SULIT

FACULTY OF COMPUTING
UTM Johor Bahru

UNIVERSITI TEKNOLOGI MALAYSIA
FINAL EXAMINATION SEMESTER I, 2022 / 2023

SUBJECT CODE	**:**	**SECJ2013**
SUBJECT NAME	**:**	**DATA STRUCTURES AND ALGORITHMS**
YEAR / COURSE	**:**	**2SECB, 2SECP, 2SECJ, 2SECR, 2 SECV, MJIT**
TIME	**:**	**3 Hours**
DATE/DAY	**:**	
VENUES	:	

INSTRUCTIONS TO THE STUDENTS:

This examination book consists of 2 parts:

Part A:	20 Objective Questions	20 marks
Part B:	5 Structured Questions	80 marks
	Question 1 (Searching)	– 15 marks
	Question 2 (Linked List)	– 20 marks
	Question 3 (Stack)	– 15 marks
	Question 4 (Queue)	– 15 marks
	Question 5 (Tree)	– 15 marks

ANSWER ALL QUESTIONS IN THE ANSWER BOOKLET PROVIDED.

Name	
Identity card / Matric Number	
Name of Lecturer	
Subject Code and Section	

This examination book consists of **19** printed pages excluding this page.

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 test booklet.

1. What is the worst-case runtime complexity of searching for an item in a binary search?

   A. O(1)
   B. O ($\log_2$ n)
   C. O (n)
   D. O (n^2)

2. Given an array A = { 4,6 ,17, 28, 39} and key = 28. By using the Binary Search, how many iterations are required until the element is found?

   A. 5
   B. 2
   C. 3
   D. 4

3. Given an array A = {43, 67, 88, 90, 94, 99, 100} and key = 99. By using Binary Search, what are the mid values (corresponding array elements) generated in the first and second iterations?

   A. 90 and 100
   B. 89 and 94
   C. 90 and 99
   D. 94 and 99

4. Based on **Figure 1** below, which algorithm is suitable to perform the Binary Search when key = 20?

[0]	**[1]**	**[2]**	**[3]**	**[4]**	**[5]**	**[6]**	**[7]**	**[8]**	**[9]**
11	13	14	16	20	23	26	27	29	30

Figure 1

A.

```
if ( array [MIDDLE] == search_key)
{
       index = MIDDLE;
       found = true;

}
```

B.

```
left = MIDDLE + 1;
```

C.

```
else if (array [MIDDLE] > search_key)
        RIGHT = MIDDLE - 1;
```

D.

```
if ( array [MIDDLE] == search_key)
{
       index = MIDDLE + 1;
       found = true;

}
```

5. When creating the linear queue, what are the initial values given to the front and back pointers of the linear queue?

   A. `front=0, back=1`
   B. `front=1, back=0`
   C. `front =0, back=-1`
   D. `front=-1, back= 0`

6. A queue, if implemented using an array of size `MAX_SIZE`, the queue gets full when____________

   A. `back= MAX_SIZE`
   B. `back= MAX_SIZE-1`
   C. `front= back+1`
   D. `front= (back+ 1)%MAX_SIZE`

7. Suppose a circular queue of capacity **(n-1)** elements is implemented with an array of **n** elements. Assume that the insertion and deletion operations are carried out using **`REAR`** and **`FRONT`** as array index variables, respectively. Initially **`REAR=FRONT=0`**. Which of the following conditions that will detect queue full and queue empty?

   A. `Full: (REAR+1)% n == FRONT`
      `Empty: REAR==FRONT`
   B. `Full: (REAR+1)% n == FRONT`
      `Empty: (FRONT+1) % n == REAR`
   C. `Full: REAR==FRONT`
      `Empty: (REAR+1) % n==FRONT`
   D. `Full: (FRONT+1)% n==REAR`
      `Empty: REAR==FRONT`

8. **Figure 2** below shows a circular linked list implementation of a queue data structure. How many changes of **backPtr->next** are required if we want to remove all items except the last item in the queue (the item that holds 7)?

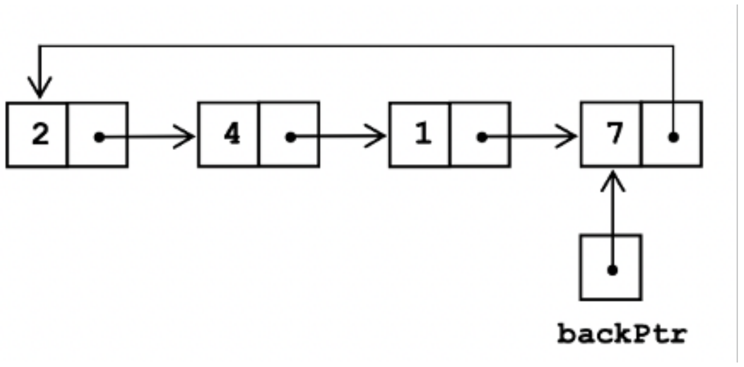


Figure 2

   A. 3 changes
   B. 4 changes
   C. 2 changes
   D. 1 change

9. What operation is **NOT** needed if the stack was implemented using a linked list?

   A. `isEmpty()`
   B. `isFull()`
   C. `stackTop()`
   D. `push()`

10. Which one of the following is **NOT** the application of the array-based stack data structure?

    A. String reversal
    B. Recursion
    C. Tracking of local variables at run time
    D. None of the above

11. Given the stack implementation below, what would be the maximum value of the **TOP** that does not cause overflow of the stack?

```
#define SIZE 11
struct STACK
{
     int arr[SIZE];
     int TOP = -1;
}
```

    A. 11
    B. 10
    C. -1
    D. 0

12. Consider the algorithm for determining whether a sequence of parentheses is balanced. What is the maximum number of parentheses that appear on the stack **at the same time** when the algorithm analyzes: (( ) (( )) (( ))) ?

    A. 1
    B. 2
    C. 3
    D. 4 or more

13. What is the optimal time complexity to count the number of nodes in a linked list?

    A. O(n)
    B. O(1)
    C. O(log n)
    D. O(n^2)

14. Which of the statements below describe the **DISADVANTAGE** of linked list compared to array?
    A. Deletion of a middle item leaves an empty space.
    B. A linked-list size is virtually unlimited.
    C. Requires space for pointers in each cell.
    D. Need to shift elements after the insertion or deletion.

15. By referring to **Figure 3**, which of the code segments below will add a new node pointed by temp and contain value 80 at the BEGINNING of the list?

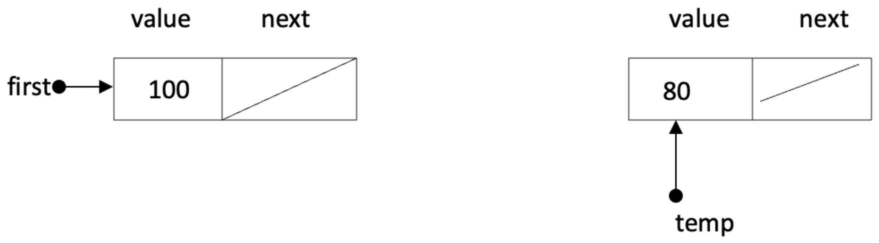


Figure 3

A. `Node* temp = new Node; temp->value = 80; temp->next = NULL; first = temp;`
B. `Node* temp = new Node; temp->value = 80; temp->next = first; first = temp;`
C. `Node* temp = new Node; first->value = 80; temp->next = first;`
D. `first = new Node; first->value = 80; first->next = first;`

16. The following code segment in **Figure 4** BUILDS a linked list with the numbers 22 and 42 as its components. What should be the statement in the missing part?

```
Struct NodeType
{
     int data;
     NodeType* link;

};

NodeType* p;
NodeType* q;

       p = new NodeType;
       p->data = 22;
       q = new NodeType;
       q -> data = 42;
       ..................................   // < -- Statement is missing here
       q->link = NULL;
```

Figure 4

A. p = q;
B. p->link = new NodeType;
C. p->link = q;
D. p->link = q->link;

17. Which of the following options is **FALSE** about the Binary Search tree?

A. The value of the left child should be less than the root node
B. The value of the right child should be greater than the root node.
C. The left and right sub trees should also be a binary search tree
D. When inserting or searching for an element, the key of each visited node has to be compared with the key of the element to be inserted or found.

18. Based on Figure 5, which of the following is the **CORRECT** output for postorder traversal of the tree in Figure below?

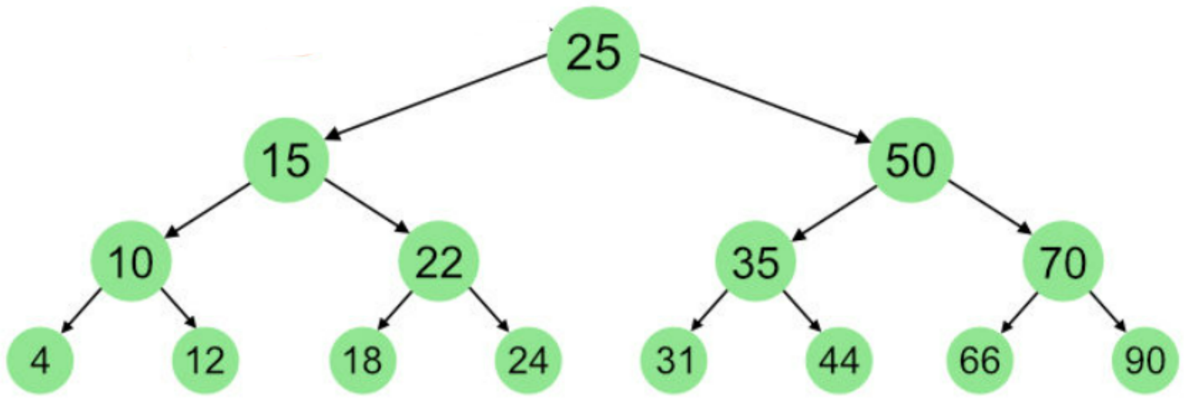


Figure 5

A. 4,12,10,15,22,18,24,31,35,44,50,66,70,90,25
B. 4,10,12,15,18,22,24,25,31,35,44,50,66,70,90
C. 4,12,10,18,24,22,15,31,44,35,66,90,70,50,25
D. 25,15,10,4,12,22,18,24,50,35,31,44,70,66,90

19. Based on **Figure 6**, which of the following is the **CORRECT** algebraic expression represented by the expression tree in Figure below?

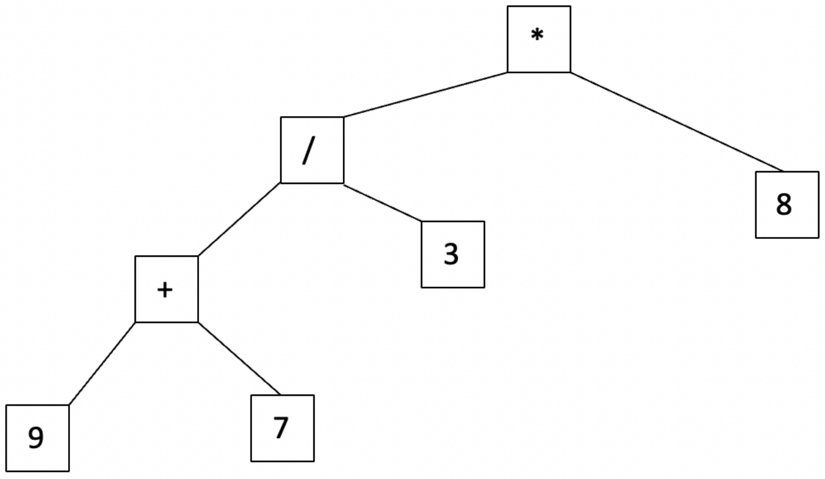


Figure 6

A. 9 + 7 / 3 * 8
B. (9 + 7) / (3 * 8)
C. 9 + (7 / 3) * 8
D. ((9 + 7) / 3) * 8

20. Which of the following options is **FALSE** about the full binary tree with height **h**?

A. The minimum number of nodes in the full binary tree is 2^h
B. You cannot add nodes to a full binary tree without increasing its height.
C. The maximum number of nodes that a binary tree of height h can have is 2^h -1
D. A full binary tree of height h >= 0 has 2^h-1 nodes

PART B - STRUCTURED QUESTIONS

[80 MARKS]

Part B consists of 5 structured questions. Answer all questions in the space provided. The marks for each part of the question is as indicated.

Question 1 **[15 MARKS]**

a) Answer the following questions based on the sorted array named **BMI** shown in **Figure B1.1**.The array contains BMI (Body Mass Index) values for seven (7) cancer patient.

	[0]	[1]	[2]	[3]	[4]	[5]	[6]
BMI	**36**	**39**	**42**	**46**	**49**	**50**	**54**

Figure B1.1: BMI array

(i) Perform **improved sequential search** using **SortedSeqSearch()** function for searching Bmi value = **46**. Show tracing of your search using variables i**ndex**, **p**,**search_key**, **Bmi[p]**, and **found** as shown in the **Table B1.1** format below: [4 marks]

Table B1.1

index	**p**	**search_key**	**Bmi[p]**	**found**

(ii) Perform **binary search** for searching **Bmi** value = **57**. Show tracing of your search using variables **index**, **LEFT**, **RIGHT**, **MIDDLE**, **search_key**, **Bmi[MIDDLE]**, and **found** as shown in the Table B1.2 format below:

[4 marks]

Table B1.2

index	**LEFT**	**RIGHT**	**MIDDLE**	**Bmi[MIDDLE]**	**search_key**	**found**
			▨	▨		

(iii) Fill in the following Table B1.3 with the number of steps and the complexity time for searching key = **39**, **46** and **57**. Based on the results, compare and discuss the efficiency of binary search and linear search (on sorted data) algorithms in the three searching case.

[7 marks]

Table B1.3

Searching Comparisons				
Search Key	**Linear Search**		**Binary Search**	
	Number of Steps	**Complexity**	**Number of Steps**	**Complexity**
39				
46				
57				
Discussion				

Question 2 **[20 MARKS]**

a) **Draw the modified linked list** based on the initial state of the circular linked list in **Figure 7** after each of the following code segments (i) to (v) has been executed. Code segments are executed in a specific order, so (i) will be executed first, followed by (ii), and so on. Each code segment is dependent on the code segment before it. After each code segment is executed, show the positions of the pointer **`head`**, **`tail`**, and any other pointer(s). Finally, **specify the operations** carried out by each code segment.

[10 Marks]

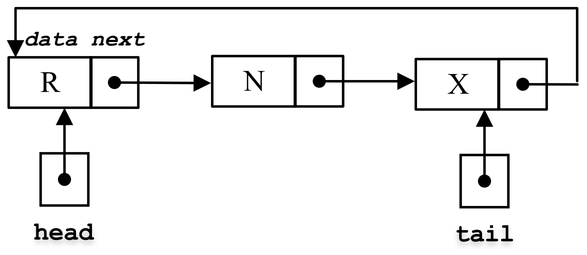


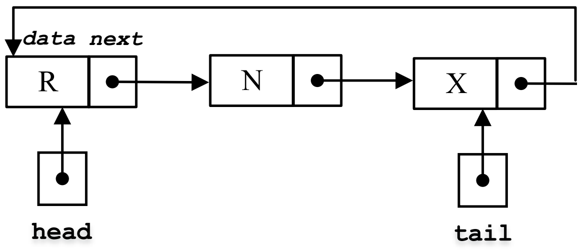
Figure 7: Circular linked list's initial state

Example:

```
//Example Code Segment
Node *tmp = head;

while (tmp != tail) {
  cout << tmp->data;
  if (tmp != tail)
    cout << "->";
  tmp = tmp->next;
}
cout << tmp->data << endl;
```

Modified linked list: ***The linked list remains the same as in Figure** 7.*



The operation performed: ***Displays all nodes in the list**.*

(i) [2.5 marks]

```
//Code Segment 1
Node *n1 = new Node;
n1->data = 'T';
int nd = 1;
tmp = head;
while (nd++ != 2 && tmp != tail) {
  tmp = tmp->next;
}
tmp->next = n1;
n1->next = tail;
```

(ii) [2.5 marks]

```
//Code Segment 2
tmp = head;
while (tmp->next != tail)
  tmp = tmp->next;

tail = tmp;
tmp = tmp->next;
tmp->next = NULL;
delete tmp;
tail->next = head;
```

(iii) [2.5 marks]

```
//Code Segment 3
Node *n2 = new Node;
n2->data = 'F';
n2->next = head;
tail->next = n2;
tail = n2;
```

(iv) [2.5 marks]

```
//Code Segment 4
tmp = head;
head = head->next;
tmp->next = NULL;
delete tmp;
tail->next = head;
```

b) **Figure 8** depicts five nodes arranged as a doubly linked list, along with four pointer variables (**`tmp`**, **`head`**, **`mid`**, and **`tail`**). Answer all of the questions (i) and (ii). Use only the variables defined in **Figure 2** and do **NOT** define any new variables. Each question is dependent on the one before it.

[10 Marks]

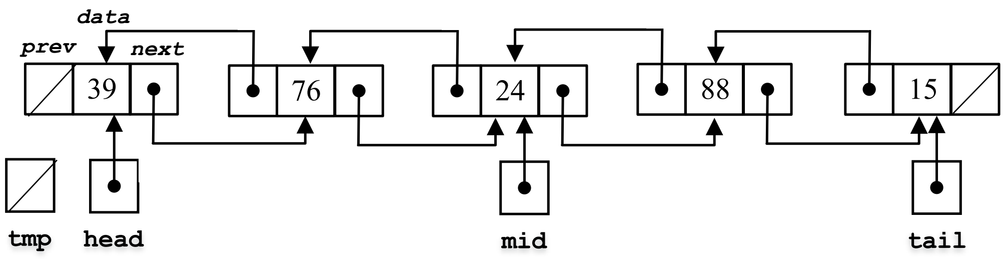


Figure 8: Five-node doubly linked list

(i) Write a C++ code segment to insert a new node, **nn** in **Figure 9**, between the fourth (88) and fifth (15) nodes in **Figure 8**. Assume the pointer variable, **nn** has been defined. **Figure 10** depicts the outcome of this task after the statements in the code segment have been executed. [5 marks]

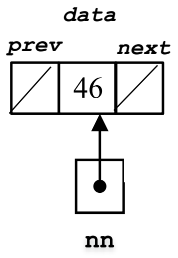


Figure 9: New node to be inserted

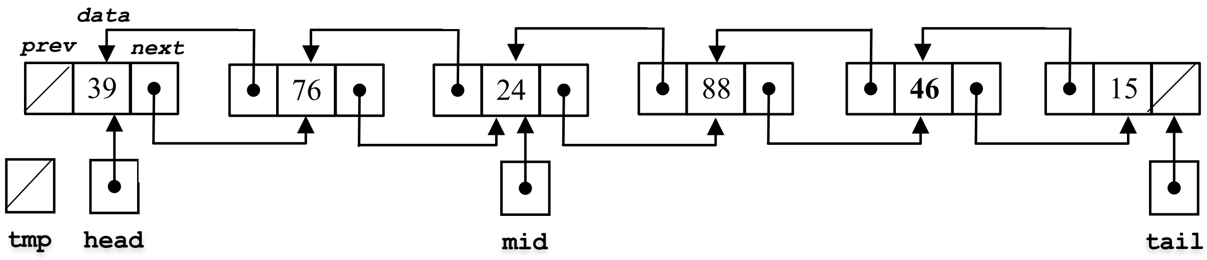


Figure 10: Current nodes in the list after the node is inserted at the fifth position in the list

(ii) Write a C++ code segment to delete the fourth node (88), and connect the third node (24) and fifth node (46) in **Figure 10**. Please release the memory allotted for the fourth node. **Figure 11** depicts the outcome of this task after the statements in the code segment have been executed. [5 marks]

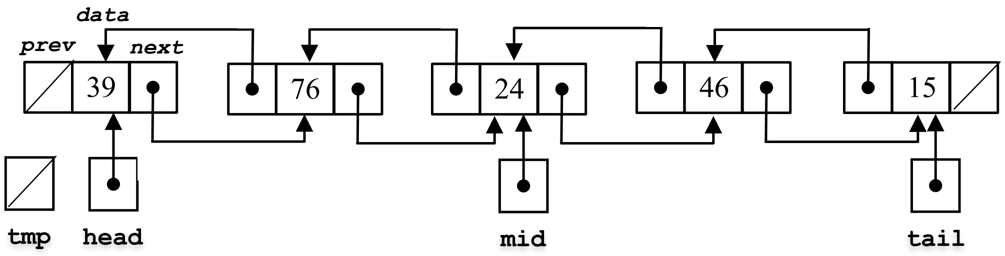


Figure 11: Current nodes in the list after the deletion of the node from the list

Question 3 **[15 MARKS]**

a) Consider the following **Figure 12** is source codes in **Program Segment 1** that will be implemented on stack using array. The size of the stack is 5.

```
1   Stack s;
2   string s1, s2, s3;
3   :
4   :
5   s1 = "Hello";
6   s2 = "Good Day";
7   s3 = "Bye!";
8   s.push(s1);
9   s.push(s2);
10  s.push(s2+s3);
11  s.push("Arigato");
12  cout<<displayStacks(s);
13  s.pop();
14  s.pop();
15  s.push("Bye Bye!");
16  s.pop();
17  s.push (s3 + s2);
```

Figure 12: Program Segment 1 Stack source codes

i) Show the content of the stack after each implementation of the sources codes based on the situation below. Assume the current content of the stack before statement Line 5 is executed as shown in first stack in the **Table B3.1** below. [8 marks]

Table B3.1

Element of Array								
[4]								
[3]								
[2]								
[1]								
[0]	Hai							
	Current Content Before Line 5	Content After Line 5-8 are Executed	Content After Line 10	Content After Line 11-12 are Executed	Content After Line 13 is executed	Content After Line 14 is executed	Content After Line 15 – 16 are executed	Content After Line 17 is executed

b) Evaluate the following postfix expression using stack concept :

`8 64 % 15 * 2 -` [7 marks]

Postfix	Ch	Op	Oprn1	Oprn2	Result	Stack
8 64 % 15 * 2 -						
64 % 15 * 2 -						
% 15 * 2 -						
15 * 2 -						
* 2 -						
2 -						
-						

Question 4 **[15 MARKS]**

a. The Faculty of IT is having Fahmi, a developer, help them develop a course registration system. A maximum of 50 students per section is requested by the faculty. A queue will be used by Fahmi as a data structure for storing data. Which would you recommend to Fahmi among queues implementing circular arrays and circular linked lists? Justify your answer by describing how it benefits to Fahmi.

[3 marks]

Description	**Mark**

b. Given Algorithm 1 – Queue implementing Circular Array. Answer the following questions.

```
1   // Algorithm 1 - Queue implementing circular array
2   class Queue {
3   public:
4          int front, rear, counter;
5          const int SIZE = 5;
6          char deletedItem;
7          int* items = new int[SIZE];
8   
9          Queue() { front = 0, counter = 0, rear = SIZE - 1; }
10  
11         bool isEmpty() { return bool(counter == 0); }
12  
13         bool isFull() { return bool(counter == SIZE); }
14  
15         void enQueue(char insertItem) {
16                cout << "\nTry insert " << insertItem;
17                if (isFull())
18                   cout << "\nCannot insert " << insertItem << ". Queue is full!";
19                else {
20                       _______________________
21                       _______________________
22                       _______________________
23                       cout << endl << insertItem << " insert successful!";
24                }
25         }
26         void deQueue() {
27                if (isEmpty())
28                       cout << "\nCannot remove item. Empty Queue!";
29                else {
30                       __________________________
31                       __________________________
32                       __________________________
33                       cout << endl << deletedItem << " delete successful!";
34                }
35         }
36  }
```

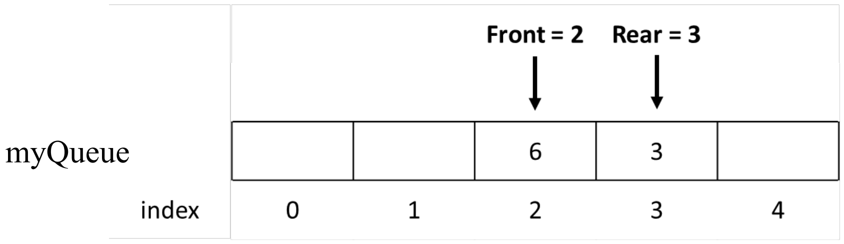
i. Complete the **enQueue** function when the insertion operation is triggered at lines 21 through 23.

[3 marks]

ii. Complete the **deQueue** function when the deletion operation is trigged at lines 32 through 34.

[3 marks]

c. Figure below shows a circular queue called myQueue, which fits maximum of 5 items. The current content of myQueue are as shown in the figure below. Given that **item1 = 6, item2 = 3, item3 = 5, and item4 = 2** are integer variables.



i. Redraw myQueue after **enQueue(item3)** and **enQueue(item4)** are executed. Label the correct location of the **front** and **rear**.

[2 marks]

ii. Redraw updated queue in (i) after **deQueue()** and **enQueue(item1 - item4)** are executed. Label the correct location of the **front** and **rear**.

[2 marks]

iii. After the updated queue in (ii). What is the output when the following pseudocode executed?

```
myQueue.deQueue()
myQueue.enQueue(item2 + item4)

item1 = myQueue.getFront()
item2 = myQueue.getRear()
cout << item1 << " " << item2
<< endl
```

[2 marks]

Question 5 **[15 MARKS]**

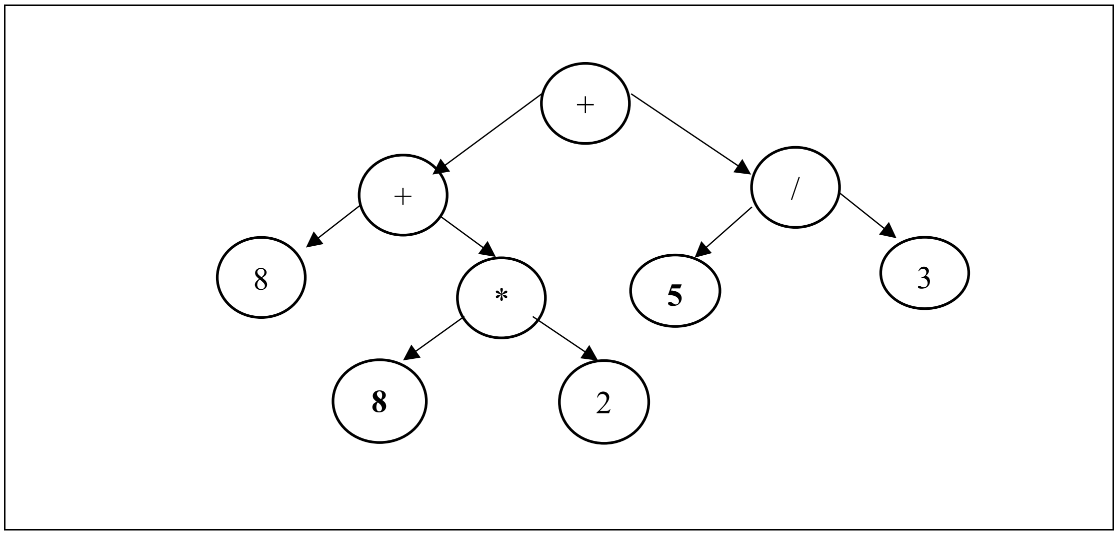


Figure 13

a) Based on **Figure 13**, answer the following questions:

   i. Give the order of node visited in a in-order traversal?

   ii. Give the order of node visited in a pre-order traversal?

   iii. Give the order of node visited in a post-order traversal?

[6 marks]

b) i. Draw the binary search tree based on the sequence of numbers as shown in **Figure 14**. The insertion starts from number `7` to `6` (left to right) in a sequential manner.

7	15	12	5	67	2	1	6

Figure 14. Sequence of numbers

[3 marks]

ii. State whether the binary search tree drawn in **b(i)** is full, complete or/and balanced.

[1 mark]

c) Given the following **Figure 15** binary search tree, answer the following questions:

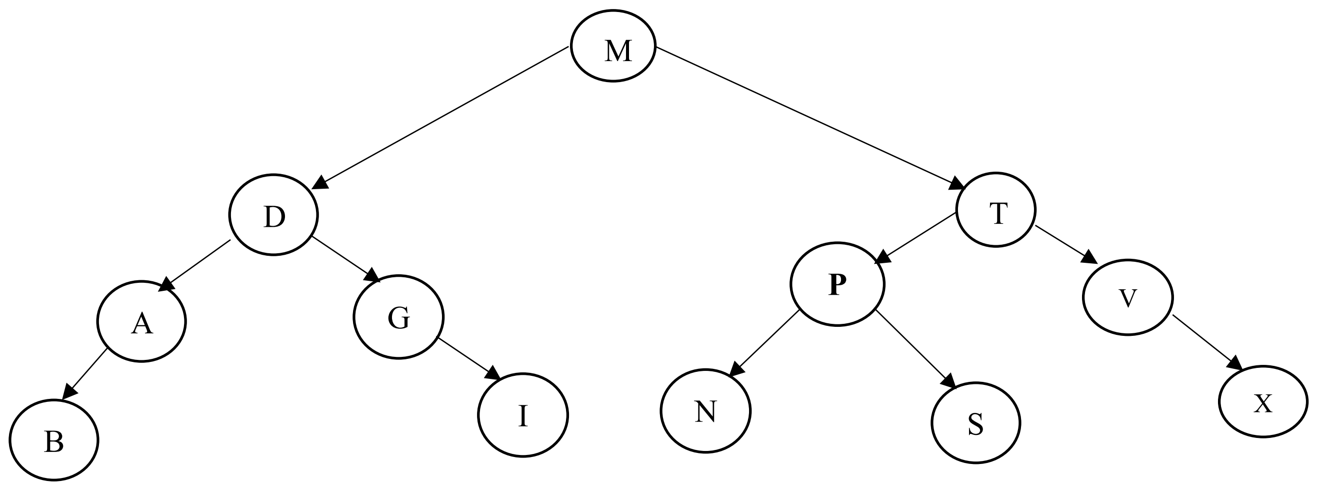


Figure 15

i) Height of the tree is : _____________ [1 mark]

ii) Level of node **I** is : _____________ [1 mark]

iii) Redraw the tree after node **M** has been deleted from the tree [1 mark]

iv) Referring to the original tree, redraw the tree after node **A** and **X** has been deleted consequently [2 marks]