

SECTION A [35 Marks]Answer **all** questions**Question I**

1. A student has written this Python algorithm:

Line	Code
1	threshold = 5
2	count = 0
3	while count >= 1 and count <= 20:
4	count = input("Enter number of attempts")
5	count = count + 1
6	for i = 1 to count:
7	print(i * threshold)

- c. The program uses variables. Describe what is meant by a variable

[2]

- c. Give the identifiers of all the variables used in this algorithm.

[1]

2. The student has used a while loop on line 3. State one difference between a count-controlled loop and a condition-controlled loop.

[1]

3. Rewrite lines 06 to 07 to use a while loop instead of a for loop.
You should write your answer using Python code.

[2]

Question II

The current contents of a queue data structure are shown in Fig. 2.1.

Fig.2.1

8	17	4	12			

headPointer points to 8. tailPointer points to the first empty space.

1. State the purpose of headPointer and tailPointer in the queue shown in Fig. 2.1.

[2]

headPointer
tailPointer

2. State the contents of the queue, in order, after the following actions have been executed on the queue shown in Fig. 2.1, in the order given:

[1]

- enqueue(9)
- dequeue()

3. The queue is used to store ID numbers of support tickets that a program needs to process. Some tickets are marked as urgent and need to be processed first.

[1]

State one reason why this queue is not a suitable data structure for this program.

Question III

The contents of a stack are stored in the 1-dimensional array called values.

topStack stores the index of the next free space in the stack.

The array is declared with space for 50 elements.

1. The function `pop()` returns the next item from the stack and updates the appropriate pointer.

Describe the steps in the function `pop()`.

[3]

2. The function push() inserts its parameter called dataValue onto the stack and updates the appropriate pointer.

Complete the function push() using pseudocode or program code.

[3]

Line	Code
1	def push(.....)
2	if!= 50:
3	values[.....] = dataValue
4	topStack = topStack + 1
5	return true
6	else :
7	return false

3. State what value push() returns if dataValue is successfully added to the stack.

[1]

Question IV

The following strings are stored in an array.

"mango"	"apple"	"kiwi"	"fig"	"date"	"pear"
---------	---------	--------	-------	--------	--------

1. Explain how a linear search would search the array for "fig".

[2]

2. State why a binary search could not be used on this array in its current order.

[1]

3. Show how a bubble sort will sort the given data into ascending alphabetical order.

[4]

"mango"	"apple"	"kiwi"	"fig"	"date"	"pear"
---------	---------	--------	-------	--------	--------

4. A sorting algorithm has a worst-case time complexity of $O(n^2)$.
Describe what is meant by the worst-case time complexity $O(n^2)$ for a sorting algorithm.

[1]

Question V

A delivery company stores parcels that are waiting to be delivered. Each parcel has an ID, a destination code, and a priority number that represents the order the parcel should be delivered in (a lower priority number means the parcel is more urgent).

The parcels are stored in a binary search tree, ordered by their priority number.

1. Give two characteristics of a binary search tree.

[2]

2. Give one advantage of storing this parcel data in a binary search tree instead of in a 1-dimensional array.

[1]

3. Tick (✓) one column in each row to identify whether each statement applies to a depth-first (in-order) tree traversal, a breadth-first tree traversal, or neither of these two traversals, when performed on a binary search tree.

[3]

Statement	Depth-first (in-order)	Breadth-first	Neither of these two traversals
All nodes at the current depth are visited before moving to the next depth			
The traversal will output the parcels in ascending order of priority number			
The algorithm processes the root node before either of its child subtrees			

Question VI

A group of students are designing a library management system. They first design a procedural version, then also create an object-oriented version using classes such as Book and Member.

1. State one difference between procedural programming and object-oriented programming. **[1]**

2. State what is meant by abstraction, and describe how it could be used when designing a Book class for this system. **[3]**

Définition	
Use	

SECTION B [35 Marks]

Attempt **all** the questions

Question VII - The Grid and Fruit class (10 marks)

1. The grid is designed to be a 1-dimensional array with the identifier tiles. Explain why an array is a suitable data structure to represent the grid. **[3]**

2. The player and bananas are designed as separate classes. 8 of the tiles on the grid will contain an instance of the class Fruit. The other tiles will be empty.

The class design for Fruit is here.

class: Fruit

attributes: private label : string, private category : string, private points : integer

methods: init(), getLabel(), getCategory(), getPoints()

init() is the constructor method. The label, category and points are passed to the constructor as parameters, which are then assigned to the attributes.

- c. The method getLabel() returns the data in the attribute label.

Write the method getLabel() using pseudocode or program code. [2]

[2]

- c. A global 1-dimensional array, allFruits, stores 8 objects of type Fruit. The fruit in index 5 has the label "Banana", the category is "fruit" and the points value is 40.

Write pseudocode or program code to create a new object for this fruit and store it in index 5 of allFruits.

[2]

--	--

--	--

- c. The game starts with 8 fruits. Each fruit is allocated to one tile on the grid. An algorithm needs designing that will generate a random tile on the grid for each fruit. Each tile can only store one fruit.

Describe the decisions that will need to be made in this algorithm and how these will affect the program flow.

[3]

Question VIII– The Player class and movement algorithm (13 marks)

1. The class design for Player is here.

class: Player

attributes: private name : string, private score : integer, private stamina : integer, private tilePosition : integer

methods: init(), getName(), getScore(), getStamina(), getTilePosition(), changePosition(), collectItem()

The four get methods return the associated attribute. The number of tiles to move is passed to changePosition() as a parameter. The method adds this value to the player's position on the grid.

The category and points of an object are passed to collectItem() as parameters:

- If the category is "fruit" the points are added to the player's score.
- If the category is "hazard" the points are subtracted from the player's stamina.
- If the category is neither "fruit" nor "hazard" no changes are made.

- a. init() is the constructor method. The name of the player is passed into the constructor as a parameter. The constructor then initialises the stamina to 10 and initialises both the score and tile position to 0.

[4]

Write the constructor method for Player using either pseudocode or program code. You do not need to declare the class, the attributes or any other methods.

--	--

- b. Write pseudocode or program code for the method `collectItem()`.
For example, for the Player `player1`:
`player1.collectItem("fruit", 10)` updates `player1`'s score by 10
`player1.collectItem("hazard", 5)` updates `player1`'s stamina by -5
`player1.collectItem("foo", 9)` has no effect on `player1`.

[4]

--	--

--	--

c. This incomplete python algorithm:

- creates a new player with the name Sonam
- loops until the player reaches the end of the grid
- generates a random number of tiles to move between 1 and 3 (including 1 and 3)
- moves the player and checks if the new tile has a fruit
- updates the player attributes if there is a fruit
- outputs the player's new tile position and, if found, the name of the fruit.

Complete the python algorithm.

[5]

line	Code
01	player1 = ... ("Sonam")
02	newPosition = 0
03	while newPosition <:
04	move = random.randint(1, 3): #this will generate a random number between 1 and 3
05	player1.changePosition(move)
06	newPosition = player1.getTilePosition()
07	newPosition < 40 and tiles[.....] != null:
08	itemCategory = tiles[newPosition].getCategory()
09	itemPoints = tiles[newPosition].getPoints()
10	player1.collectItem(....., itemPoints)
11	print("You are on tile", newPosition, "and found", tiles[newPosition].getLabel())
12	print("You reached the end of the grid")

Question IX – Debugging and encapsulation (12 marks)

1. The procedure `displayGrid()` outputs the contents of each tile on the grid. The number of each tile is output with either "empty" (if there is no gem) or the points value of the gem (if there is one).

Line	Code
01	<code>def displayGrid():</code>
02	<code> for x in range (0, 45):</code>
03	<code> print("Tile", y)</code>
04	<code> if tiles[x] == null:</code>
05	<code> print("empty")</code>
06	<code> else:</code>
07	<code> print(tiles[x].getPoints())</code>

The algorithm contains errors. Give the line number of three different errors and write the corrected line for each error.

Error 1 — Error line: Correction:

Error 2 — Error line: Correction:

Error 3 — Error line: Correction:

[3]

2. * A programmer has created the Gem and Player classes above with all attributes marked private, accessed only through getter methods. A colleague suggests making the attributes public instead, to simplify and speed up development.

Compare the use of public and private (encapsulated) attributes in this program.

You should include the following in your answer:

- the use of public and private attributes
- alternative methods to direct public access, such as getters/setters or property decorators
- the appropriateness of each to this program design.

[9]

--	--