COIT20256 Assessment item 1— Assignment 1

Due date: Thursday of Week 6 (23rd April 2020) 11.45 pm AEST

ASSESSMENT

Weighting: 20%

Length: NA 1

1. Objectives

The purpose of this assessment item is to assess your skills attributable to the following learning outcomes and your achievement of the expected graduate attributes of advanced level communication, knowledge, cognitive, technical, and creative skills, and self-management.

- Design and implement appropriate data structures for application development
- Analyse, develop and implement software solutions with the focus on data structures and algorithms
- Apply classes, inheritance, polymorphism, and exception handling

2. Assessment task

Your task in this assessment is to analyse the given problem, model, and design the required data structures using UML class diagrams. You will be implementing the software solution, applying efficient algorithms, inheritance, polymorphism, and exception handling. The topics required for this assessment task are from Weeks 1-5. Your Java Application should have an interactive Graphical User Interface (GUI) using JavaFX GUI components. You should also write a report, as specified in this document, demonstrating your conceptual knowledge. You will be required to use the topics learnt in the pre-requisite unit *Introduction To Programming*.

2.1 Problem

The Australian Bureau of Statistics (ABS) defines a homeless as a person
"who does not have suitable accommodation and whose current living arrangement:
is in a dwelling that is inadequate, has no tenure
is where the initial tenure is short and not extendable
does not allow the person to have control of, and access to space for social relations".

On Census night in 2016, more than 116,000 people were estimated to be homeless in Australia—58% were male, 21% were aged 25–34 and 20% identified as Aboriginal and Torres Strait Islander Australians (ABS 2018). As of June 2016, there were 194,592 applicants waiting for social housing across the country's social housing programs (Patterson, 2017). Specialist Homelessness Services (SHS) are charity and government funded organisations working to support homeless people.

End Homelessness (EH) is a newly registered SHS who is determined to tackle the problem of homelessness in innovative ways by using information and communication technology to collect data, increase advocacy and if needed employ crowd funding to raise money. The very

basic requirement for addressing the issue is to have accurate information on each homeless person. Therefore, the directors have decided to have a Homeless Information System (HIS) which is reliable, flexible, and expandable.

You are invited to design and develop an initial prototype for the HIS. The prototype needs to have only limited functionality as described below. As a first step the HIS will be designed to collect data about single persons living in privately rented dwelling and having a very low income. If the weekly income is very low, any changes to their employment or other income source or any changes to their needs can affect their ability to pay rent, and increasing their risk of becoming homeless. The initial data collection will be targeted on people aged 50 or above. The support for the full development of the system depends on the quality of the prototype.

You will be developing a Java application with a GUI.

The users accessing the HIS should able to:

1. Add new location
2. Update the statistics of people at risk of becoming homeless by choosing:
    - appropriate location
    - gender of the person to be added
    - correct age group and
    - income category
3. Display statistical reports to view:
    - all males at the risk of homelessness
    - all people in a chosen location at the risk of homelessness
    - all females at the risk of homelessness
    - all at the risk of homelessness

Note: You will be given a data file containing sample data of single persons belonging to a number of locations in Western Australia. You are free to add any new location anywhere in Australia

2.2 Design Guidelines

You can use the following guidelines in your modelling and GUI design.

Data Structures

You may follow the class design given below. You should read the specification carefully and design the classes using UML class diagrams and include the attributes with correct data types and all the methods, clearly indicating public or private. Clarify your doubts during the design stage so that your implementation can be less error prone and faster to complete.

All classes should have a default constructor, parameterised constructor, copy constructor, accessor mutator (get, set) methods, and properly written toString() methods. Include other methods as needed. This doesn't apply to the JavaFX classes for the GUI.

i) Location class

The Australian Bureau of Statistics uses the geographical mesh map which divides the total geographic area into Mesh Blocks which is the smallest in size in terms of number of people living in the area. Statistical Areas Level 3 (SA3s) generally have populations between 30,000 and 130,000 persons and these are often the functional areas of regional towns and cities with a population in excess of 20,000 (ABS, 2018). The statistical data will be collected and studied from various SA3s.

This class is to read the given file and store the locations in an ArrayList. A Location has a *SA3code* which is a five digit number. Each state has a range for the SA3code and for Western Australia all SA3codes start with digit '5'. The Location has a *name* which can be more than one word separated by '- 'and a space before ' ' and space ' ' after (eg: "Augusta - Margaret River – Busselton". The name should be validated so that it contains only alphabets other than the '-' and space characters.

ii) Person class

This is a generic class and has an age category, and a gender category. This class will remain abstract as no objects will be required of type Person.

iii) SA3TenantCategory class

This class extends the Persons class by including:

Location
income category
income source.

Override the inherited methods as required. Ensure that the source code is re-used and not repeated unnecessarily leading to redundant code.

The income sources included in the data file are: "employed" and "other".

iv) RiskyPersons class

This class is for recording and displaying the statistics of people at the risk of homelessness.

This class will have:

SA3Category object,
Number of Persons in a Cateogry

Hints and tips: Refer to Week 3 Lecture and tutorial solutions, and the Textbook chapter on Polymorphism for the correct implementation of polymorphic behaviour.

v) DataFile class

This class is for implementing the file operations to open, read the values from the given '*futureDemand.csv*' file, store the values in an ArrayList<RiskyPersons>(). This file will be available from the Unit website.

Use specific file handling exceptions for various checked exceptions related to file operations.

vi) enum GENDER

Create an enum type GENDER to store male, female gender values as these remain constants.

vii) enum WEEKLY_INCOME

The actual weekly income is not recorded. The risk of homelessness is identified by the range of income. The interested ranges are:

- (-2000 -0) - NilIncome
- $1-$399 – below$400
- $400-$599 – below $600
- $600-$999 - below$1000

Create the enum type to enable use of these categories as constants.

viii) enum AGE

The actual age of the people recorded is not kept, but the age category within which they belong to are kept. The categories used are:

1. 50-54
2. 55-59
3. 60-64
4. 65 or over

ix) JavaFX classes

The standard classes required include the main class to create the Stage and launch, the controller to initialize GUI components, handle events, and validate user entries, the .fxml file for the layout design of GUI components, and the .css file for formatting.

You can have:

1. *HomelessInfoMain* class that has the main method and starts the application
2. *HomelessInfoController* class for event handling and data collection.
3. HomelessInfo.fxml for the .fxml

Graphical User Interface

The GUI should have the necessary components to enable the user to execute all the functions as provided in Section 2.1 above. You may use the guidelines provided below for your GUI design and implementation. Variations to the provided guidelines are acceptable if it meets the user requirements. Follow the User Interface design guidelines learnt in Week 4 and design the GUI to meet chosen aspects of an easy to use user interface that provides informative error messages, and clear instructions.

You can have a GUI with multiple tabs for taking the user through the step by step process of entering data about a new location, add new RiskyPerson/s belonging to an SA3TenanatCategory to the records, and finally displaying reports choosing one of the options. The user should be supported with the GUI in each step as given below.

i) Location

Use this Tab to enable the user to enter the SA3code, and name of a Location and add the Location. You can choose appropriate controls (Labels, TextFields, and a Button) for these.

Display error message if the user hasn't entered the Location correctly.

ii) Tenant

Use this tab to enable the user to enter the age, income, choose gender, SA3code, and income source. The user should then be able to increment number of RiskyPersons in the corresponding SA3TenantCategory in the existing ArrayList<>().

You can use Labels, TextFields, RadioButtons, ComboBox or ListView, and a Button to add the data.

iii) Reports

Use this Tab to enable the user to choose one of the available options and display the data. You can use a TextArea to display the report details. A full GUI design diagram is not given to encourage you to design user friendly interface.

You can switch between Tabs by using a *Continue* button. Once the *Continue* button is pressed and the Tab is changed the entries in the previous fields should be cleared.

3. Coding

Include necessary accessor, mutator methods, constructors, and toString() method for each class. Also, follow good coding practices, using meaningful names, camel case notation for naming, constants as necessary, and include meaningful comments. You can use NetBeans to develop your application. *Follow the coding standards given in the Unit website.*

3.1Guidance to the level of assistance you can use.

You are expected to understand a number of concepts and apply those concepts to design and build a software solution. At this level you can use the provided materials, online resources for further reading and take assistance from your classmates or teammates to develop deeper understanding of the concepts. You can also sort help to debug the implemented program. But you should implement and test your program on your own.

Table 1 Allowed Assistance

S.No	Source	Activity
1	Unit Textbook, Unit notes and examples	Understanding concepts, design
2	Instructors	Understanding concepts, design, debugging
4	Classmates, Online resources	Understanding concepts
5	Everyone else	Understanding concepts
6	**No help acceptable**	**Implementing code**

4. Report

You should submit a report containing the following details.

1. UML class diagrams for the classes
   Note: UML class diagrams generated using a software tool after completing the coding will not be accepted.
2. Test plan showing input data, expected results, and actual results. Show testing of erroneous entries also.
3. Write clearly which aspects of the User Interface Design principles you have followed and how did you implement those.

5. Assignment Submission

You should submit the following source code files and word document using the Moodle online submission system. (Note: the file names/class names could be changed to meaningful names). *Please do not zip the whole project folder and submit it.*

- Location.java – Source code for the Location class
- Person.java– Source code for the Person class
- SA3Tenant.java - Source code for the SA3Tenant class
- RiskyPersons.java - Source code for the RiskyPersons class
- DataFile.java – Source code for DataFile class
- AGE.java, GENDER.java, and WEEKLY_INCOME.java for the Java enums.
- .java files for the main and controller classes
- .fxml and the .css files
- Report.doc – Your word document containing the report.

Assessment Item 1 Marking criteria

S.No	**Total Marks - 20**	Marks Allocated	Marks Scored
1	Graphical User Interface Presentation: User Friendly Design	2	
2	Design and use of appropriate data structures	1	
3	Implementing classes and using objects, methods, aggregation, and polymorphism	3	
4	Efficient program design: absence of redundant or repetitive code, correctly designed loops, blocks, and methods	2	
5	Use of exception handling	1	
6	Enables User to correctly enter Location	1	
7	Enables User to add a person/s to an SA3TenantCategory	2	
8	Enables User to choose any option and display each one of the reports	3	
9	Good coding practices including comments, indentation, use of constants as required, use of meaningful names, and camelCase notation for names	2	
10	Well-presented report with student details, UML class diagram, demonstration of testing all functions of the program, and description of the GUI principles implemented	3	
	Penalties		
11	Late Penalty (-1 mark: 5% of total allocated marks per calendar day)		
12	Source code found entirely different from the styles followed in the Unit or containing constructs outside of this Unit will be penalised (-1 to -10 marks depending on the amount of source code)		
13	Plagiarism (penalty as per the plagiarism policy)		
14	**Total**	**20**	

References

1. Patterson, D., 2017. A national homelessness strategy: Why we need it. *Parity*, *30*(6), p.10.
2. ABS 2018. Census of Population and Housing: Estimating homelessness, 2016. ABS cat. no. 2049.0. Canberra: ABS. available from: https://www.abs.gov.au/ausstats/abs@.nsf/mf/2049.0
3. ABS 2018 Australian Statistical Geography Standard (ASGS). Available from: https://www.abs.gov.au/websitedbs/D3310114.nsf/home/Australian+Statistical+Geography+Standard+(ASGS)