Assessment Student Instructions

Assessment Title	Assignment – Data Structures and Algorithms

Competency Details

Unit code/s and title/s	ICTPRG547 Apply advanced programming skills in another language
Qualification code/s and title/s	ICT50220 Diploma of Information Technology
Business unit/Work group	Business and Arts/ IT Studies

Instructions

Method/s of assessment	Product (API creation) Oral communication
Overview of assessment	This is a practical hands-on assignment that will require you to interpret a requirements specification- to design, develop, build, test and document a library of data structures API's that have the capabilities to store, retrieve, sort and search for generic data types. The assessment has nine parts
Task/s to be assessed	You will be assessed on the successful completion Parts 1 – 9 of a requirements specifications as follows: 1 **–** Dynamic Data Structure Specifications 2 – Hashing Specifications 3 – Sorting Specification 4 – Searching Specifications 5 – Exception Handling Specifications 6 – Tools | Debugging | Coding Specifications 7- Version Control Specifications 8 – Testing 9 – Further Research You will be assessed on your communication and negotiating skills.
Time allowed	Refer to your schedule for submission dates
Location of assessment	Assessment can be completed anywhere with access to the resources required. (see Resources Required section below)
Decision making rules	To receive a satisfactory outcome for this assessment you must complete all parts according to the assessment specification
Assessment conditions	This assessment must be undertaken in a workplace or simulated environment where the conditions are typical of those in a working environment in this industry. This is unsupervised assessment, and you may access any required resources. This is not group work and must be completed as an individual
Resources required	To complete this assessment, you will require the following:

	• Access to Learn with Internet access • Learn resources • Word processing software such as Microsoft Word. • Visual Studio 2019 with SQL Server Express • Windows based machines with VS 2019 and SQL Server Express installed are provided in your practical classes. You can use a Mac if you prefer but these are not provided in the classrooms. • C#.NET Coding Standards • IT Works Organizational Policies and Legislative Requirements • Project Documentation Template
Result notification and reassessment information	You will be provided feedback and the result for your assignment on TAFESA Learn. You will be and given the chance to resubmit with required corrections only once. Refer to the TAFE SA assessment policy for more information https://www.tafesa.edu.au/apply-enrol/before-starting/student-policies/assessment

Assignment – Data Structures and Algorithms

Requirements Specification

IT Works has recently won a contract to implement an advanced application known as the TafeSA Enrolment System. The systems analyst has completed a design document as shown in the Appendix below. The first stage of the project is to develop a library of data structures and algorithms that will be used in the second stage of the project when building the web view front end. The scope of this project is for you to complete the first stage library.

The organization anticipates that this library will undergo several changes over the period of the project and it is recommended that source code be checked-in and out from a Local Git Repository in Visual Studio as the means of implementing an effective version control strategy.
The organizations software development chief architect had previously assessed the available data structures in the C#.NET libraries library and found them inadequate for the new onlinn enrolment system.. Several other projects will also require the use of this customized library of dynamic data structured components written in C#.NET.

Part 1: Dynamic Data Structures Specification

The library should comprise a set of generic dynamic data structures that are defined below. Use Visual studio to create your data structures as .NET Classes in C#

1.1. Implement a generic double linked list to enable the inclusion of any C# objects of type <T>. Include methods to delete, traverse, find and insert nodes at the head and tail and at any position in the list .
Document an explanation of your design by demonstrating diagrammatically how a double linked list works when deleting a node from the middle of a list with at least 5 items. Two 2 diagrams required, one before and one after deletion showing the links between nodes and how they change.

1.2. Implement a generic Binary Tree data structure for adding and traversing to retrieve C# objects of type <T>. You Binary Tree class should implement the three different traversal methods, PreOrder, PostOrder and InOrder
Document an explanation of your design by demonstrating diagrammatically how a balanced binary Search Tree of at least 7 nodes containing Strings would output the data if traversed using the InOrder traversal methods

1.3. Use your GitHub account to check-In your library of classes and document the GitHub repository link and credentials in the appropriate section (1) of the Project Documentation to enable the Project lead to review your code and provide you with feedback after each iteration, where features and other code /design related issues will be negotiated during this iteration.

The feedback/negotiations will be conducted orally in a question/answer session and documented in the 'Verification and Validation' section (2) of the Project Documentation - which will be finalized by the end of the project.
NOTE: You will be assessed on your communication skills and your ability to negotiate changes with the Project lead.

Part 2: Hashing Specifications

One of the classes identified in the Domain model found in the Appendix below is a Student class. You are required to select two properties of this class that would be suitable for Hashing and identify and implement an appropriate Hashing technique using .NET frameworks Object.*GetHashCode ()* and *GetHasCodeUnique ()* methods as optimization techniques to enable the efficient sorting and searching of several instances of Student objects.

Include as comments in your code the reason(s) for selecting the two properties as the best choices for hashing. Include these comments and explain in about 100 words, how using the above two Hashing techniques can optimize the efficient searching and sorting of items in a Collection in the Project Documentation under 'Hashing Specifications'

Part 3: Sorting Specifications

As part of the requirements specification, you have also been tasked with examining the different sorting and searching techniques with the view of selecting the most appropriate techniques for the components in the library to be developed.

Select *three* sorting algorithms and document the one advantage and one disadvantage of each of them using the appropriate section (3) in the Project Documentation
Out of the three documented sorting algorithms, select a sorting technique and implement a **Utility** class that is capable of sorting any C# object of Type <T> in both ascending and descending order – Use two different functions to sort in ascending and descending order respectively.

Part 4: Searching Specifications

Searching for data in your library of data structures is also part of the requirements specification. Using the previously created **Utility** class, implement two functions that can search through a sorted (binary search) and unsorted (sequential search) List of any C# objects of type <T>.
Use the Student and Course classes defined in the Appendix below to implement, and override the IComparator interface CompareTo() to determine if the objects are equal, greater or less than each other.

Part 5: Exception Handling Specification

Use the C#.NET language features that catch and handle/respond any Operating System and other Exceptions that may occur during the execution of any code during testing (see Part 8 on Testing) with exceptions responded to by outputting to the Console and error specify message.. During the Testing phase, all errors/exceptions must be documented in the Error Report section(4) of the Project Documentation

Part 6: Tools | Debugging | Coding Specifications

Use the version of Visual Studio available at IT Works, to create the customized libraries, and use of the built-in IDE debugging tool to detect and resolve syntactical, runtime and logic/design errors, by including break-points in various sections of your code to find and resolve the errors that occurred during testing. After making changes to your code that would resolve the errors, run the code again in de-bugging mode until all errors have been resolved.

Document the process by including screen shots of the break points and comments on the errors that you found in the appropriate section (4) of the Project Documentation.

To achieve the organizations standard for code maintenance, all aspects of your code must adhere to IT Works C#.NET coding standard found at: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/

Use the comment generating features of Visual Studio to document all comments made in your code, for example when explaining how each Data Structure method works.

Part 7: Version Control Specifications

All source code should be checked into a Local Git Repository in Visual Studio as the means to implementing proper version control. All code changes should be made on code that is checked out of the repository. After completion of any code changes, the module must be checked-in to the Repository. Include the link details in section (1) of the Project Documentation to enable the project lead to review your code changes and provide you with feedback on the functionality of your code through an oral question and answer session.

Part 8: Testing

Use NUnit or a testing framework from another third-party library.

Evaluate the third-party tools API documentation to guide you in setting up and coding the relevant test harnesses- and how the results could be captured and documented. Test code must implement the relevant Exception handling features of the language

Run your Unit Tests and document the test results of the implemented tests using the Test Plan template found in section 5(a) of the Project Documentation.

8.1 Testing your Searching and Sorting Algorithms

8.1.1 Create a method to search an Array of ten Student objects using a Linear Search algorithm (You will have to use the Equals ( ) or the '==' overloaded operator of the Student (which you created earlier) to compare the students in the array for equality.

8.1.2 Create another method to search an Array the above student objects using a Binary Search algorithm. Use the Array. Sort ( ) to sort the students by Name, before applying the Binary Search.

8.1.3. Create a method to Sort the above array of Students instances in ascending order of Name using a Bubble Sort algorithm. Use the overloaded relational operators of the Student class when making the Comparisons using the Relational operators .

Run all the above tests and document your test results as screen shots in the Testing section (5(b)) in the Project Documentation.

8.2 Testing Linked Lists Data Structures

8.2.1 -Testing Single Linked List

(a) Create a test method and add a Student instance to the Head of the linked list and enumerate through list to ensure it was added at the Head of the list. Check the results with the expected values and document this in the Test Plan

(b) Create a test method an add a Student instance to the Tail of the linked list and enumerate through list to ensure it was added at the Tail of the List. Check the results with the expected values and document this in the Test Plan

(c) Create a test method and using the appropriate method in the linked list find if a particular student instance is found in the list. Check the result with the expected values and document this in the Test Plan

(d) Create a test method to Remove a Student instance from the beginning of the list and test if it worked. Check the result with the expected values and document this in the Test Plan

(e) Create a test method to Remove a Student from the end of the list and test if it worked. Check the result with the expected values and document this in the Test Plan

8.2.2 – Testing Doubly Linked List

Create methods to implement above tests (a – e) to test your Doubly Linked List

Run all the above tests and document your test results as screen shots in the Testing section (5(c)) in the Project Documentation

8.3 Testing Binary Tree Data Structures

Implement a test method and create BTree instance. Add some Student objects with random id numbers to the tree.

Run all the above tests and document your test results as screen shots in the Testing section (5(c)) in the Project Documentation

Part 9: Further Research

The lead architect of IT Works is looking at projects the company has in its pipeline and has tasked you to research and report on the following areas to support the planning and implementation of these projects:

1. The .NET framework offers you the option of programming in a number of .NET programming languages such as C++, C# and VB.NET. The lead architect has asked you to complete the following Table analysing the suitability for using each listed languages for different types of application development.
   Add this table to Section 9 of the Project Documentation.

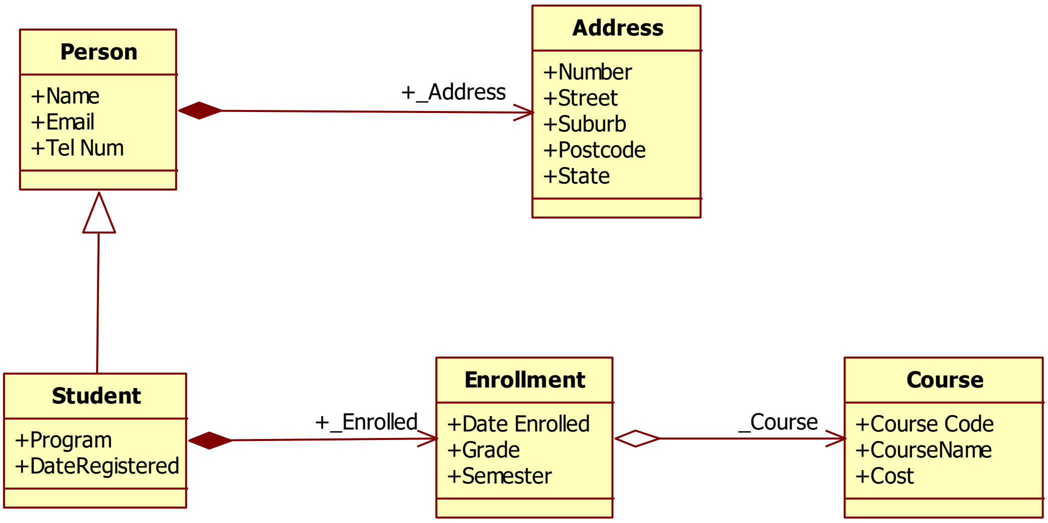
Language	**Suitability** Provide at least 2 reasons why it would be suitable for this type application
C++	Data Structure Utilities and MVC (GUI) Applications:
C#	Data Structure Utilities and MVC (GUI) Applications:
VB.NET	Data Structure Utilities and MVC (GUI) Applications:

2. The organization is keen on moving to a more Agile approach when developing larger scale applications as they feel the current ad-hoc approach would be inadequate. Your experience and research has convinced you that the SCRUM approach will be best suited for the company’s future large scale project management/implementation. Outline four reasons (100 words) why agile techniques are suitability for the development large scale application

Document your findings in a Section 9 of the Project Documentation.

Appendix

TafeSA Enrolment System – Domain Model


Person
+Name
+Email
+Tel Num
+_Address
Address
+Number
+Street
+Suburb
+Postcode
+State
Student
+Program
+DateRegistered
+_Enrolled
Enrollment
+Date Enrolled
+Grade
+Semester
_Course
Course
+Course Code
+CourseName
+Cost