ASSIGNMENT 1 FRONT SHEET

Qualification	**BTEC Level 5 HND Diploma in Computing**		
Unit number and title	Unit 19: Data Structures and Algorithms		
Submission date		**Date Received 1st submission**	
Re-submission Date		**Date Received 2nd submission**	
Student Name	Nguyen Quoc Viet	**Student ID**	Gcc18157
Class	GCC0701-1649	**Assessor name**	Duong Trung Nghia
Student declaration I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.			
		Student's signature	Nguyen Quoc Viet

Grading grid

P1	P2	P3	M1	M2	M3	D1	D2

❒ Summative Feedback:	❒ Resubmission Feedback:	
Grade:	**Assessor Signature:**	**Date:**
Internal Verifier's Comments:		
IV Signature:		

ASSIGNMENT 1 BRIEF

Qualification	**BTEC Level 5 HND Diploma in Business**
Unit number	Unit 19: Data Structures and Algorithms
Assignment title	Examine and specify ADT and DSA
Academic Year	

Unit Tutor			
Issue date		**Submission date**	
IV name and date			

Submission Format:
Format: The submission is in the form of an individual written report and a presentation. This should be written in a concise, formal business style using single spacing and font size 12. You are required to make use of headings, paragraphs and subsections as appropriate, and all work must be supported with research and referenced using the Harvard referencing system. Please also provide a bibliography using the Harvard referencing system. *Submission* Students are compulsory to submit the assignment in due date and in a way requested by the Tutors. The form of submission will be a soft copy in PDF posted on corresponding course of http://cms.greenwich.edu.vn/ *Note:* The Assignment *must* be your own work, and not copied by or from another student or from books etc. If you use ideas, quotes or data (such as diagrams) from books, journals or other sources, you must reference your sources, using the Harvard style. Make sure that you know how to reference properly, and that understand the guidelines on plagiarism. *If you do not, you definitely get fail*
Assignment Brief and Guidance:
Scenario: You work as in-house software developer for Softnet Development Ltd, a software body-shop providing network provisioning solutions. Your company is part of a collaborative service provisioning development project and your company has won the contract to design and develop a middleware solution that will interface at the front-end to multiple computer provisioning interfaces including SOAP, HTTP, JML and CLI, and the back-end telecom provisioning network via CLI . Your account manager has assigned you a special role that is to inform your team about designing and implementing abstract data types. You have been asked to create a presentation for all collaborating partners on how ADTs can be utilised to improve software design, development and testing. Further, you have been asked to write an introductory report for distribution to all partners on how to specify abstract data types and algorithms in a formal notation. **Tasks** **Part 1**

You will need to prepare a presentation on how to create a design specification for data structures, explaining the valid operations that can be carried out on the structures using the example of:

1. A queue ADT, a concrete data structure for a First In First out (FIFO) queue.
2. Two sorting algorithms.

Part 2

You will need to provide a formal written report that includes the following:

1. Explanation on how to specify an abstract data type using the example of software stack.
2. Explanation of the advantages of encapsulation and information hiding when using an ADT.
3. Discussion of imperative ADTs with regard to object orientation.

<table>
<tr><th colspan="3">Learning Outcomes and Assessment Criteria</th></tr>
<tr><th>Pass</th><th>Merit</th><th>Distinction</th></tr>
<tr><td colspan="2">LO1 Examine abstract data types, concrete data structures and algorithms</td><td rowspan="2">D1 Analyse the operation, using illustrations, of two sorting algorithms, providing an example of each.</td></tr>
<tr><td>P1 Create a design specification for data structures explaining the valid operations that can be carried out on the structures.

P2 Determine the operations of a memory stack and how it is used to implement function calls in a computer.</td><td>M1 Illustrate, with an example, a concrete data structure for a First In First out (FIFO) queue.
M2 Compare the performance of two sorting algorithms.</td></tr>
<tr><td colspan="2">LO2 Specify abstract data types and algorithms in a formal notation</td><td rowspan="2">D2 Discuss the view that imperative ADTs are a basis for object orientation and, with justification, state whether you agree.</td></tr>
<tr><td>P3 Using an imperative definition, specify the abstract data type for a software stack.</td><td>M3 Examine the advantages of encapsulation and information hiding when using an ADT.</td></tr>
</table>

P1 Create a design specification for data structures explaining the valid operations that can be carried out on the structures.

Data structures used in this case are Linked List, Linked implementation of a Stack, Linked implementation of a Queue

Singly Linked List: A singly linked list is a list whose node includes two data feilds: inFo and next. The information feild is used to store information, and this is important to the user. The next feild is used to link to its successor in this sequence. The Following image depicts a simple integer linked list.

Stack:

- A stack is a linear data structure that can be accessed only at one of its ends for storing and retrieving data
- A stack is a Last In, First Out (LIFO) data structure
- Anything added to the stack goes on the "top" of the stack
- Anything removed from the stack is taken from the "top" of the stack
- Things are removed in the reverse order from that in which they were inserted

Queue:

- A queue is a waiting line that grows by adding elements to its end and shrinks by taking elements from its front
- A queue is a structure in which both ends are used:
    + One for adding new elements
    + One for removing them
- A queue is an FIFO structure: First In/First Out

In SMS, I use these data structures because it's simple and suitable For SMS's data. Singly linked list has so many advantages:

- Insertions and Deletions can be done easily.
- It does not need movement of elements for insertion and deletion
- It space is not wasted as we can get space according to our requirements
- Its size is not fixed.
- It can be extended or reduced according to requirements
- Elements may or may not be stored in consecutive memory available even then we can store the data in computer.

P2 Determine the operations of a memory stack and how it is used to implement function calls in a computer.

Define

What are Stacks in Computer Memory?

Memory stacks are linear data structures (locations) used to store data in a computer's memory. They may also be referred to as queues. Data within a stack must always of the same type.

Items in a stack are inserted or removed in a linear order and not in any random sequence. As in any queue or collection that is assembled, the data items in a stack are stored and accessed in a specific way. In this case, a technique called LIFO (Last In First Out) is used. This involves a series of insertion and removal operations which we will discuss in the following section.

1. The Push Operation

The push operation involves inserting data items into a stack. Let us examine our restaurant plate dispenser in Figure 2. The push operation adds plates (data items) to the plate dispenser (stack). The first plate is pushed to the bottom of the stack with all subsequent plates following in order after it. The first data item inserted is the most inaccessible and positioned at the bottom of the stack.

2. The Pop Operation

The pop operation involves removing data items from a loaded stack. In our plate dispenser illustration, the last plate (data item) added is positioned at the top of the stack. This data item is popped out of the stack as the first item to be removed. Think of the spring loading system at the base of the plate dispenser. It pushes the stack of plates upwards each time a plate is removed. In memory, items continue to be popped out in that order.

3. The Peek Operation

In this operation, no data item is added to or removed from the stack. The peek operation simply requests the address location of the data item at the top of the stack (the last item to be pushed)

4. The search operation.

In this operation, no data item is added to remove from the stack either. The search operation requests the address location of any data item in the stack. It determines item location in reference to the item at the top the stack.

P3 Using an imperative definition, specify the abstract data type for a software stack

An Abstract Data Type (ADT) is a collection of objects and functions, that is, algebra, where one ignores how the objects are constructed and how the functions are implemented. More precisely, if A and B are isomorphic algebras, that is, there exists a choice between A and B that respects the operations, then A and B are regarded as identical.

What is the Java Abstract Data Type (ADT)?

Java abstract data type (ADT) in a data structure is a type of data type whose behaviour is defined by a set of operations and values.

In Java abstract data type, we can only know what operations are to be performed and not how to perform them i.e. it does not tell how algorithms are to be implemented or how the data will be organized in the memory. This type is thus called abstract as it does not give the view of implementation.

1. List ADT

A list abstract data type is the type of list which contains similar elements in sequential order and following are the operations which can be performed on list.

I. get – Return an element from the list.
II. insert – Insert an element at any position.
III. remove – Remove the first occurrence of any element from a non-empty list.
IV. removeAt– Remove the element at a predefined area from a non-empty list.
V. Replace – Replace an element by another element.
VI. size– Returns number of elements of the list.
VII. isEmpty – Return true if the list is empty, else return false.
VIII. isFull – Return true only if the list is full, else return false.

2. Stack ADT

A stack contains similar elements which are in an ordered sequence. All the operations in stack takes place at the top of the stack. Following are the operations which are performed on the stack-

I. push – Insert an element at the top of stack.
II. pop – Remove an element from the top of the stack, If it is not empty.
III. peep – Returns the top element of stack without removing it.
IV. size – Returns the size of the stack.
V. isEmpty – Return true if the stack is empty, else it returns false.
VI. isFull – Return true if the stack is full, else it returns false.

3. Queue ADT

The elements of the queue are of the same type which are arranged sequentially. The operations can be performed at both ends, insertion is done at rear end deletion is done at the front end. Operations performed on the queue are as follows.

I. enqueue() – Inserting an element at the end of queue.

II. dequeue() – Removing an element of the queue.

III. peek() – Returns the element of the queue without removing it.

IV. size() – Returns the number of elements in the queue.

V. isEmpty() – Return true if the queue is empty, else it returns false.

VI. isFull() – Return true if the queue is full, else it returns false.

https://data-flair.training/blogs/abstract-data-type-adt/

https://en.wikipedia.org/wiki/Command-line_interface