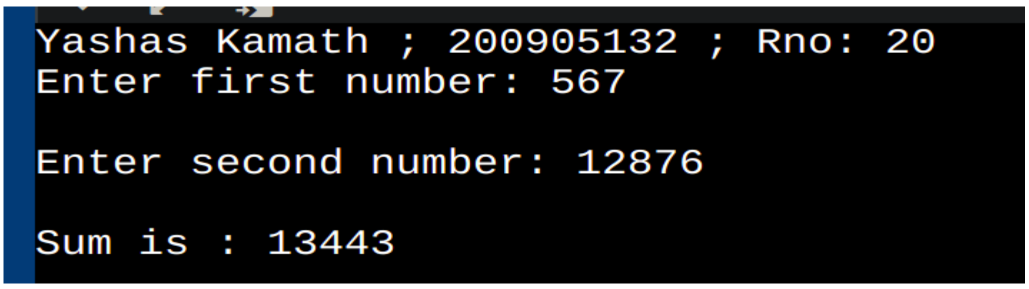
DSA LAB 8

1) Add two long positive integers represented using circular doubly linked list with header node.
Solution:

```
#include<stdio.h>
#include<stdlib.h>
typedef struct node{
int data;
struct node * next;
struct node * prev;
} * NODE;
void insertFront(NODE head, int val){
NODE first = head->next;
head->data++;
NODE n = (NODE)malloc(sizeof(struct node));
n->data = val;
n->next = NULL;
n->prev = NULL;
if(first == NULL){
n->next = n;
n->prev = n;
head->next = n;
return;
}
n->next = first;
n->prev = first->prev;
(first->prev)->next = n;
first->prev = n;
head->next = n;
return;
}
void display(NODE head){
if(head->data == 0){
printf("\nList is empty\n");
return;
}
NODE first = head->next;
NODE temp = first;
while(temp->next!=first){
```

```
printf("%d",temp->data);
temp = temp->next;
}
printf("%d\n",temp->data);
}void insert(NODE head, int val){
int x;
while(val>0){
x=val%10;
val/=10;
insertFront(head,x);
}
}
NODE addLists(NODE head1, NODE head2){
if(head1->data == 0){
return head2;
}
if(head2->data == 0){
return head1;
}
int c1 = head1->data;
int c2 = head2->data;
int diff = c1-c2;
int s = 0,i;
if(diff < 0){
diff *= -1;
for(i=0; i<diff; ++i){
insertFront(head1,0);
}
}
else if(diff > 0){
for(i = 0; i<diff; ++i){
insertFront(head2,0);
}
}
NODE sum = (NODE)malloc(sizeof(struct node));
sum->data = 0;
int carry = 0;
NODE f1 = head1->next;
NODE f2 = head2->next;
NODE op1 = f1->prev;
NODE op2 = f2->prev;
while(op1!=f1 && op2!=f2){
```

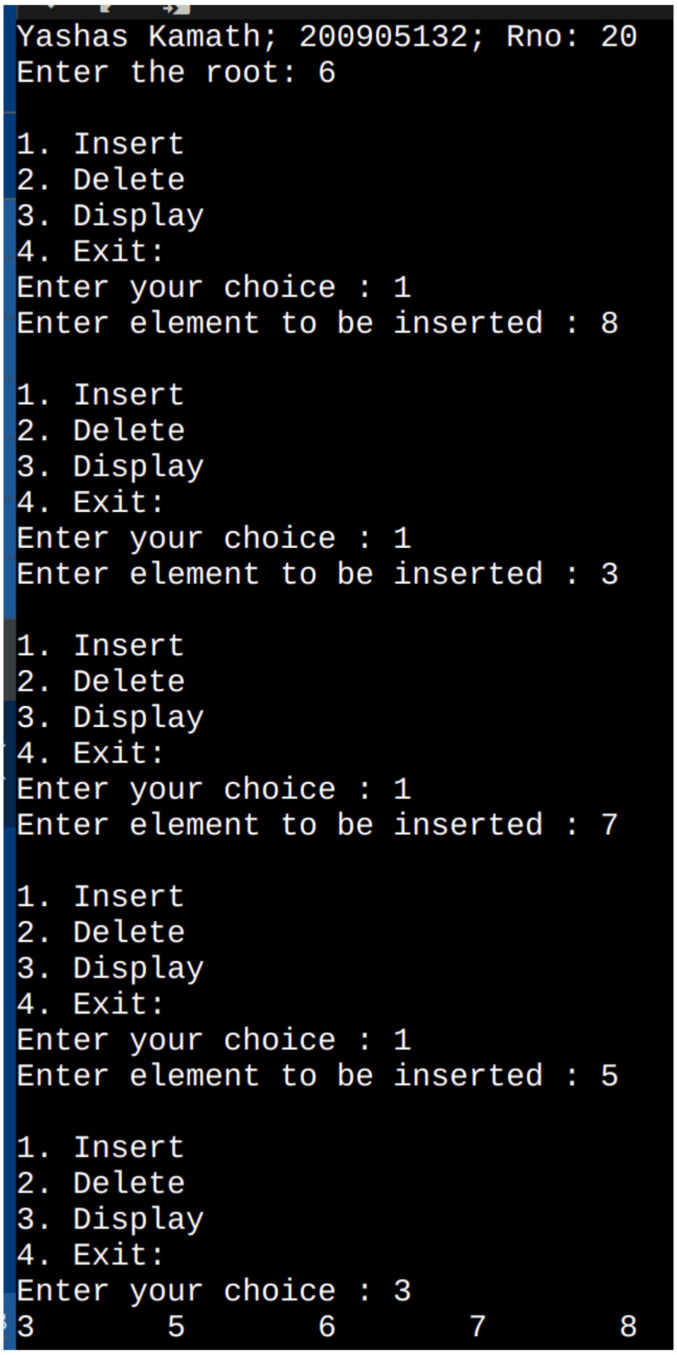
```
s = (op1->data)+(op2->data)+carry;
carry = s/10;
s%=10;
insertFront(sum,s);
op1 = op1->prev;
op2 = op2->prev;
}
s = (op1->data)+(op2->data)+carry;
carry = s/10;
s%=10;
insertFront(sum,s);
if(carry != 0){insertFront(sum,carry);
}
return sum;
}
int main(){
   printf("Yashas Kamath ; 200905132 ; Rno: 20");
NODE head1 = (NODE)malloc(sizeof(struct node));
NODE head2 = (NODE)malloc(sizeof(struct node));
NODE sum = NULL;
int v1,v2;
head1->data = 0;
head2->data = 0;
printf("\nEnter first number: ");
scanf("%d",&v1);
insert(head1,v1);
printf("\nEnter second number: ");
scanf("%d",&v2);
insert(head2,v2);
sum = addLists(head1,head2);
printf("\nSum is : ");
display(sum);
return 0;
}
```



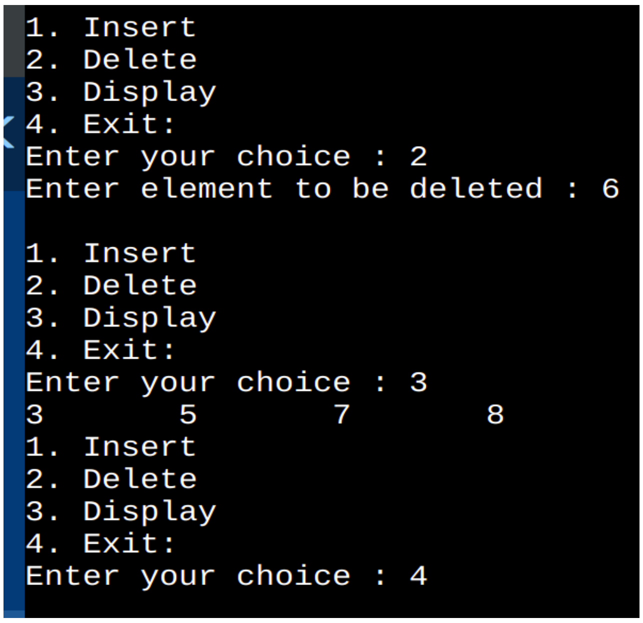
```
Yashas Kamath ; 200905132 ; Rno: 20
Enter first number: 567

Enter second number: 12876

Sum is : 13443
```

Q2) Write a menu driven program to do the following using iterative functions:
i) To create a BST for a given set of integer numbers
ii) To delete a given element from BST.
iii) Display the elements using iterative in-order traversal.
Solution:

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 10
typedef struct node{
int key;
struct node *left, *right;
}* NODE;
typedef struct{
NODE S[MAX];
int tos;
}STACK;
NODE newNODE (int item){
NODE temp = (NODE)malloc(sizeof(struct node));
temp->key = item;
temp->left = temp->right = NULL;
return temp;
}
void push (STACK *s, NODE n){
s->S[++(s->tos)] = n;
}
NODE pop (STACK *s){
return s->S[(s->tos)--];
}
void inorder (NODE root){
NODE curr;
curr = root;
STACK S;
S.tos = -1;
push(&S, root);
curr = curr->left;
while (S.tos != -1 || curr != NULL){
while (curr != NULL){
push(&S, curr);
curr = curr->left;
}
curr = pop(&S);
printf("%d\t", curr->key);
curr = curr->right;
}
}NODE insert (NODE node, int key){
if (node == NULL)
return newNODE(key);
if (key < node->key)
node->left = insert(node->left, key);
else if (key > node->key)
node->right = insert(node->right, key);
return node;
}
NODE minValueNode (NODE node){
NODE current = node;
while (current && current->left != NULL)
current = current->left;
return current;
}
NODE deleteNode (NODE root, int key){
if (root == NULL)
return root;
if (key < root->key)
root->left = deleteNode(root->left, key);
else if (key > root->key)
root->right = deleteNode(root->right, key);
else{
if (root->left == NULL){
NODE temp = root->right;
free(root);
return temp;
}
else if (root->right == NULL){
NODE temp = root->left;
free(root);
return temp;
}
NODE temp = minValueNode(root->right);
root->key = temp->key;
root->right = deleteNode(root->right, temp->key);
}
return root;
}
int main(){
    printf("Yashas Kamath; 200905132; Rno: 20\n");
NODE root = NULL;
int k;
printf("Enter the root:\t");
scanf("%d", &k);
root = insert(root, k);
int ch;
while(1){
printf("\n1. Insert\n2. Delete\n3. Display\n4. Exit:\n");
printf("Enter your choice : ");
scanf("%d", &ch);switch (ch){
case 1: printf("Enter element to be inserted : ");
scanf("%d", &k);
root = insert(root, k);
break;
case 2: printf("Enter element to be deleted : ");
scanf("%d", &k);
root = deleteNode(root, k);
break;
case 3: inorder(root);
break;
case 4: return 0;
}
}
}
```



```
Yashas Kamath; 200905132; Rno: 20
Enter the root: 6

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 8

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 3

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 7

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 1
Enter element to be inserted : 5

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 3
3       5       6       7       8
```



```
1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 2
Enter element to be deleted : 6

1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 3
3       5       7       8
1. Insert
2. Delete
3. Display
4. Exit:
Enter your choice : 4
```