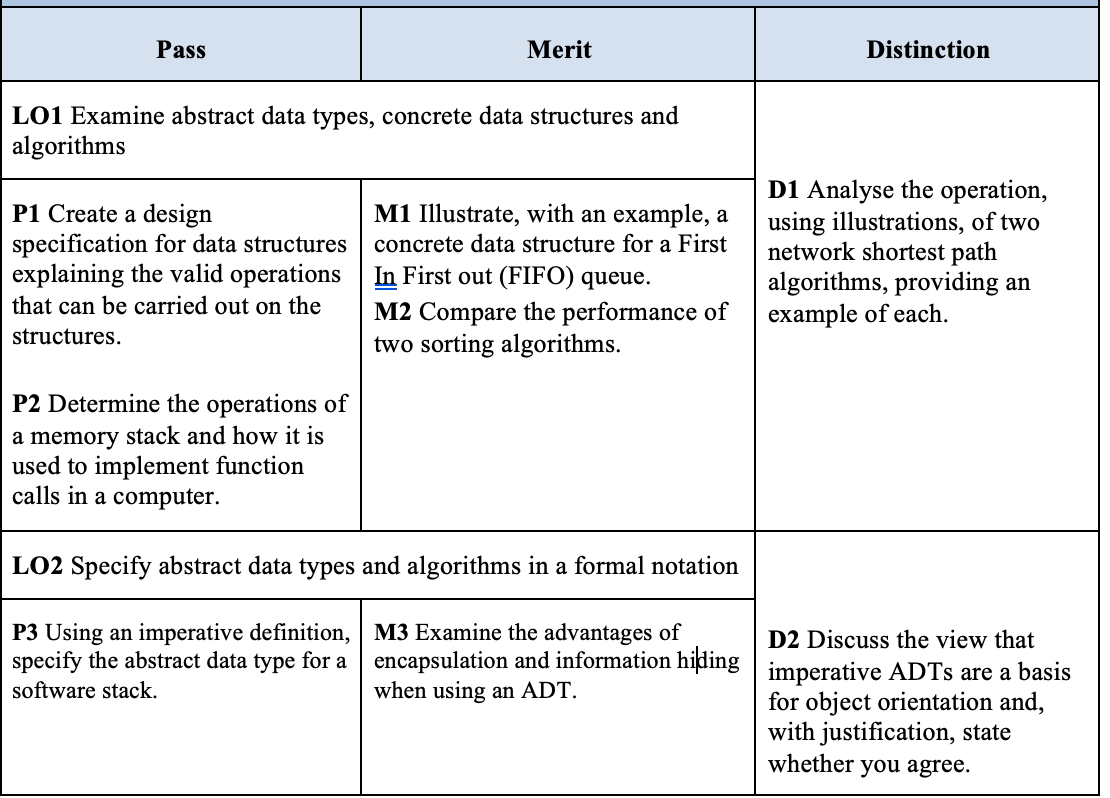
# Brief



# 1. Data structures

## 1.1 Abstract data type (P1)

### 1.1.1 Definition

Give definition of what is ADT, common ways to represent ADT

Hint:

* 1. *What is ADT?*
  2. *How to define and implement ADT? (Base on 5 steps on Page 9 Slide 1 – describes more detail each step by yourseft)*

### 1.1.2 Examples

Give examples of a general ADT, using a representation to illustrate. You can choose one of the following ADT: Linked List, Doubly Linked List, Circular Linked List.

(The best way to illustrate operations of an ADT is using table)

*Hint:*

*Choose an ADT: define what it is, supported operations and describe all them (in natual language) (explanation + illustration).*

## 1.2 ADT usages

### 1.2.1 Application of Stack in memory (P2)

Describe an application of Stack in memory management: how the memory is organized, how a method (function) calls is implemented with stack.

*Hint:*

* *What is a stack? Definition, operation, exceptions of stack.*
* *How memory stack works a*
* *What is a function call in a computer? How does it work?*
* *How can stack work in Function Calls? (shows: code, illustration and detail explanation).*

### 1.2.2 Application of an ADT (P3)

Describe a problem where you need an ADT (Array, Linked List, DLL, CLL), describe that data structure operations can be used for that problem.

*Hint:*

1. *Why you choose describing the ADT in 1.1.2*
2. *What is an imperative ADT or imperative definition in ADT?*
3. *Using an imperative definition, specify your chosen ADT (all supported operations you mentioned in 1.1.2).*
4. *Use your console application to describe the problem.*
5. *Show what effect does the ADT obtain after an operation is called by the user.*

*(using table to describe)*

|  |  |  |
| --- | --- | --- |
| *Code in your main* | *Actual Output* | *Explanation* |
|  |  |  |

### 1.2.3 Application of Queue (M1)

Describe a problem where you need Queue, describe a concrete queue for that problem and why using queue can solve that problem.

Hint:

* 1. What is a queue?
  2. Support operations
  3. Queue exceptions | Pros and Cons of Queue
  4. Describe a problem and why using queue to solve it.
  5. Show your code to solve this problem by calling the Queue’s operations and illustration of them | Code and Illustration

## 1.3 ADT vs OOP (M3, D2)

### 1.3.1 Using OOP to represent ADT

Represent an ADT in Java programming language (you can use abstract class or interface), explain how the ADT operations can be represented by methods of your abstract class / interface.

*Hint:*

1. *What is Encapsulation and Information Hiding?*
2. *The differences between encapsulation and information hiding.*
3. *In Java, what do you use to represent an ADT and why?*
4. *When using an ADT, what is the role of Encapsulation and Information Hiding? (You can choose the ADT you mentioned in 1.1.2 to explain this section easily).*

### 1.3.2 Discussion

Discuss about the state: “imperative ADTs are a basis for object orientation”. Explain your opinion.

Discuss how OOP can help to represent ADT and implement ADT (encapsulation, override, inheritance)

# 2 Algorithms

## 2.1 Sorting algorithms (M2)

What is sorting algorithms? Classify them.

### 2.1.1 Bubble sort (or other O(n2) sort algorithms)

Describe how it works

Analyze its performance (time, memory)

### 2.1.2 Quick sort (or other O(nlogn) sort algorithms)

Describe how it works

Analyze its performance (time, memory)

## 2.2 Shortest path algorithms (D1)

What is the shortest path problem?

### 2.2.1 Dijkstra’s algorithm

Describe how it works by using concrete example

### 2.2.2 Bellman Ford’s algorithm

Describe how it works by using concrete example