Q1

Ans:

Infix

A * (B * C / D - E) + F

step 1 : Add a left parenthesis in stack and right parenthesis at end of infix exp.

step 2 :

Modified Infix exp in step 1	Stack	Postfix P
A	(	A
*	(*	A
(	(* (	A B
B	(* (	A B
*	(* (*	A B C
C	(* (*	A B C *
/	(* (/	A B C * D
D	(* (/	A B C * D /
-	(* (-	A B C * D / E
E	(* (-	A B C * D / E -
)	(*	A B C * D / E - *
+	(+	A B C * D / E - * F
F	(+	A B C * D / E - * F +
)	Empty	

Answer

Ans-2

```
struct node * enqueue(struct node * start, int item, int priority)
{
        struct node *nn,*temp,*prev;        prev = NULL;
        nn=(struct node *)malloc(sizeof(struct node));
        nn->info=item; nn->priority=priority; nn->next=NULL;
        if (start==NULL )
                start=nn;
        else
        {
                temp=start;
                while (temp!=NULL && temp->priority <= priority)
                {
                        prev=temp; temp=temp->next;
                }
                if (temp!=NULL)
                        nn->next=temp;
                if (prev!=NULL)
                        prev->next=nn;
                else
                        start=nn;
        }

        return start;
}
```

Q3.

Ans:

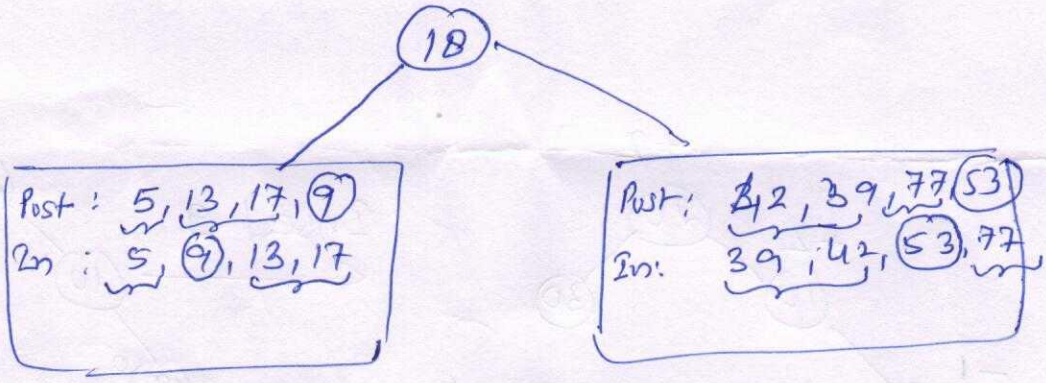
Post-order: 5, 13, 17, 9, 42, 39, 77, 53, 10 (root)

As tree is BST, therefore inorder will be

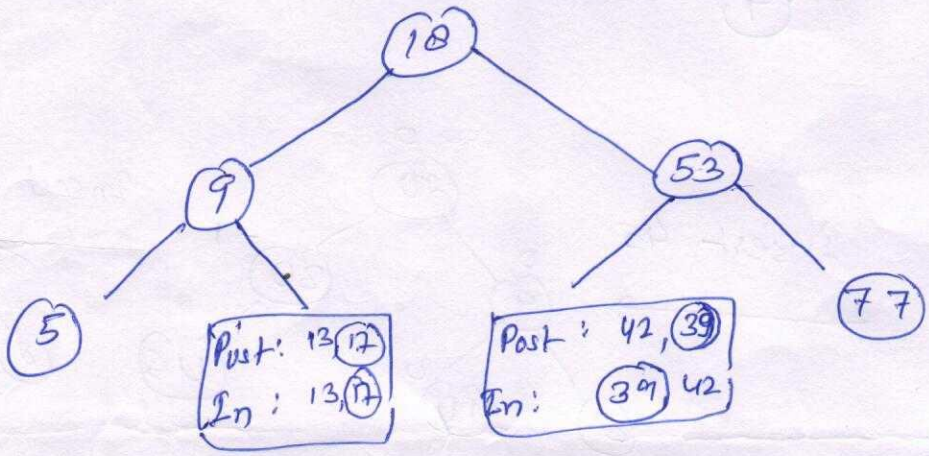
In-order: 5, 9, 13, 17, 10, 39, 42, 53, 77

(5, 9, 13, 17: left subtree; 10: root; 39, 42, 53, 77: right subtree)

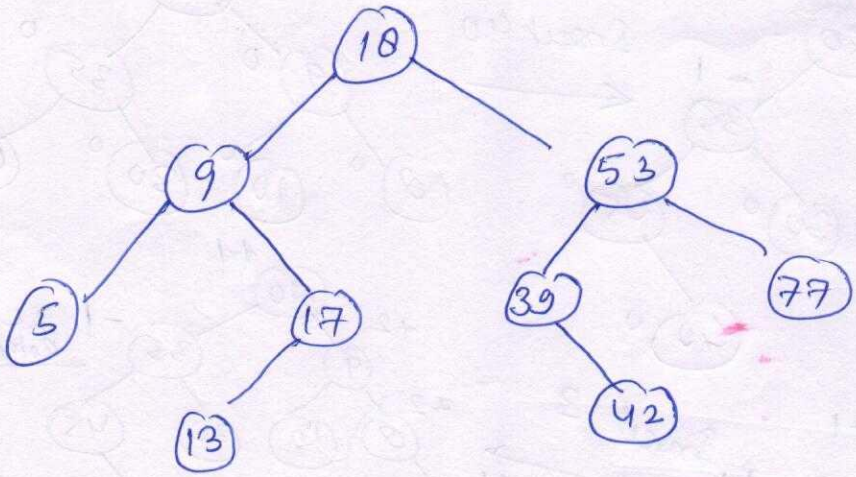
Step 1:



Step 2:



Step 3

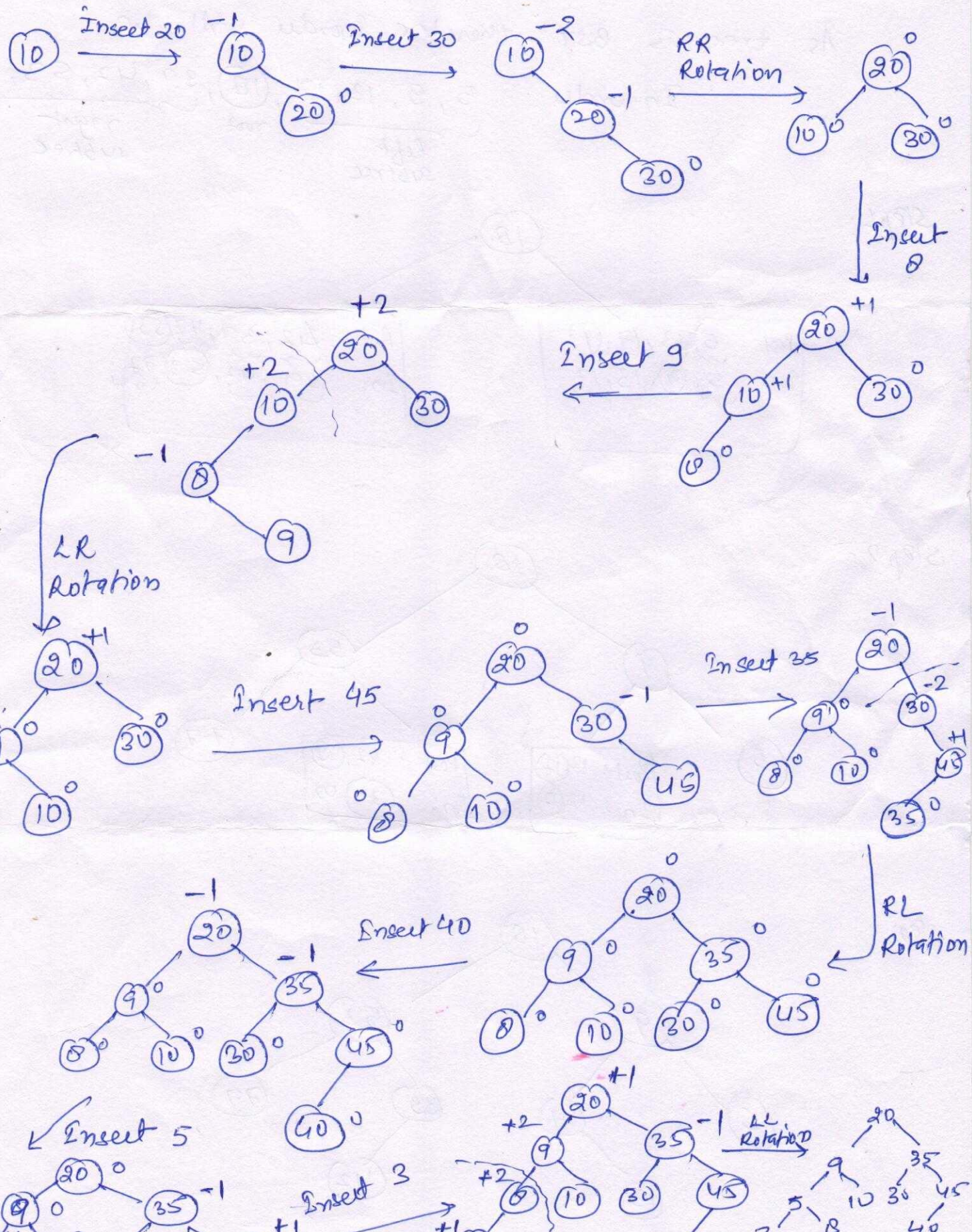


Ans 4

```
int countNonRec (struct node * root)
{
        STACK s;    s.top = -1;
        struct node *p; int c=0;
        p=root;
        do
        {
                while (p!=NULL)
                {
                        push(&s,p)
                        p=p->left;
                }
                p=pop(&s);
                c++;
                p=p->right;
        }while(p!=NULL || !empty(s));
        return c;
}
```

Q5

Ans: 10, 20, 30, 8, 9, 45, 35, 40, 5, 3



Delete 20

Scanned