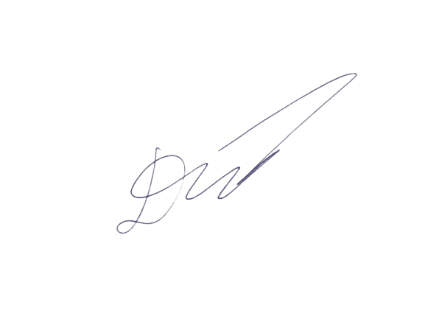
**ASSIGNMENT 1 FRONT SHEET**



|  |  |  |  |
| --- | --- | --- | --- |
| **Qualification** | **BTEC Level 5 HND Diploma in Computing** | | |
| **Unit number and title** | Unit 19: Data Structures and Algorithms | | |
| **Submission date** | 24/08/2022 | **Date Received 1st submission** |  |
| **Re-submission Date** |  | **Date Received 2nd submission** |  |
| **Student Name** | Mai The Duc | **Student ID** | GCH200681 |
| **Class** | GCH0907 | **Assessor name** | Hong-Quan Do |
| **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** |  |

**Grading grid**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **P1** | **P2** | **P3** | **M1** | **M2** | **M3** | **D1** | **D2** |
|  |  |  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **❒ Summative Feedback: ❒ Resubmission Feedback:** | | |
| **Grade:** | **Assessor Signature:** | **Date:** |
| **Internal Verifier’s Comments:** | | |
| **IV Signature:** | | |

Table of Contents

[Table of figures 4](#_Toc112276293)

[Introduction 5](#_Toc112276294)

[I. Abstract data type (ADT) 5](#_Toc112276295)

[I.1. Definition 5](#_Toc112276296)

[II.2. Example of abstract data type: 6](#_Toc112276297)

[II. Stack ADT 7](#_Toc112276298)

[III. Memory Stack 10](#_Toc112276299)

[IV. Queue ADT 13](#_Toc112276300)

[V. Explaination on how to specify an abstract data type using the example of software stack 14](#_Toc112276301)

[1. Introduction 14](#_Toc112276302)

[2. Describe 14](#_Toc112276303)

[3. Formal specification language 15](#_Toc112276304)

[References 16](#_Toc112276305)

# Table of figures

[Figure 1: Linked List 1 6](#_Toc112276283)

[Figure 2: Linked List 2 6](#_Toc112276284)

[Figure 3: Stack operation - push() 7](#_Toc112276285)

[Figure 4: Stack operation - pop() 8](#_Toc112276286)

[Figure 5: Stack operation - peek() 8](#_Toc112276287)

[Figure 6: Stack application/example 9](#_Toc112276288)

[Figure 7: Memory stack operation 1 11](#_Toc112276289)

[Figure 8: Memory stack operation 2 12](#_Toc112276290)

[Figure 9: Memory stack operation 3 12](#_Toc112276291)

[Figure 10: Queue 13](#_Toc112276292)

# Introduction

I am a software developer working for Softnet Development Ltd. The company providing network provisioning software solution. Softnet has won a value contract last session. We design and develop a middleware solution which can be interface at the front-end to multiple computer provisioning interfaces and the back-end telecom provisioning network.

In my report, I will inform my team about designing and implementing abstract data types. Also, I’ll present how improve software design, development and testing by using ADTs. Furthermore, how to specify abstract data types and algorithms will be introduced.

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

# I. Abstract data type (ADT)

## I.1. Definition

ADT defines a particular data structure in terms of data and operations. It also offers and interface of the objects as instances of an ADT.

An Abstract Data Type (ADT) consist of:

* Declaration of data.
* Declaration of operations
* Encapsulation of data and operations.

ADT is unclear what algorithms will be used to carry out the operations and how the data will be organised in memory. Because it provides an implementation-independent view, it is called "abstract" (Chauhan, 2022).

Encapsulation means the data is hidden from user and can be controlled only by means of operations. We do not need to know how that data type is implemented, we only need to know how that data type can do. ADT have operations to help us use a data type. I will define Stack ADT and Queue ADT.

## II.2. Example of abstract data type:

Below is Linked List:

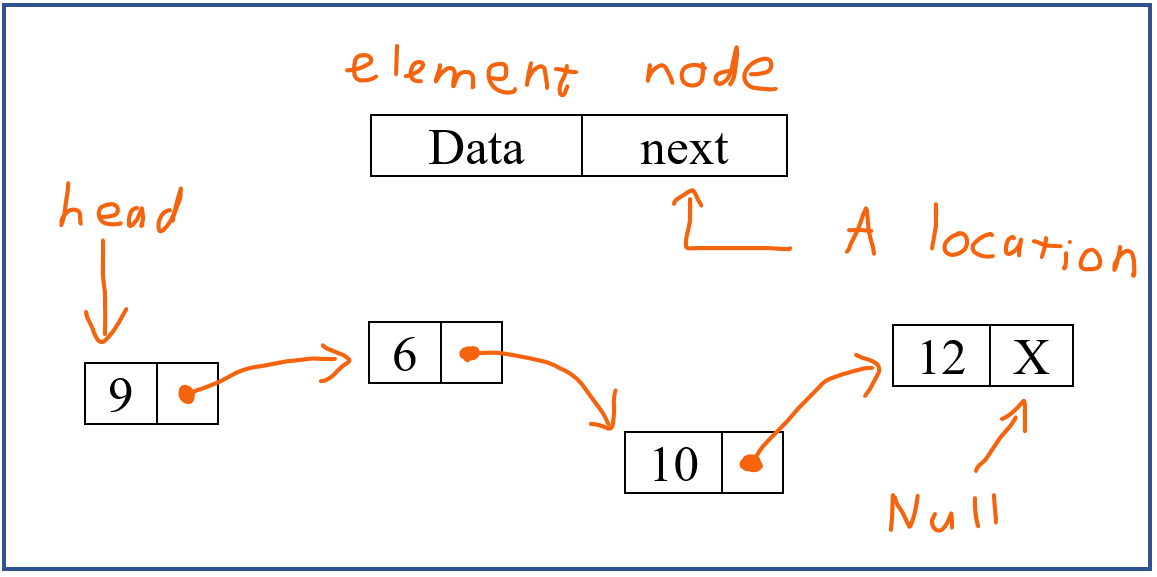


Figure : Linked List 1

‘next’ is a location in RAM.

For example:

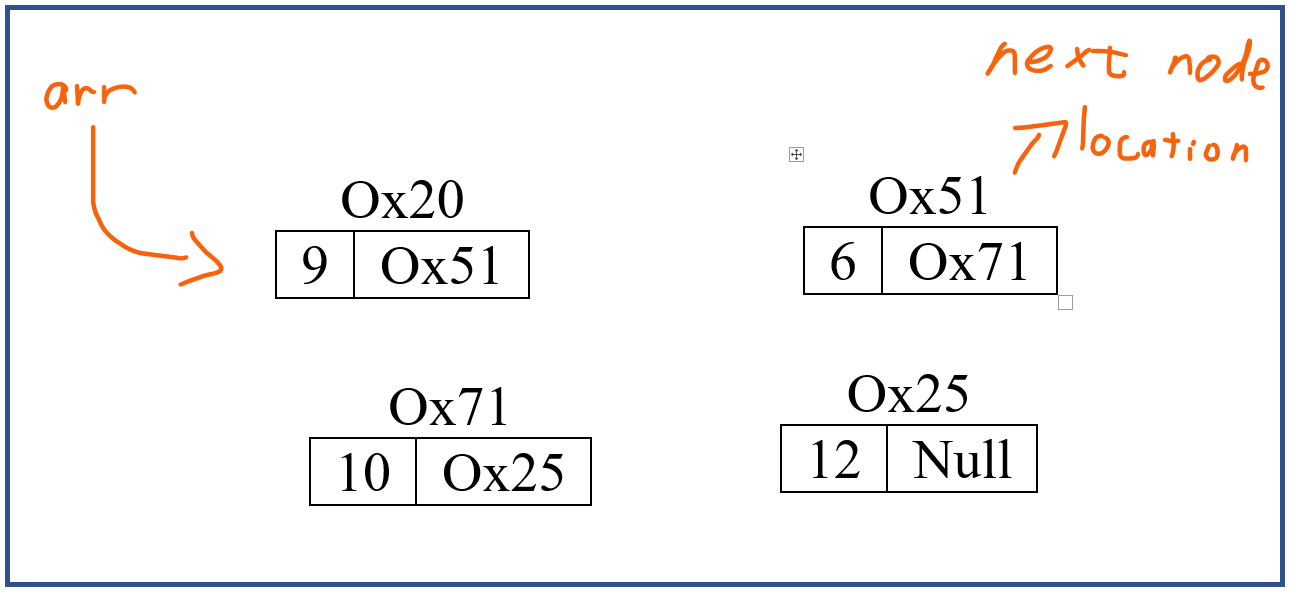


Figure : Linked List 2

# II. Stack ADT

Stack is a linear data structure which keeps the operations performed in a specific order (GeeksforGeeks, 2022). Stack is the implantation order of LIFO (Last In First Out) or some call it FILO (First In Last Out).

According to Oracle (2022), Primitive operations of a Stack are:

* empty() : To test if the stack is empty.
* peek() : To looks at the object at the top of this Stack without removing it from the stack.
* pop() : To removes the object at the top of this Stack.
* push(E item) : To pushes an item onto the top of this stack.
* search(Object o) : To search a item in this Stack

+) push

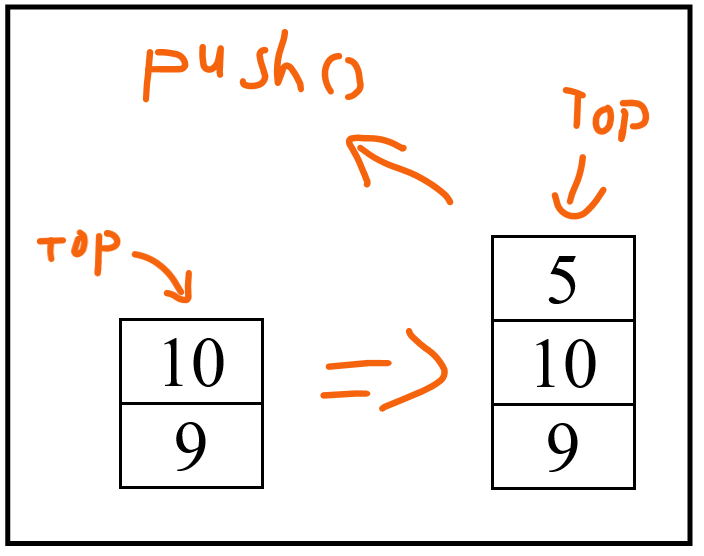


Figure : Stack operation - push()

+) pop

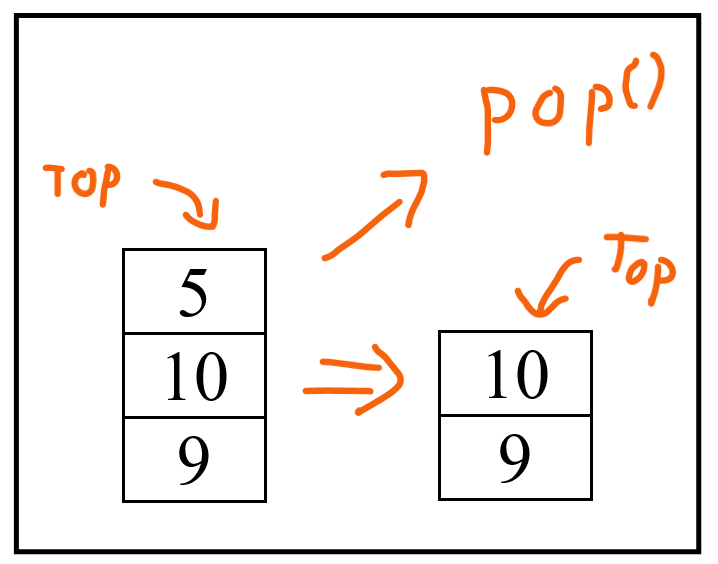


Figure : Stack operation - pop()

+) peek

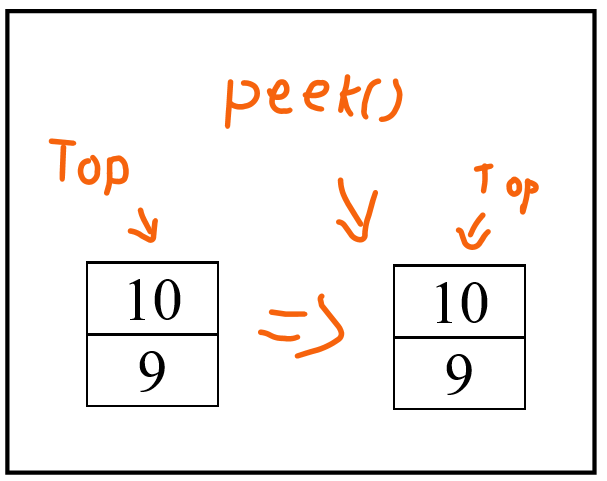


Figure : Stack operation - peek()

Application / example:

A String can be reversed using Stack by adding(push) each character one at a time to the stack and removing(pop) them one at a time. The String will now be returned in the reverse way.

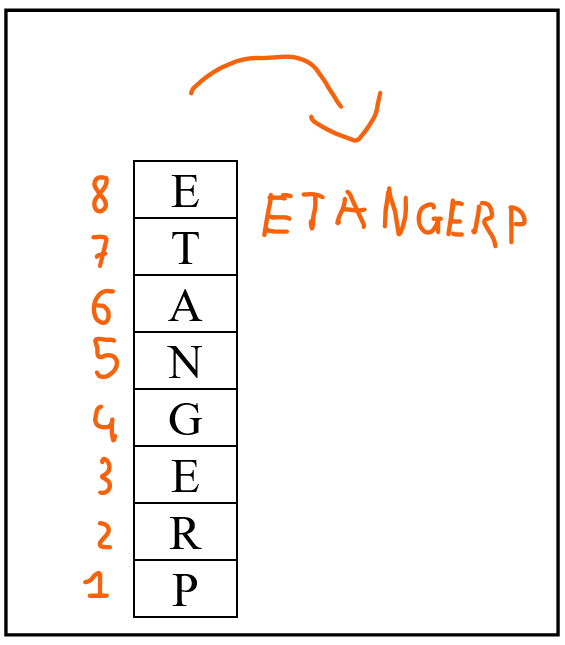


Figure : Stack application/example

Input: PREGNATE 🡪 ETANGERP

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

# III. Memory Stack

A stack is a distinctive section of computer memory where temporary variables created by a function are kept. Variables in a stack are declared, saved, and initialised at runtime.

Accorsing to Martin (2022), Stack have some advantages and disadvantages of using Stack memory.

+) Advantages of Stack memory:

* Allow user to control how memory is allocated and deallocated.
* Stack automatically cleans up the object.
* Not easily corrupted.
* Variables can not be resized.

+) Disadvantages of Stack memory:

* Stack memory is very limited.
* Creating too many object on the stack can increase the risk of stack overflow.

Example of using memory stack operation to implement function calls in a computer:

I create a function of summation, my target is summation of every number from 1 to n (number).

For example:

n = 6

* Result = 1 + 2 + 3 + 4 + 5 + 6

**public** int Main(String[] args)

    {

        int result = Sum(6);

    }

**public** int Sum (int n)

    {

        if( n == 1 ) return 1;

        else return Sum( n - 1 ) + n;

    }

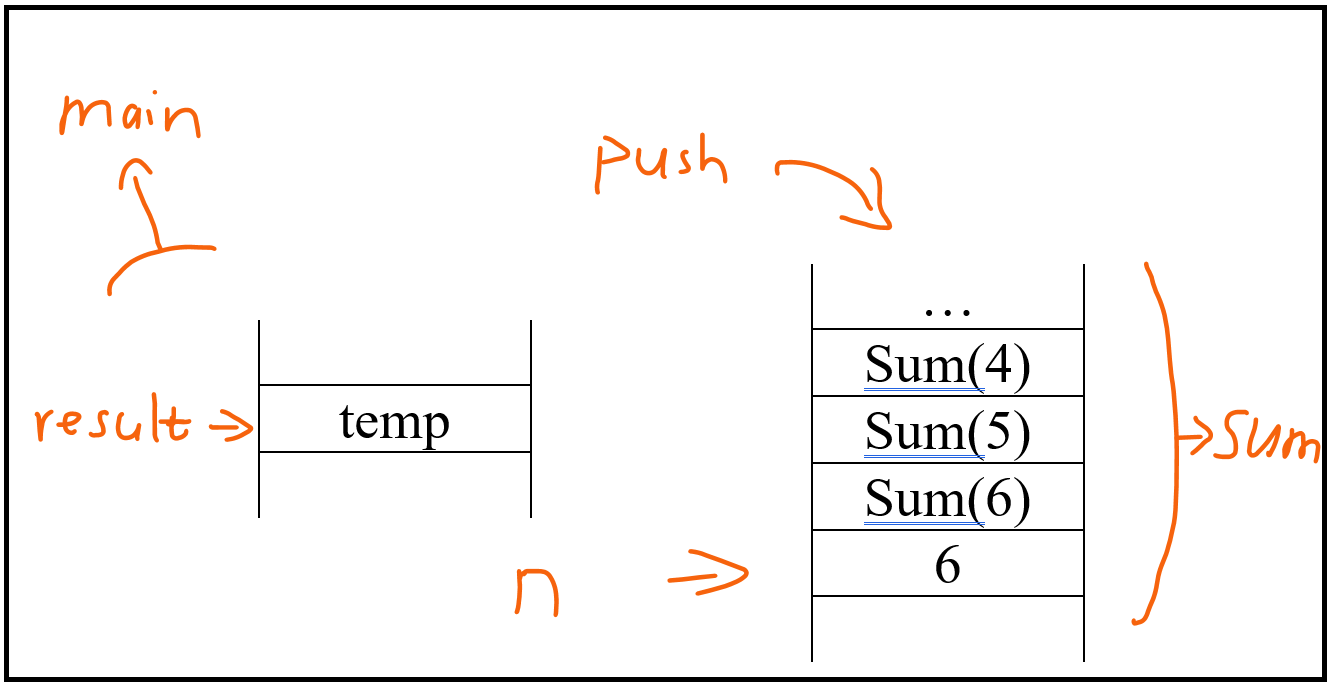


Figure : Memory stack operation 1

By using recursive method, the computer will calculate Sum(1) then Sum(2) and go on. Therefore, Sum(2) is sum of 1 and 2; and Sum(3) is Sum(2) + 3.

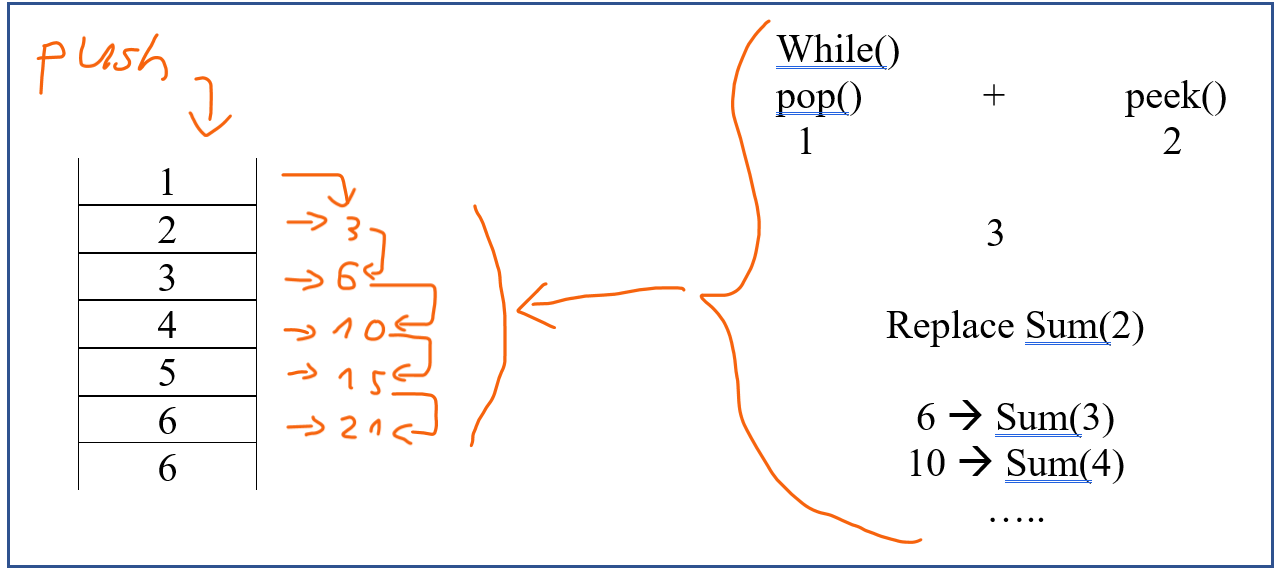


Figure : Memory stack operation 2

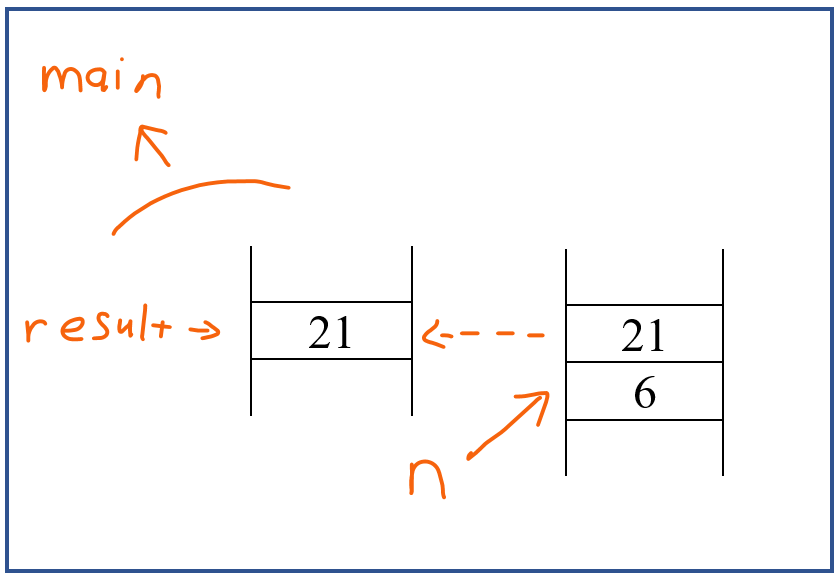


Figure : Memory stack operation 3

The Sum(6) method will return the result = 21.

# IV. Queue ADT

In programming, a queue is a crucial data structure. A queue is open at both ends and operates according to the FIFO (First In First Out) principle. At one end of the queue, known as the rear or tail, data insertion is performed; and at the other end, known as the front or head of the queue, data deletion is performed (Deepali, 2022).

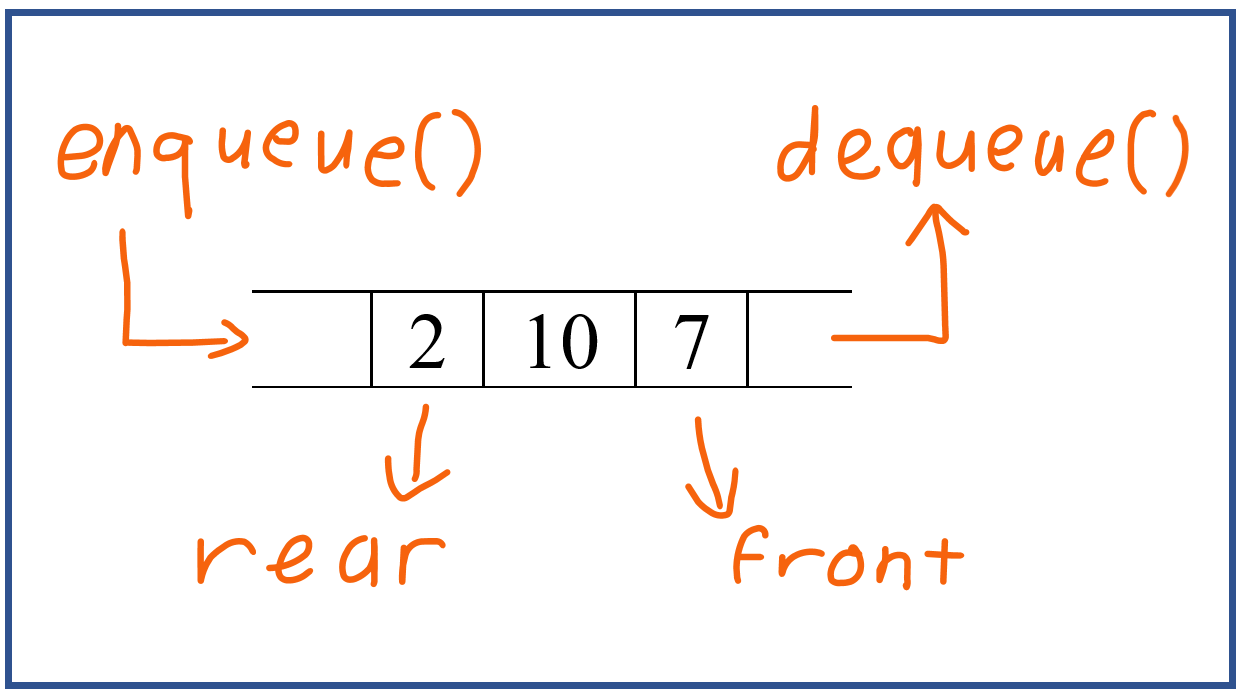


Figure : Queue

Primitive operations of a Stack are:

* add(): Inserts the specified element into this queue if it is possible to do.
* element(): Retrieves, but doesn’t remove, the head of this queue.
* offer(): Inserts the specified element into this queue.
* peek(): Retrieves, doesn’t remove, the head of this queue, or returns null.
* poll(): Retrieves and removes the head of this queue, or return null.
* remove(): Retrieves and removes the head of this queue.

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

# V. Explaination on how to specify an abstract data type using the example of software stack

## 1. Introduction

First-order logic is used to write the pre- and post-conditions in order to specify the operations of the system in the form of axioms. The prerequisites are conditions that must be met in order for an operation to be successfully invoked (Webeduclick.com, 2022).

## 2. Describe

The pre-conditions essentially encapsulate the demands placed on a function's input parameters. The post-conditions are the requirements that must be met in order for a function to be considered successful. They are essentially restrictions on the outcomes that can be produced (Webeduclick.com, 2022).

According to Longman (1997):

* “The precondition statement indicates what must be true before the function is called”.
* “The postcondition statement indicates what will be true when the function finishes its work”.

Error-condition is statement indicates what will be when the function fail to finishes its work.

Example: VDM

The Vienna Development Method, also known as VDM, is a set of procedures for the formal specification and development of computer systems. It includes the VDM-SL specification language, rules for data and operation refinement that enable connections between abstract requirements specifications and detailed design specifications all the way down to the level of code, and a proof theory that enables rigorous arguments about the characteristics of specified systems and the validity of design choices (ViennaCC, 2022).

## 3. Formal specification language

**push** (S: Stack, i: Item)

pre not\_full(S)

post size(S) = n + 1 and top(S) = i

error none

**pop** (S: Stack)

pre not\_empty(S)

post size(S) = n – 1

error is\_empty(S)

**peek** (S: Stack)

pre not\_empty(S)

post top(S)

error is\_empty(S)

# References

Chauhan, A., 2022. *Abstract Data Types.* [Online]   
Available at: https://www.geeksforgeeks.org/abstract-data-types/  
[Accessed 22 August 2022].

Deepali, 2022. *Queue Data Structure: Types, Implementation, Applications.* [Online]   
Available at: https://www.naukri.com/learning/articles/queue-data-structure-types-implementation-applications/  
[Accessed 23 August 2022].

GeeksforGeeks, 2022. *Stack Data Structure (Introduction and Program).* [Online]   
Available at: https://www.geeksforgeeks.org/stack-data-structure-introduction-program/  
[Accessed 22 August 2022].

Longman, A. W., 1997. *Preconditions and Postconditions.* [Online]   
Available at: http://www.cs.albany.edu/~sdc/CSI310/MainSavage/notes01.pdf  
[Accessed 24 August 2022].

Martin, M., 2022. *Stack vs Heap: Know the Difference.* [Online]   
Available at: https://www.guru99.com/stack-vs-heap.html  
[Accessed 22 August 2022].

Oracle, 2022. *Stack.* [Online]   
Available at: https://docs.oracle.com/en/java/javase/18/docs/api/java.base/java/util/Stack.html  
[Accessed 22 August 2022].

ViennaCC, 2022. *VDM Vienna Development Method.* [Online]   
Available at: http://www.vienna.cc/e/evdm.htm  
[Accessed 24 August 2022].

Webeduclick.com, 2022. *Axiomatic and Algebraic Specification in Software Engineering.* [Online]   
Available at: https://webeduclick.com/axiomatic-and-algebraic-specification/  
[Accessed 24 August 2022].