9** is to implement a TestStorageclass (in the class file TestStorage. java). This class is intended to demonstrate the RecordsStoragefunctionality. Create the class TestStorage class to perform the following tasks.

Make appropriate re-use of the retrieveRecordsToDisplay method of the RecordsStorage class.

• Create two closing items, one a coatand the second one a blouse. In terms of conditionone should be usedwhile the other new. Select any size, brand, colourand value(they must be different).

• Create two object items, one a householditem and the second one a tool. In terms of conditionone should be forScrapwhile the other used. Select any typeand value(they must be different).

• Add these four items to the storage

• Retrieve the information from the storage class for all textile items only • Retrieve the information from the storage class for all object items only • Retrieve the information from the storage class for all items.

• Make sure that the total value is accurate for the three tests

20

~~RGU: School of Computing CMM024 Coursework~~

**Task** **3**

***Create a ObjectRecordPanel GUI to create Object Items***

The requirements for Task 10 are to implement GUI components that allow the user to create Textile (clothing) Items to store**.** This class extending JPanel is called ObjectRecordPanel

**Note:** Your solution does not have to look identical, nor work in exactly the same manner as described below but should provide the same functionality as specified below. Make use of the screenshots, UML and information provided below. Marks allocated for each interaction is indicated in brackets.

You can use Netbeans Design tool but you also have the opportunity of coding it without the Netbeans designer.

This panel will be created and initialised during the MainFrameinitialisation sequence.

The ObjectRecordPanel class should have public object-level methods with the standard behaviours indicated by the method name.

• Clear • Get

• The initComponent method in the ULM is auto-generated by Netbeans when using the GUI designer.

21

~~RGU: School of Computing CMM024 Coursework~~

Create a GUI class, which should have combo-boxes, labels as shown in picture above

The Clearmethod has no parameters. Its function is to clear the input fields and to reset the combo box to their default selection. It will be invoked when user click on the Clearbutton in the main GUI frame window,

The Getmethod has no parameters and will be invoked when user click on the Add Itembutton in the main GUI frame window. The functionality of this method is to build a specialised ObjectItemobject from the information available in the panel and return it as a generic Item.

Note: Special care should be taken when parsing the Stringvalueto a double value to satisfy the ObjectItemparameter requirements.

Create the GUI Interface (3)

• Create a GUI interface extending JPanelby using GUI components that will fulfil the requirement of being able to create and ObjectItemusing coding or using Netbeans Designer.

Create the Clear method (1)

• Must clear the text fields and reset the combo boxes to their original value. Create The Get method (4)

• Retrieve all the value entered in the text fields and combo boxes selection. • Parse the text value to a double

• Create an ObjectItemand return it

***Create a TextileRecordPanel GUI to create Textile Items***

The requirements for Task 11 are to implement GUI components that allow the user to create Textile (clothing) Items to store**.** This class extending JPanel is called TextileRecordPanel

**Note:** Your solution does not have to look identical, nor work in exactly the same manner as described below but should provide the same functionality as specified below. Make use of the

22

~~RGU: School of Computing CMM024 Coursework~~

screenshots, UML and information provided below. Marks allocated for each interaction is indicated in brackets.

You can use Netbeans Design tool but you also have the opportunity of coding it without the Netbeans designer.

This panel will be created and initialised during the MainFrameinitialisation sequence.

The TextileRecordPanel class should have public object-level methods with the standard behaviours indicated by the method name.

• Clear • Get

• The initComponent method in Netbeans when using the GUI

the ULM is auto-generated by designer.

23

~~RGU: School of Computing CMM024 Coursework~~

Create a GUI class, which should have combo-boxes, labels as shown in picture above

The Clearmethod has no parameters. Its function is to clear the input fields and to reset the combo box to their default selection. It will be invoked when user click on the Clearbutton in the main GUI frame window,

The Getmethod has no parameters and will be invoked when user click on the Add Itembutton in the main GUI frame window. The functionality of this method is to build a specialised TextileItemobject from the information available in the panel and return it as a generic Item.

Note: Special care should be taken when parsing the Stringvalueto a double value to satisfy the TextileItemparameter requirements.

Create the GUI Interface (3)

• Create a GUI interface extending JPanelby using GUI components that will fulfil the requirement of being able to create and ObjectItemusing coding or using Netbeans Designer.

Create the Clear method (1)

• Must clear the text fields and reset the combo boxes to their original value. Create The Get method (4)

Retrieve all the value entered in the text fields and combo boxes selection. • Parse the text value to a double

• Create an ObjectItemand return it

24

~~RGU: School of Computing CMM024 Coursework~~

**Task** **4**

***Create a DisplayPanel GUI***

The requirements for Task 12 are to implement GUI components that allow the user to create Textile (clothing) Items to store**.** This class extending JPanel is called DisplayPanel

**Note:** Your solution does not have to look identical, nor work in exactly the same manner as described below but should provide the same functionality as specified below. Make use of the screenshots, UML and information provided below. Marks allocated for each interaction is indicated in brackets.

You can use Netbeans Design tool but you also have the opportunity of coding it without the Netbeans designer.

This panel will be created and initialised during the MainFrame initialisation sequence.

RecordType.TEXTILE

RecordType.OTHER

RecordType.ALL

25

~~RGU: School of Computing CMM024 Coursework~~

The DisplayPanel class should have one public constructor.

• The constructor that takes one input parameters of type IRecordsStorage.

• This constructor should set the storageInterface properties using the parameters.

The DisplayPanel class does not have public object-level. However it should have listeners to check for action performed. If you are using the Netbeans GUI designer, this is straightforward. If you are however coding the class with out it, you will have to implement that functionality for each of the RadioButtoncomponents. In addition a ButtonGroup should be used to ensure that that only one of the radio button is selected at one time.

• You should have a method for the TEXTILE radio button. • You should have a method for the OTHER radio button.

• You should have a method for the ALL radio button.

For each of these methods, you should use the IREcordsStorageinterface passed as a parameter in the constructor i.e. storageInterface, retrieve the information required as a string and use this information to update the JTextAreaof the GUI called reviewArea.

Create a GUI class, which should have combo-boxes, labels as shown in picture above Create the GUI Interface (3)

• Create a GUI interface extending JPanelby using GUI components that will fulfil the requirements using coding or using Netbeans Designer.

Create the RadioButtonaction listener methods (5)

• Three methods are needed, each will be using the RecordsStorageinterface to query and retrieve the relevant information.

26

~~RGU: School of Computing CMM024 Coursework~~

***Create a MainFrame GUI***

The requirements for Task 12 are to implement GUI components that allow the user to create Textile (clothing) Items to store**.** This class extending JPanel is called MainFrame

**Note:** Your solution does not have to look identical, nor work in exactly the same manner as described below but should provide the same functionality as specified below. Make use of the screenshots, UML and information provided below. Marks allocated for each interaction is indicated in brackets.

You can use Netbeans Design tool but you also have the opportunity of coding it without the Netbeans designer.

This panel will be created and initialised during the MainFrame initialisation sequence.

The MainFrame class should have the following object-level properties

• A TextileRecordPanelobject variable called textileRecordPanelas an object instance of TextileRecordPanel

• A ObjectRecordPanelobject variable called objectRecordPanelas an object instance of ObjectRecordPanel.

• A DisplayPanelobject variable called dislayPanelas an object instance of DisplayPanel

• Four JPanelused as container for JButtonGUI components etc..

• A RecordsStorageobject variable called recordsStorageas an object instance of RecordsStorage

• Two JButtomobject variable called respectively AddButton and clearButton, which are going to be used to invoke methods of some of the panels.

The MainFrame class should have one public constructor. • The constructor takes no input parameters.

• The GUI is constructed in the constructor and the RecordsStorageinstance is also created inside it.

Create the GUI Interface (5)

• Create the storage object instance

• Set layout of the MainFrame to Borderlayout

27

~~RGU: School of Computing CMM024 Coursework~~

• Create MainFrame panels i.e. northPanel, southPanel, westPanel, eastPanel, southPanel • Create the specialised panels i.e. ObjectRecordPanel, TextileRecordPanel, displayPanel • Create five JButton buttons i.e. textileBtn, objectBtn, reviewBtn, and addBtn, clearBtn

• Set size for panels for each BorderLayout areas

• Set layout for MainFrame panels – westPanel: GridLayout, northPanel: Specialised FlowLayout

• Add buttons and the label to the relevant MainFrame panels - textileBtn, objectBtn, reviewBtn to westPanel and addBtn, clearBtn to southPanel

• Add the northPanel, southPanel, westPanel and eastPanel to the MainFrame • Add the displayPanel to the center area and set the visibility to this panel

• Add action listeners for all buttons

• Set default closing behaviour to exit on close

Fill the textileBtn, objectBtnand reviewBtnaction listener methods (4)

• The functionality relates to swap panels in the central area of the BorderLayout. Pseudo code examples relating to the buttons is show below.

displayObject method

Hide textileRecordPanel

Hide displayPanel

Show ObjectRecordPanel

Show add button

Show clear button

Add objectRecordPanel to Mainframe

Set panelVisible enumerated type to OBJECT

End of method

displayTextile method

Show ObjectRecordPanel

Hide displayPanel

Hide ObjectRecordPanel

Show add button

28

~~RGU: School of Computing CMM024 Coursework~~

Show clear button

Add textileRecordPanel to Mainframe

Set panelVisible enumerated type to TEXTILE

End of method

displayReview method

Hide ObjectRecordPanel

Show displayPanel

Hide ObjectRecordPanel

Hide add button

Hide clear button

Add DisplayPanel to Mainframe

Set panelVisible enumerated type to REVIEW

End of method

Implement the addItemmethod (2)

• The functionality relies on the visibility to determine which panel is active in the central areas of the BorderLayout.

• It invoke the getmethod for the active panel to retrieve an Item and invoke the add method of recordsStorageclass to store that Item

Pseudo code example. AddItem method

Switch panelVisible

Case TEXTILE

Get Item record from textileRecordPanel

Invoke recordsStorage add method to store item

Clear entries

Case OBJECT

Get Item record from ObjectRecordPanel

Invoke recordsStorage add method to store item

Clear entries

End method

Implement the Clearmethod (2)

• The functionality relies on the visibility to determine which panel is active in the central areas of the BorderLayout.

• It invoke the clearmethod for the active panel to clear fields to their default values Pseudo code example.

AddItem method Switch panelVisible

Case TEXTILE

Invoke textileRecordPanel clear method

Case OBJECT

Invoke ObjectRecordPaneld clear method

End method

29

~~RGU: School of Computing CMM024 Coursework~~

**Uploading** **your** **work**

1. Once you are satisfied that your program fulfils the requirements SAVE your *Netbeans* Java project and then close Netbeans.

2. Navigate to whatever folder you saved the Netbeans project in.

3. Select the folder of your Java project. If you have made multiple copies of this, then ensure that you have the correct one (i.e. the one you intend to be your coursework submission) and create a *ZIPPED* copy of the folder.

4. Open the **Resit** **Submission** **Dropbox** link on the *CMM024* Moodle page. 5. Upload the *ZIPPED* copy of your *Netbeans* project.

**6.** **[END** **OF** **COURSEWORK** **SPECIFICATION]**

30

~~RGU: School of Computing CMM024 Coursework~~

**Indicative** **Grading**

• Marks will be awarded for your coding within each of the tasks specified in the *Overview* *of* *Tasks* *for* *the* *Coursework*

• If you attempt a question but there are errors preventing it compiling, then comment out the code with problems so that the rest of your system will compile. You may still gain marks for the progress made in the commented-out code.

• Partial marks will be awarded in the event that a task is partially complete, so do attempt and submit attempts to all tasks even if they are not fully functional.

• The total or marks will be used as follows to assign the module grade.

31

~~RGU: School of Computing CMM024 Coursework~~

**Grading** **Grid**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Grading | A | B | C | D | E | F |
| Task 1 | You have created a Netbeans project meets that meets the requirements and is as expected  Your coding of all enumerated types is excellent and is as expected as well  Your coding of the Item class was properly codes and had has no errors  Overall there is no detected issues | You have created a Netbeans project meets that meets the requirements and is as expected  You effort at coding all the enumerated types is good and is mostly as expected. Some constants you used in the code may not be exactly as they should be. However overall they convey the idea  The Item class you coded was overall properly done but may have small flows | You have created a Netbeans project meets that meets the requirements and is approximately as expected  You effort at coding of the enumerated types is overall adequate. However, some constants you used in the code maybe named differently or one or two enums are not named properly or one is missing  The Item class you coded was overall adequately programmed but has small flows or maybe one key defect | You have created a Netbeans project meets that meets some the requirements but is functional  You effort at coding of the enumerated types has issues. However, some constants you used in the code maybe named differently or one or two enums are not named properly or one is missing  The Item class you coded contains errors some of which are key errors | You have created a Netbeans project that does not meet the requirements  You effort at coding of the enumerated types has key issues or many enums are missing  The Item class you coded is not complete and lacks functionality due to key mistake or omissions | Non-Submission or Non-relevant content |
| Task 2 | Your coding of the classes is excellent and is meeting all the requirement as far as the functionality is concerned but there may have a few very small flows in the coding that are not important in terms of the overall functionality | Your coding of the classes is good and meets most to the requirements. By this I mean that some small errors or flows in the coding are present but will not affect the functionality of the backend to great extend | The coding of the backend class is overall mostly adequate but there are issues and some key issues. This means that in terms of the backend functionality this might lead to a defective functionality when it comes to integrate it with the GUI etc., or a some lack of functionality when it comes to the storage class | You have attempted to code the backend some classes but there are major issues with the code. This means that some of the classes have little code to make them functional or some class are missing  This means that in terms of integration with the GUI there will be major key issues. In addition these key issues will impair the storage of | A small attempt at programming the classes was made. Either many classes are largely incomplete or a number of them are missing.  This means that the integration of the backend with the GUI will be impossible or that the functionality of the storage class if it is coded will not be right at all. | Non-Submission or Non-relevant content |

32

~~RGU: School of Computing CMM024 Coursework~~

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | Items in the storage |  |  |
| Task 3 | The programming of the two GUI panels you have done is excellent. This means that the GUI, which maybe different from the provided screenshots contains all the required components and that the two extra object methods i.e. Clear and Get are coded properly with the expected functionality | The design of the two GUI panels is good and contains all the required components to make it functional. The GUI might look different from the proposed layout but it is functional.  The coding of the Clear and Get methods are overall coded properly but may have some little flows. An Item object is created. | The design of the two GUI panels is adequate and contains most of the required components. The GUI might look different from the proposed layout but it is somewhat close to what is expected in terms of functionality.  The coding of the Clear and Get methods has been attempted but there are flaws; some of them being key defects  An attempt at creating the Item object has been made | You attempted at creating the design of the two GUI panels. However there are major issues with the layout which will lead to the panels not being very functional  This means that the GUI panel might look different from the proposed layout but it is lacking components to make it functional  Some coding of the Clear and Get methods has been attempted but there are major flows with the code or no attempt is creating the Item in the Get methods was made. | A small attempt at creating was made. However the effort did not produce functional panels with maybe many components missing or they have major other issues.  The coding of the Clear and Get methods may not have been attempted but the code is far from what is expected | Non-Submission or Non-relevant content |
| Task 4 | The design and programming of the Display panel is excellent. The mainframe design is also well done. They may look different from the proposed layout though  All the functionality in terms of interaction is present  Both classes have almost all the functionality that is required but may have some little flows | The design and programming of the Display panel and the mainframe design is close to what was expected. They may look different from the proposed layout though  Almost all the functionality in terms of interaction is present  The items stored in the display screen are shown but they maybe has defects for selection types etc., or | A good attempt at designing and programming the Display panel and the mainframe design was made  There is however some key issues both in terms of interaction and functionality i.e. some panels may not show, or some commands are not succeeding in what they are supposed to do. Etc.  E.G. he items stored in the display screen | Some attempt at designing and programming the Display panel and the mainframe design was made. Some GUI components have been laid out but some are missing etc.  The issue is that there are many issues with both the coding and design of of the panel and frame. These are key defects which affects both interaction and of the classes. I.e. there maybe lack of possible interaction | You have made a small attempt at designing the panel and frame but there are many key issues with the design and coding.  Very little is there in terms of visual design and the backend code of these classes is minimal i.e. very little functionality and interaction is present | Non-Submission or Non-relevant content |

33

~~RGU: School of Computing CMM024 Coursework~~

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | in terms of interaction  The Items stored in the storage class are displayed properly and the GUI interfaces is also as expected. | expected output  In terms of programming the functionality the code may contains some small errors or omissions but it is almost there. | might not show in the display panel at all. Defects like lack of interaction when selecting the type of items to view is detected etc.  The programming the functionality the code may contains some major errors or there are omissions | or the code in those classes are not properly coded, which impairs the overall functionality greatly  For instance nothing is displayed in the display panel and click on buttons or the radio buttons does not produce the expected results etc. |  |  |

34

~~RGU: School of Computing CMM024 Coursework~~

35

~~RGU: School of Computing CMM024 Coursework~~

**Appendix**

**Six** **enumerated** **types**

36

~~RGU: School of Computing CMM024 Coursework~~

**Item** **Class**

37

~~RGU: School of Computing CMM024 Coursework~~

**TextileItem** **Class**

38

~~RGU: School of Computing CMM024 Coursework~~

**ObjectItem** **Class**

39

~~RGU: School of Computing CMM024 Coursework~~

**IRecordsStorage** **interface**

40

~~RGU: School of Computing CMM024 Coursework~~

**RecordsStorage** **Class**

41

~~RGU: School of Computing CMM024 Coursework~~

**ObjectRecordPanel** **Class**

42

~~RGU: School of Computing CMM024 Coursework~~

**TextileRecordPanel** **Class**

43

~~RGU: School of Computing CMM024 Coursework~~

**DisplayPanel** **Class**

44

~~RGU: School of Computing CMM024 Coursework~~

**MainFrame** **Class**

45