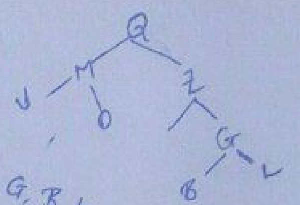
VMO



Z GBL

Pre : Q, M, V, O, Z, G, B, L
In : V, M, O, Q, Z, B, G, L

PART I: SHORT ANSWER AND OUTPUT ITEMS (25 pts)

Instruction: For the following questions, read the instructions carefully and provide proper and neat answers in space provided **ONLY**

1. Write the conditions for stack overflow and underflow in case of array based implementation of stack? **[1 pt.]**

→ If we want to add an element after the stack is full then it becomes overflow (it says overflow).
→ If we want to remove an element after the stack is empty, then it becomes under flow or it says under flow

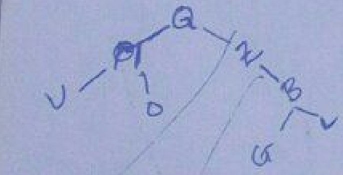
2. Consider the array elements: **6, 21, 35, 3, 6, 2, 13** and answer the following two questions.

I. If the array elements listed above are added into an empty stack, in the order given, what is the output obtained and their order when **pop( )** operation is applied to the stack until it becomes empty? **[1 pt.]**

13, 2, 6, 3, 35, 21, 6

II. If the array elements listed above are added into an empty queue, in the order given, what is the output obtained and their when **dequeue( )** operation is applied to a queue until it becomes empty? **[1 pt.]**

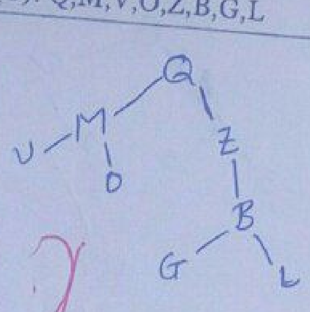
6, 21, 35 3, 6, 2, 13



Pr, Q, M, V, O, Z, B, G, L
In, V, M, O, Q, Z, G, B, L

5

3. Given the following two traversals of a tree **T**, construct the tree **T**? **[2 pts.]**
Inorder(T): V,M,O,Q,Z,G,B,L
Preorder(T): Q,M,V,O,Z,B,G,L

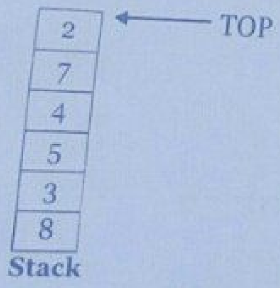


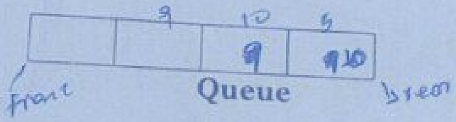
2

2. Write a C++ code segment to remove the front element from a circular queue implemented in array. Assume values of Front, Rear, array size, and queue size are given. **[2 pts.]**

0

3. Given a full stack with a size of 6 and an empty queue with a size of 4 bellow:



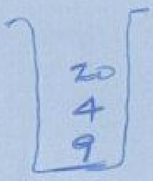


Given the following code segment which works on the above Stack and Queue:

```
int a,b;
enqueue(pop());
enqueue(pop());
a = dequeue();
b = pop();
push(a+b);
enqueue(a+b);
push(dequeue());
enqueue(10);
enqueue(pop());
push(20);
```

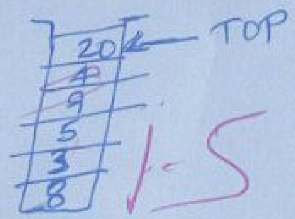
9 9

9



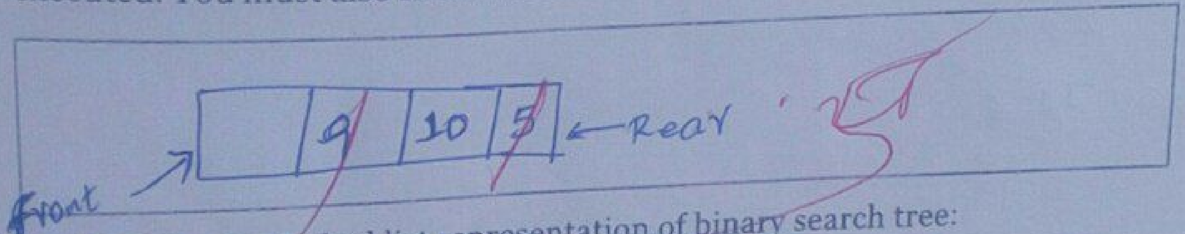
Answer the following two questions (I and II) based on the above givens.

I. Redraw and show what the stack looks like after the above code segment is executed. You must properly show where the 'Top' point is as well. **[2 pts.]**

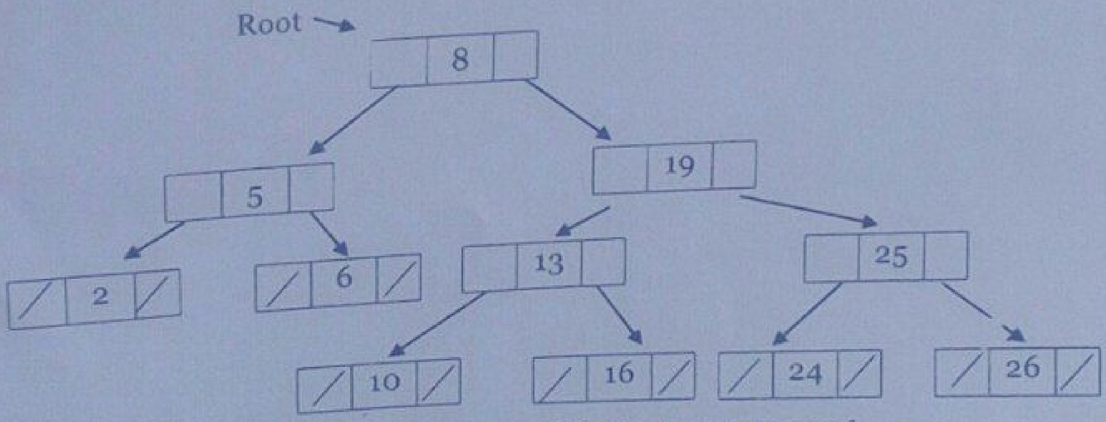


1.5

II. Redraw and show what the queue looks like after the above code segment is executed. You must also show where the 'Front' and 'Rear' points are. **[2 pts.]**



4. Given the following linked list representation of binary search tree:



Given the following structure based on which the tree is created:

```
struct bstree{
        int data;
        bstree *left;
        bstree *right;};
```

Based on the above two givens what is the output of the following code segment? **[2 pts.]**

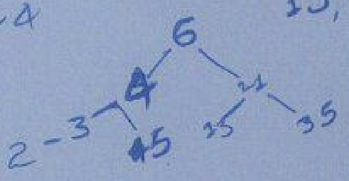
```
bstree *temp;
temp = root;
while(temp!=NULL)
 {
  if(temp->data % 2 == 0)
    {
      cout << temp->data;
      temp = temp->right;
    }
   else
    {
      cout << temp->data;
      temp = temp->left;
   }
```

8, 19, 13, 16, 10, 5, 6, 2, 25
26, 24

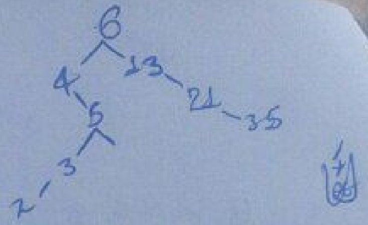
0

AB*CD/+ 21

5,4 13, 35



AB*CD/+



4. Convert the given infix expression into postfix using stack: A*B+C/D? Show all necessary steps pictorially.

[2 pts.]

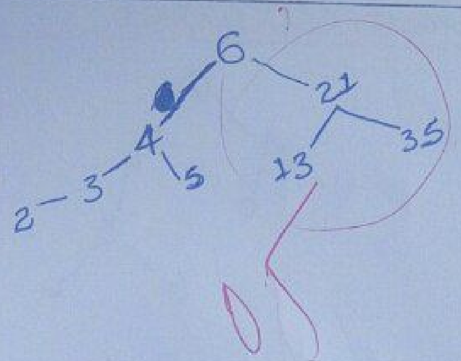
Symbol	stack	Post fix expression
A	empty	A
*	*	A
B	*	AB
+	*+	AB*
C	+	AB*C
/	+/	AB*C
D	empty	AB*CD/+

OR

AB*CD/+

5. Consider the list of integer elements: **6, 21, 35, 3, 2, 13, 5, 4** and construct a binary search tree rooted at node "**6**", in the order given, and answer the following six questions (I-VI) based on the binary search tree you have constructed.

I. Draw the binary search tree. **[1 pt.]**



25+5 = 75

II. How many number of leaves the Binary Search Tree could have? **[1 pt.]**

4

III. List all leaf nodes of the binary search tree? **[1 pt.]**

2, 5, 13, 35

IV. Post-order traversal of the binary search tree? **[1 pt.]**

2, 3, 5, 4, 13, 35, 21, 6

V. Pre –order traversal of the binary search tree? **[1 pt.]**

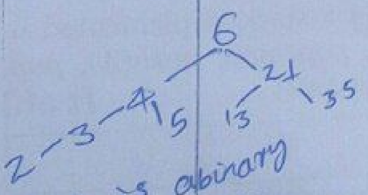
6, 4, 3, 2, 5, 21, 13, 35

VI. If node '**5**' has children, then clearly mention the left and right child? **[1 pt.]**

node '5' has not any child

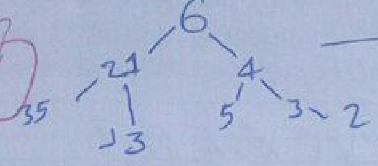
6. Let us suppose that we have a list of integers, that we wish to sort into **ascending** order. After having inserted the numbers into the binary search tree, in order to produce a sorted list of the number in ascending order, we need to traverse the tree in **inorder** traversal method. On the contrary, what if we want to produce a sorted list of the number in **descending** order from the same binary search tree, write the order in which the nodes will be visited? You can supplement your answer pictorially. **[2 pts.]**

let the integers be 2, 3, 4, 5, 6, 13, 21, 35



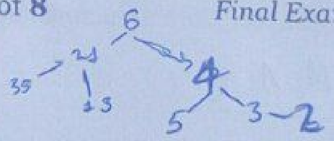
this is a binary tree which the given integers are sorted in ascending order.

0



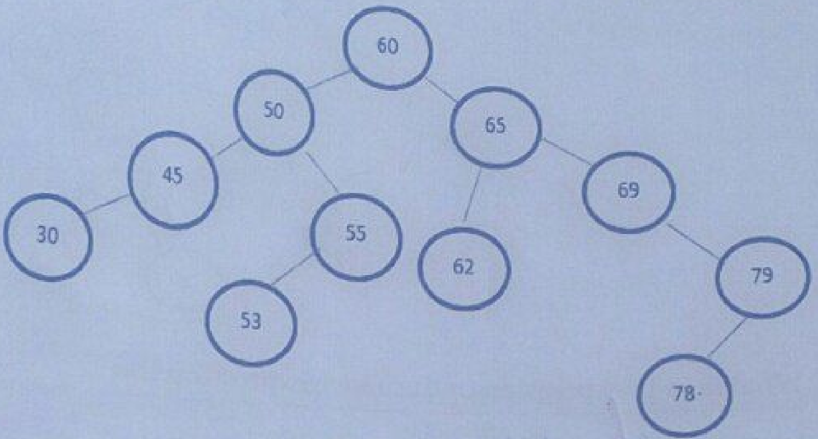
→ from this tree if we want to traverse the given integers in inorder travers method the result will be 35, 21, 13, 6, 5, 4, 3, 2 which are descending order. But this tree is not a binary search tree.

→ Because in the binary search tree the left children should be less than thier nodes as well as the right children



4 + 4 + 1 = 9

10. Consider the following tree and answer question four questions bellow.



I. What is the height of the above tree? **[1 pt.]**

4

II. Is the above tree a full binary tree? Why? **[1 pt.]**

Yes because ~~it has~~ each node has less or equal to two children. i.e. no one node has more than two children.

0

III. Traverse the above binary search tree in: - **[3 pts.]**

Postorder:

30, 45, 53, 55, 50, 62, 78, 79, 69, 65, 60

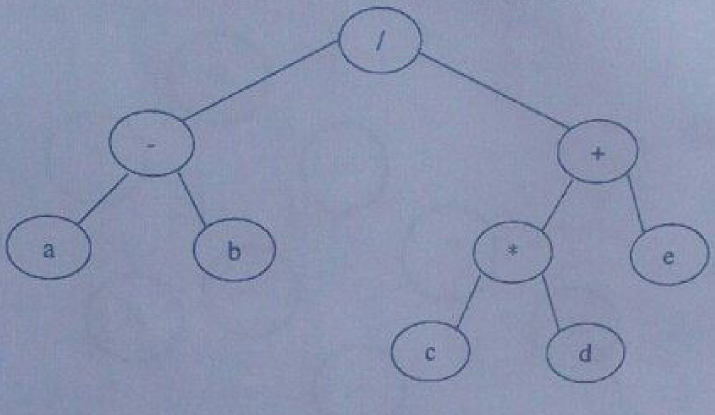
Preorder:

60, 50, 45, 30, 55, 53, 65, 62, 69, 79, 78

Inorder:

30, 45, 50, 53, 55, 60, ~~60~~, 62, ~~64~~ 65, 69, 78, 79

11. Consider the following expression tree for the expression:



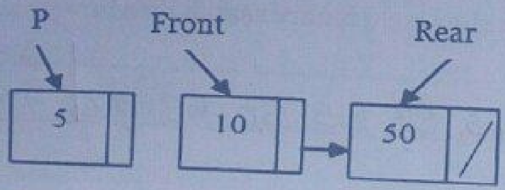
What is infix expression of the above expression tree? **[1 pt.]**

a-b/c*d+e

PART II: PROGRAMMING ITEMS (10 pts)

Instruction: For the following questions, read the instructions carefully and provide proper and neat answers in space provided **ONLY**

1. Given the following queue, write a C++ code segment which will perform **enqueue()** operation to insert an element pointed by **p** to this queue? **[2 pts.]**



```
Rear=-1;
front = -1;
void enqueue(struct queue * front=-1]
{
if(front=-1, rear=-1);
cout<<" the queue is full";
    else
    queue(++ front);
    queue();
```

5