Student Written Assessment (Practical Assignment) -DRAFT

Business Unit/Work Group	IT Studies		
Qualification Code	National Code: ICT50715	**Qualification Title**	Diploma of Software Development
Unit Code/s	ICTPRG532	**Unit Title/s**	ICTPRG532 -Apply advanced object-oriented language skills
Assessment Task Title	Assessment 2 - Algorithms and Data Structures Assignment		
Student Name	Submit your solution via your LEARN account	**Student SIS ID**	Submit your solution via your LEARN account
Assessor Name	You have been added to a LEARN group which defines your assessor. This is normally your Course Registration Number (CRN) lecturer.	**Date**	2020 Semester 2

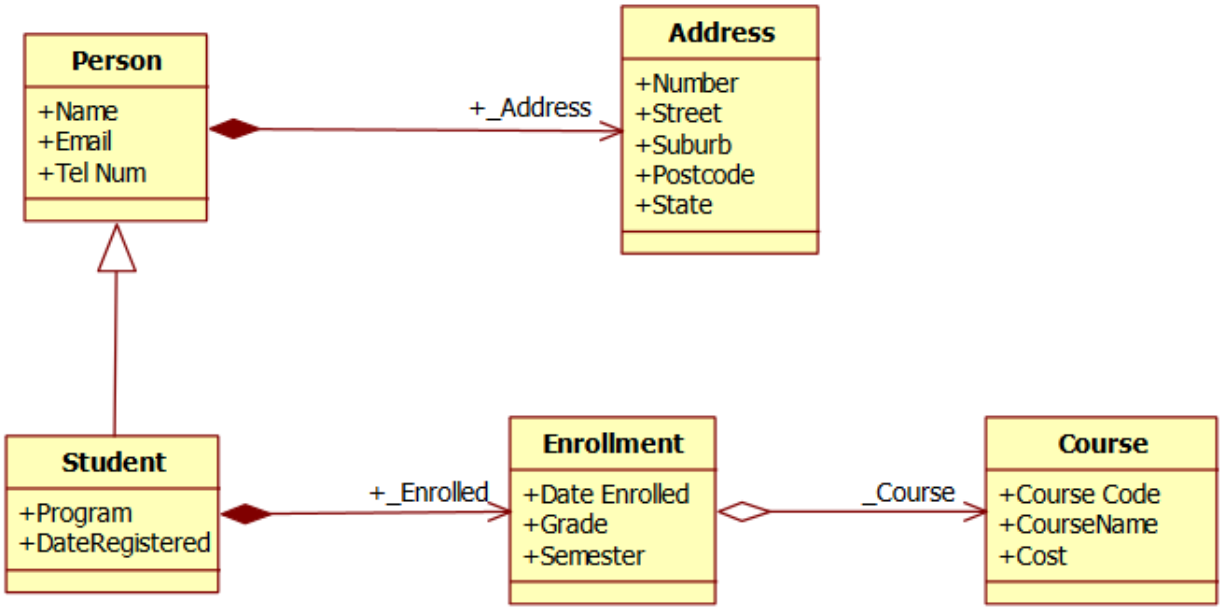
Student Guide for Written Assessment	
Overview of Assessment	This is a practical hands-on assessment that will require you to develop a set of algorithms and Data Structures based on a set of specifications
Task/s to be assessed	You will be assessed on the successful completion of all sections of the Main requirements as set out in this assessment.
Time allowed	The assessment must be completed and uploaded by the end of the first term of the semester
Location	You could complete this assessment in class or at another suitable approved site
Decision making rules	To receive a satisfactory outcome for this assessment you must complete all the steps as outlined in the Basic Requirements section
Assessment conditions	• You may be asked to explain your solution to the instructor in class • You are not allowed copy or use code that is not yours without acknowledgement. • You have till the end of the term from the time of release of this assessment to complete and upload your solution on the subject LEARN site • Ensure you follow the provided IT Works organizational guidelines for developing maintainable code, and adhere to the provided IT Works Java (SE) coding standards.
Resources required	To complete this assessment, you will need to use NetBeans 8.x with Glassfish 4.x and access to the subject Learn resources. NetBeans 8.x based machines are provided during this assessment. You can use a Mac if you prefer but these are not provided.
Results/Re-assessment	You will be provided with feedback for the assessment and be given the opportunity to resubmit any required corrections only once.
Submission Instructions	• Create a folder called <your Name>_ 5JAW_Assignment on the desktop and add your completed NetBeans project inside this folder. • Add your name to the comments at the beginning of each file.

Main Requirements
Specification for Library of Algorithms and Data Structures

IT Works has won a contract to develop a Student Enrolment System and has hired you in the role of an analyst developer to create a library of Data Structures and algorithms based on the following Object-Oriented Model. The components will be included in a Java Class library and exposed by a set of relevant APIs.

In keeping with the organization standard development tools, you are required to use Netbeans to develop the required Java components.

Create a Java project in NetBeans to Plan and Implement the following requirement specifications



Part A (Equality/Hashing)

You are required to develop the above classes in Java and implement a suitable equals methods to test equality of the Student, Course and Enrolment classes. What instance variables will you select to test for equality for each of the required classes and justify your choice as comments in your code.

The specification recommends that you override the equals ( ) and getHashCode( ) methods of each of the required classes

Test your classes by referring to the Test Documentation section below.

Part B (Comparators)

The Student, Course and Enrollment classes will be used in different Collections and the ordering of such elements would be an important requirement for this application.

The application architect has recommended the implementation of the Java Comparator Interface to enable the proper ordering of the required classes. The Comparators for each of the classes should ignore the properties of the classes that are case sensitive, such as String types. The idea here is that, when ordering, for example a List of Student /Course/Enrollment objects in either ascending or descending order, the Comparator will ensure that the ordering makes sense.

Test your Comparators by adding them to a List and Set and test if your ordering works correctly and refer to the Test Documentation section below.

Part C (Searching)

The led architect has requested that you investigate the various techniques to search for objects in the Java Collections. This requirement arises from the need to read data from tables in a RDBMS and cache them in suitable Collection repositories for processing.

Document and at least different type of searching algorithms and discuss the pros and cons of using each of them to search for data in a Java Collection.

Select at least two of the three searching techniques and implement an algorithm in Java. The architect requires these algorithms be available as a set of APIs in a class library.

Test your search algorithms work for a List of Students, Courses and Enrolment's. Please refer to the Test Documentation section below for more details

Part D (Sorting)

The architect has requested your opinion of how best to sort a list of Student/Course/Enrollment object from a range of sorting algorithms available. He requests that you select three such sorting algorithms implement them in Java and demonstrate their use in the sorting on an Array /List of Students, Courses and Enrolments in ascending order. Include these algorithms in a Java class library to be available as a set of APIs.

NOTE – You must write your own algorithms and NOT use the built-in sort methods of the Java Collections classes.

Test your sorting algorithms on a List of Students, Courses and Enrolment's. Please refer to the Test Documentation section below for more details

Part E (Linked Lists)

The architect has decided not to use the built in Linked List collections in the Java SE library and tasked you to create a set of custom built Linked List (both Single, and Doubly) that will be part of the custom Java library of Components.

As part of the requirements, you are tasked in creating a customized set of Generic Single and Doubly linked lists that can be used by any of the classes defined in the above UML model and others that might be added in the future. The customized lists will be available as a set of APIs. The list should implement methods to add objects to the head, tail or anywhere in the list.

Other methods include a find, remove (from head, tail or anywhere) object from the list and traverse the list forward or reverse (if Doubly Linked List) and return its values.

All the classes and methods must be properly documented by including code comments appropriately.

Demonstrate that your Data Structures work by implementing the various methods across several type of classes.

Please refer to the Test Documentation section below for more details

Part F (Binary Trees)

The architect has also suggested that the implementation of a custom Binary Tree (BTree) data structure would be a better solution than using the available BTree Collection components of the Java SE library.

The customized BTree once implemented would be part of the Class library APIs that could be used across different projects that IT Works will be involved. The various methods of the customized BTree would form part of APIs available from the library.

The customized BTree should have the capability to add, remove and find elements in the tree.

When searching for elements in this data structure, the traversal algorithms should be capable of Pre-Order, In-Order and Post-Order traversals.

Testing and Test Documentation

As part of the IT Works quality standards you will need to have a test plan and all test cases and testing must be documented using a desk check as evidence that you have tested all requirements. The recommended strategy for unit testing your beans is to use nested inner classes to test the implementation of classes specified in the above UML model. You must use JUnit to test requirements.

User Documentation

Use the Java Doc capabilities in NetBeans, create set of GUI web-based API help pages to document the Data Structures and Algorithms developed according to the above specifications. . Modify the comments in all code to maintain the updated documentation.

Logic Code errors should be documented with evidence of code debugging using the features of the IDE including the use of break-points to step through code and examining of variable content. The idea here is to document the cause of the error and how it was fixed and results of re-testing.

Version Control

Use GitHub as a repository to store and retrieve your Java Components, with the implementation of secure check-in and check-out procedures and version control. Document structural, access details and procedures to retrieve APIs.

Documentation/Code- Review

On completion of the requirements according to the above Plan - you will be required to demonstrate the working APIs to the project architect and explain the workings of the components.

This will take the form a code review where you will require to use capabilities of the IDE to insert breakpoints in parts of the code and step through solution to demonstrate and explain code features.

The architect will sign off on the acceptance/agreement (or not) of the implemented components which will be part of the customized library of Java Components

C#.NET Lib Components

The solutions architect has requested a similar API of components to be implemented and built in C#.NET. The same requirement specifications will apply.

Compliance with IT Works Standards

All submitted code should adhere to the ITWorks coding standards

1. Java SE coding standards as documented at <Reference here>
2. C#.NET coding standards as documented at <Reference Here>
3. IT Works OO coding standards as documented at <Reference here>

Additional Requirements

TBA

S