PART A – OBJECTIVE QUESTIONS **[20 MARKS]**

Part A consists of 20 objective questions. Each question carries 1 mark. Choose the correct answer and write your answer in the answer booklet.

1. What is the main advantage of performing a Sorted Sequential Search compared to unsorted Sequential Search?

   A. It eliminates the need for a search key.
   B. It can significantly reduce the number of comparisons.
   C. It transforms the time complexity to $O(\log_2 n)$.
   D. It works well with large data sets.

2. What is the worst-case time complexity of performing a search with the Binary Search algorithm?

   A. $O(1)$
   B. $O(n)$
   C. $O(\log_2 n)$
   D. $O(n \log_2 n)$

3. Analyses the erroneous program given below. Which of the following modifications is required to correct the given program that uses Binary Search to locate an element x in an array Y?

```
1.  f(int Y[10], int x){
2.        int i, j, k;
3.        i = 0; j = 9;
4.        do {
5.              k = (i + j) / 2;
6.              if (Y[k] < x) i = k; else j = k;
7.        } while (Y[k] != x && i < j);
8.        if (Y[k] == x) printf("x is in the array ");
9.        else printf(" x is not in the array ");
10. }
```

   A. Change line 6 to: `if (Y[k] < x) i = k + 1; else j = k-1;`
   B. Change line 6 to: `if (Y[k] < x) i = k - 1; else j = k+1;`
   C. Change line 6 to: `if (Y[k] <= x) i = k; else j = k;`
   D. Change line 7 to: `} while ((Y[k] == x) && (i < j));`

4. Given an array A = {11, 17, 21, 35, 41, 57, 63, 66, 73, 74, 89} and a key = 41. By using the Binary Search algorithm, what are the mid values (corresponding array elements) generated in the **second** and **third** iterations?

   A. 35 and 57
   B. 57 and 35
   C. 21 and 35
   D. 66 and 57

5. In a circular array implementation of a queue, what happens when the rear reaches the end of the array?

   A. An error is thrown.
   B. Rear wraps around to the beginning.
   C. Front is incremented.
   D. The array is resized dynamically.

6. What is a key advantage of using a linked list to implement a queue?

   A. Efficient use of memory.
   B. Constant time random access.
   C. Dynamic size.
   D. Fast front and rear updates.

7. Which operation is more efficient in a circular linked list implementation compared to a linear linked list implementation of a queue?

   A. enqueue
   B. dequeue
   C. Both are equally efficient
   D. Neither is efficient

8. In a circular array implementation, what is the purpose of using the modulo operation?

   A. Ensures even distribution of elements.
   B. Facilitates dynamic resizing.
   C. Avoids the need for a rear pointer.
   D. Handles wrapping around the array.

9. Which of the following statements about **stack ADT** is **FALSE**?

   A. Stack can check whether an expression contains balanced parenthesis.
   B. Insert and delete nodes will follow the FIFO principle.
   C. A new node can only be added at the top of the stack.
   D. The last node at the bottom of the stack has a NULL link.

10. Which of the following represents the manipulation of an array for stack implementation?

A.

	Steps & array content			
Array Index	1	2	3	4
3				
2				
1		B		C
0	A	A	A	A

B.

	Steps & array content			
Array Index	1	2	3	4
3				
2			A	
1		A	B	B
0	A	B	C	C

C.

	Steps & array content			
Array Index	1	2	3	4
3				
2			C	C
1		B	B	B
0	A	A	A	

D.

	Steps & array content			
Array Index	1	2	3	4
3				
2				
1		B		C
0	A	A	B	B

11. Consider the following operation performed on **an array stack of size 5**, which a character will be the input elements. Index `top` is `-1` when a stack is empty :

```
1. push('a');
2. push('z');
3. pop();
4. push('b');
5. push('k');
6. push('m');
7. pop();
8. pop();
9. push('v');
```

Choose the **TRUE** statement(s) based on the above operation.

i. After the completion of all operations, the number of elements present on the stack is 2.

ii. After line 1 - line 6 operations, the number of elements present on the stack is 4.

iii. After completion of all operations, the top element will be 'v' at index 2.

iv. After completion of all operations, the bottom element will be 'a' at index 1.

A. i, ii and iii B. ii and iii only C. ii, iii and iv D. iii and iv only

12. Evaluate the postfix expression 4 5 1 * + 9 -, which of the following stack contents does **NOT HAPPEN** in the stack?

Note: The last value (**put in bold**) is the top of the stack

A. 4, 5, **1** B. 4, **5** C. 1, 9, **0** D. 9, **9**

13. Consider situation implementation of unsorted singly linked list. Which of the following operation can be implemented in $O(1)$ time?

A. Insertion new node at the front.

B. Insertion new node in the middle.

C. Deletion of the last node.

D. Searching for a node in the list.

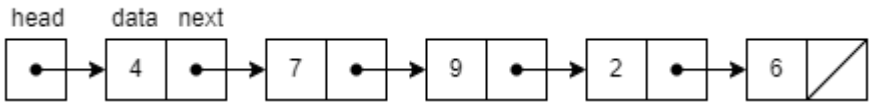
14. Which linked list type enables traversal in both directions?

A. Singly linked list.
B. Linear linked list.
C. Doubly linked list.
D. Circular linked list.

15. Consider situation implementation of unsorted singly linked list. Which of the following operation **TRUE** about the implementation of linked list in $O(n)$ time?
    i. Insertion new node at the front.
    ii. Insertion new node in the middle.
    iii. Deletion a front node in the list.
    iv. Deletion a node in the middle of the list.

A. i, ii B. i, iii C. iii, iv D. ii, iv

16. Consider the following singly linked list:



What is the output of the following C++ code segment?

```
Node *temp = head;
while (temp->next) {
      temp = temp->next;
      if (temp->next)
            cout << temp->data << " ";
}
```

A. 4 7 9 2 6 B. 7 9 2 6 C. 4 7 9 2 D. 7 9 2

17. Which of the following code segments is **CORRECT** for declaration of binary tree node?

A.
```
class Node
{
public:
    Node *children[2];
};
```

B.
```
class Node
{
public:
    Node *children[10];
};
```

C.
```
class Node
{
public:
    Node *child;
    Node *parent;
};
```

D.
```
class Node
{
public:
    Node *left;
    Node *right;
    Node *middle;
};
```

18. Consider the function **`isTerminalNode()`** as follows. This function is meant to determine whether the node ***n*** is a terminal node. What should you write in the blank lines to complete the function?

```
bool isTerminalNode(Node *n){
    return _____________________________;
}
```

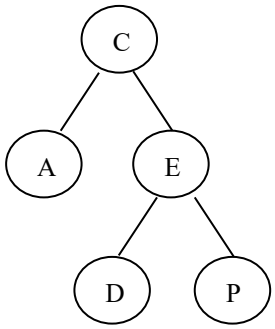
A. `(n->left==NULL)`

B. `(n->right==NULL)`

C. `(n == NULL)`

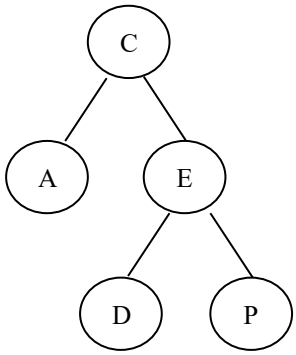
D. `(n->left==NULL && n->right==NULL)`

19. Given the following binary tree. What is the right order of visiting each node if an post-order traversal is performed on the tree?



A. C-A-D-E-P
B. A-D-P-E-C
C. C-E-A-D-P
D. P-D-E-A-C

20. Consider the following binary tree and the function `print()` in the given snippet code below:




```
void print(Node *n){

    if (n==NULL) return;
    print(n->right);
    cout << n->getData();

}
```

Assume a pointer named **`root`** is pointing to the root node (i.e. node C). What would be printed by the statement, **`print(root)`**;

A. C
B. PDECA
C. CEP
D. PEC

PART B – STRUCTURE QUESTIONS **[80 MARKS]**

Part B consists of 5 structured questions. Answer all questions in the answer booklet. The marks for each part of the question are as indicated.

Question 1 **[15 MARKS]**

Consider the array of integers denoted as **NOM**, consisting of 10 elements in descending order, as illustrated in Figure 1-1.

index	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
NOM	85	74	70	63	57	52	41	30	24	15

Figure 1-1: NOM array in descending order.

a) What is the number of steps and the time complexity when searching for **key = 14** on **NOM** array in figure above using Sequential Search technique?

(2 marks)

b) What is the search key and the time complexity for the best-case if using Sequential Search technique to search on the **NOM** array?

(2 marks)

c) Perform Binary Search for searching **key = 14** on the **NOM** array in Figure 1-1. Show the tracing of your search using variables **left**, **right**, **middle**, **NOM[middle]** and **found** using the table format below.

left	**right**	**middle**	**NOM[middle]**	**found**

(4 marks)

d) Based on your answer in Question 1 c), what is the number of steps and the time complexity when using Binary Search technique?

(2 marks)

e) Perform Binary Search for searching **key = 70** on the **NOM** array in Figure 1-1. Show the tracing of your search using variables **left**, **right**, **middle**, **NOM[middle]** and **found** using the table format below.

left	right	middle	NOM[middle]	found

(3 marks)

f) Discuss the efficiency of the Sequential Search and the Binary Search for the best-case and the worst-case.

(2 marks)

Question 2 **[15 MARKS]**

a) Given the current state of the **queue (linear array implementation)** as Figure 2-1.

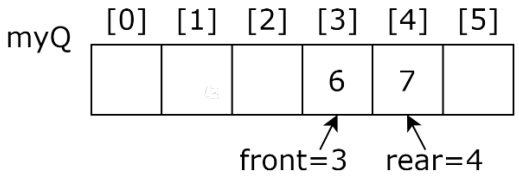


Figure 2-1: myQ linear array implementation

i. How many elements are currently stored in the queue that can be dequeued? List them.

(1 mark)

ii. Update the value of **front**, **rear**, **isFull/isEmpty** and operation **success/fail** for the following sequence of operations.

Operation: `myQ.enqueue(20)`

Operation: `myQ.dequeue()`

Operation: `myQ.enqueue(10)`

Answer using the table format below.

operation	status	Value of front	Value of rear	Operation success or fail?
`myQ.enqueue(20)`	isFull: t/f?			
`myQ.dequeue()`	isEmpty: t/f?			
`myQ.enqueue(10)`	isFull: t/f?			

(4 marks)

b) Given the current state of the **queue (circular array implementation)** as Figure 2-2.

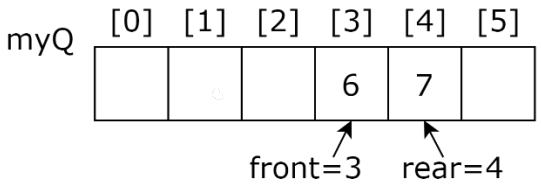


Figure 2-2: myQ circular array implementation

i. Update the value of **front**, **rear**, **count**, **isFull/isEmpty** and operation **success/fail** for the following sequence of operations:

Operation: `myQ.dequeue()`

Operation: `myQ.enqueue(20)`

Operation: `myQ.enqueue(10)`

Answer using the table format below.

operation	status	Value of front	Value of rear	Value of count	Operation success or fail?
`myQ.dequeue()`	isEmpty: _t/f?__				
`myQ.enqueue(20)`	isFull: _t/f?__				
`myQ.enqueue(10)`	isFull: _t/f?__				

(4 marks)

c) Given the current state of **myQ (linear link-list implementation)** as Figure 2-2.

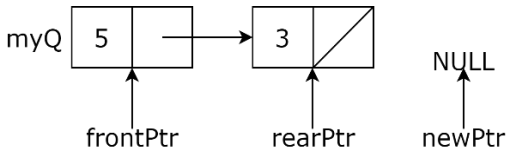


Figure 2-2: myQ linear link-list implementation

i. Redraw Figure 2-2 to show how the queue looks after performing `myQ.dequeue()`.

(2 marks)

ii. Based on its state in (i), redraw the queue to show its new state after performing the operation `myQ.dequeue()`.

(2 marks)

iii. Based on its state in (ii), redraw the queue to show its new state after the operation `myQ.enqueue(33)`.

(2 marks)

Question 3 **[15 MARKS]**

a) Evaluate the following infix expression by providing a step-by-step solution, clearly indicating the order of corresponding operation:

(5 + 3) * 2 - 6 / 2

Note: Use format: $\underbrace{a + b}_{ab}$

(4 marks)

b) Figure 3-1 shows the stack after executing `push()` four (4) numbers onto the stack:
`push(5), push(10), push(15), push(25).`

The last value (**put in bold and underlined**) is the top of the stack.

5	10	15	**<u>25</u>**

stack

Figure 3-1: Stack after inserting 4 numbers.

Given the following codes in Program 1 that uses `pop()` and `isEmpty()` operations over the stack (in Figure 3-1).

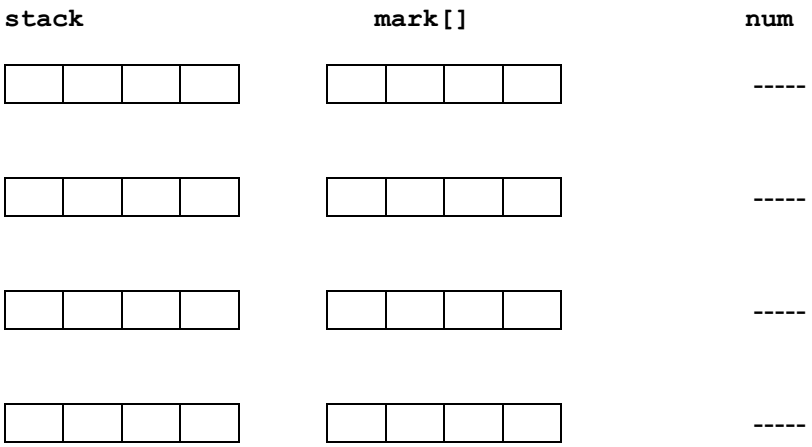
Show the content of the **stack**, the array **`mark`**, and the **`num`** values, during the execution of the **`while`** loop in Program 1.

```
  //Program 1
{
    int mark[4];
    int i = 0, num;

    while (!stack.isEmpty()) {
        mark[i] = stack.pop();

        if (i == 0)
            num = mark[i];
        else if (mark[i] < num)
            num = mark[i];
        i++;
    }
}
```

Answer using the format below.



(6 marks)

c) Using stack operations, show step-by-step conversion of the following infix expression to postfix notation.

P * Q – (R / S) + X

Answer using the table format below.

Read ch - Infix	Stack	Postfix
P	#	

(5 marks)

Question 4 **[20 MARKS]**

a) Figure 4-1 shows the initial linked list. Answer the following questions using **ONLY** the pointers and/ or variables given in the figure and code segment. Adding new pointers and/ or variables is **NOT** permitted.

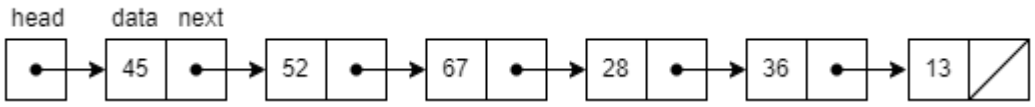


Figure 4-1: Initial linked list

i. Complete the code segment below to insert a new node after the second node in the initial list. Figure 4-2 illustrates the resulting list after the node insertion process.

```
Node *n = new Node;
n->data = 99;
//Write the code for the insertion process
```

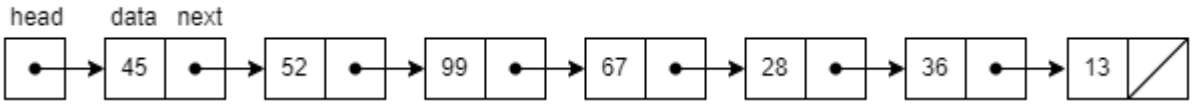


Figure 4-2: List after the insertion process

(4.5 marks)

ii. Complete the code segment below to delete the fifth node in the list produced by (a) (see Figure 4-2). Figure 4-3 illustrates the resulting list after the node deletion process.

```
Node *t = head;
//Write the code for the deletion process
```

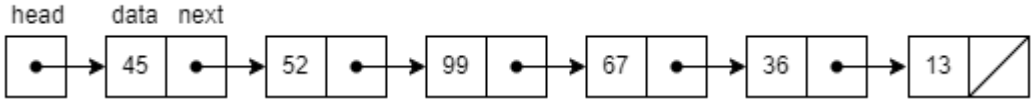


Figure 4-3: List after the deletion process

(6.5 marks)

iii. Complete the code segment below to display all nodes with odd numbers in the list produced by (b) (see Figure 4-3). Figure 4-4 shows the expected output that will be displayed on the screen.

```
Node *t = head;
//Write the code for the process of displaying nodes
```

```
45 99 67 13
```

Figure 4-4: Expected output

(4 marks)

b) **Figure 4-5** shows a circular doubly linked list with five (5) nodes and two (2) pointer variables (**head** and **temp**). Answer each of the **INDEPENDENT** questions b(i) to b(iii).

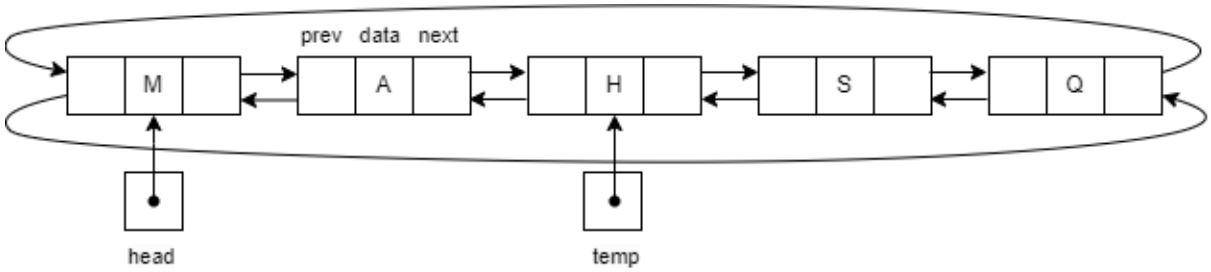


Figure 4-5: Circular doubly linked list

i. Redraw the linked list diagram in **Figure 4-5** to show the result obtained after executing the following code segment:

```
Node *nnode = new Node;
nnode->data = 'D';
head->prev->prev = nnode;
nnode->next = head->prev;
temp->next->next = nnode;
nnode->prev = temp->next;
```

(2 marks)

ii. Redraw the linked list diagram in **Figure 4-5** to show the result obtained after executing the following code segment:

```
Node *nnode = new Node;
nnode->data = 'Z';
nnode->prev = temp->prev;
temp->prev->next = nnode;
nnode->next = temp;
temp->prev = nnode;
```

(2 marks)

iii. Based on the linked list diagram provided in **Figure 4-5**, what output will be displayed if the following code segment is executed?

```
cout << temp->data << " ";
while (temp != head) {
   cout << temp->data << " ";
   temp = temp->prev;
}
```

(1 mark)

Question 5 **[15 MARKS]**

a) Using an appropriate example, explain the following terminologies about Trees.

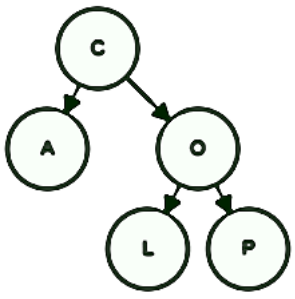
i. Full Tree

ii. Binary Search Tree

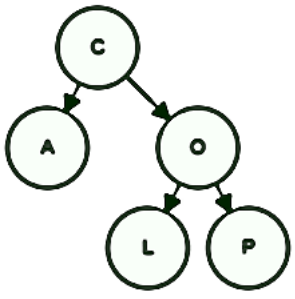
(4 marks)

b) Consider that each of the following problems are based on Binary Search Tree (BST). Each question is not related to each other. Given the initial state of a tree as shown in the question i, ii, and iii. Redraw the tree diagram that shows the update state of the tree after performing the operation stated in the question.

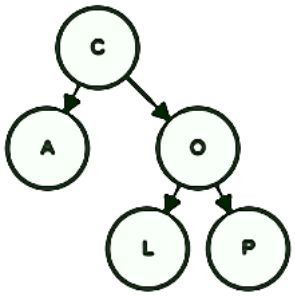
i. Insert a new node with data 'D'



ii. Delete the node 'O'



iii. Delete the node 'C'



(6 marks)

c) Given code that declare the node class for a tree as follows:

```
class Node
{
private: char data;

public:
    Node *parent;
    Node *left;
    Node *right;

    Node(char data){
        this->data = data;
        parent = left = right = NULL;
    }

    void insert(char data);
    void delete(char data);

    bool isInternal() const;
    int height() const;
};
```

Complete the code for the methods below:

i. `isInternal()`. This method returns the value of true if the node is an internal node.

(2 marks)

ii. `height()`. This method returns the height of the node in the tree.

(3 marks)