Ramakrishna Mission Vidyamandira

(An Autonomous College Under University of Calcutta)

Computer Science (Honors) Semester II 2024

Paper: 2CMSCOC 1 Practical

Submitted by
Class Roll Number: 340
Registration Number:
B.Sc.
2th Semester
Batch: 2023-27

SI NO.	ASSIGNMNET STATEMENT	D-O-A	D-O-S	SIGNATURE
1.	Write a program to insert an element at desired position in an array.			
2.	Write a program to delete an element at desired position in an array.			
3.	Write a program a Sparse matrix as a user input and create Triplet.			
4.	Write a program to create a Stack and implement is operation push, pop, empty, full, peek and display.			
5.	Write a program infix to postfix.			
6.	Evaluating postfix Expression in Stack.			
7.	Write a program Tower of Hanoi.			
8.	Write a program a Linear Queue.			
9.	Write a program to implementation a Circular Queue.			
10.	Write a program to implement of insertion Sort.			
11.	Write a program to implement of Selection Sort.			
12.	Write a program to implement of Bubble Sort.			
13.	Write a program to implement of Linear Search.			
14.	Write a program to implement of Binary Search.			
15.	Create a Binary Tree and perfrom a Inorder Traversal.			
16.	Create a Binary Tree and perfrom a Preorder Traversal.			

17.	Create a Binary Tree and perfrom a Postorder Traversal.			

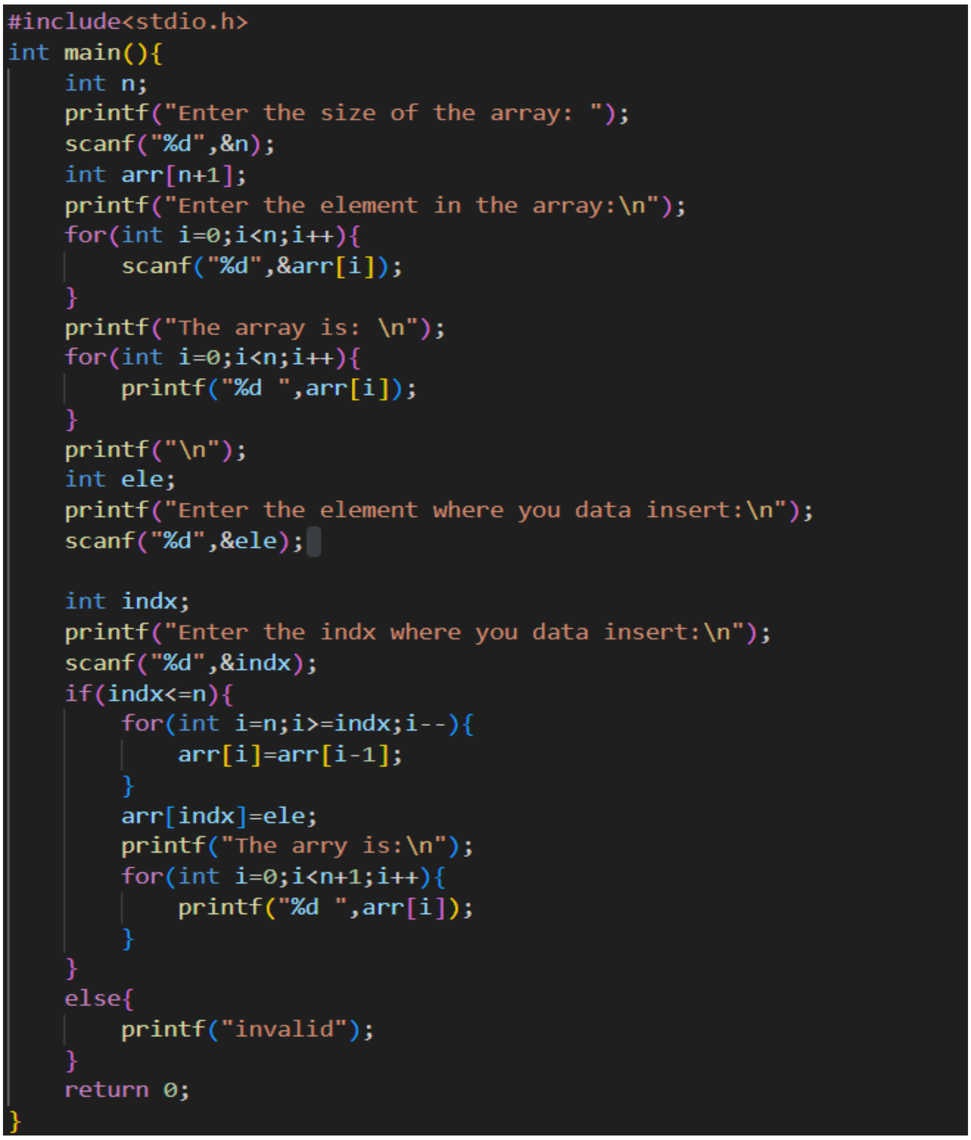
NOTE:

INDEX

Question:1

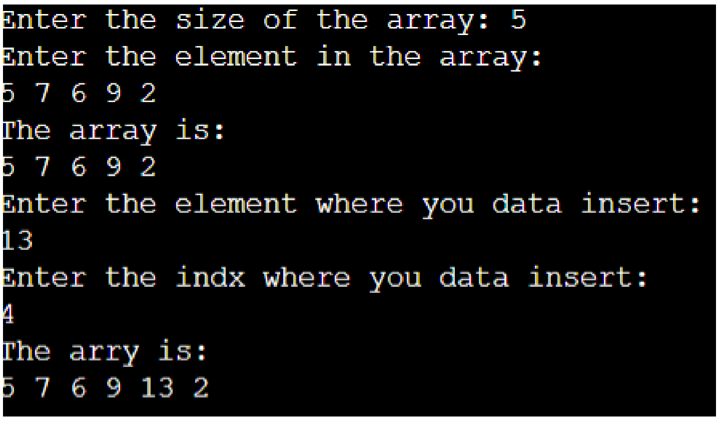
Statement: Write a program to insert an element at desired position in an array.

Source Code:



```c
#include<stdio.h>
int main(){
    int n;
    printf("Enter the size of the array: ");
    scanf("%d",&n);
    int arr[n+1];
    printf("Enter the element in the array:\n");
    for(int i=0;i<n;i++){
        scanf("%d",&arr[i]);
    }
    printf("The array is: \n");
    for(int i=0;i<n;i++){
        printf("%d ",arr[i]);
    }
    printf("\n");
    int ele;
    printf("Enter the element where you data insert:\n");
    scanf("%d",&ele);

    int indx;
    printf("Enter the indx where you data insert:\n");
    scanf("%d",&indx);
    if(indx<=n){
        for(int i=n;i>=indx;i--){
            arr[i]=arr[i-1];
        }
        arr[indx]=ele;
        printf("The arry is:\n");
        for(int i=0;i<n+1;i++){
            printf("%d ",arr[i]);
        }
    }
    else{
        printf("invalid");
    }
    return 0;
}
```

Output:

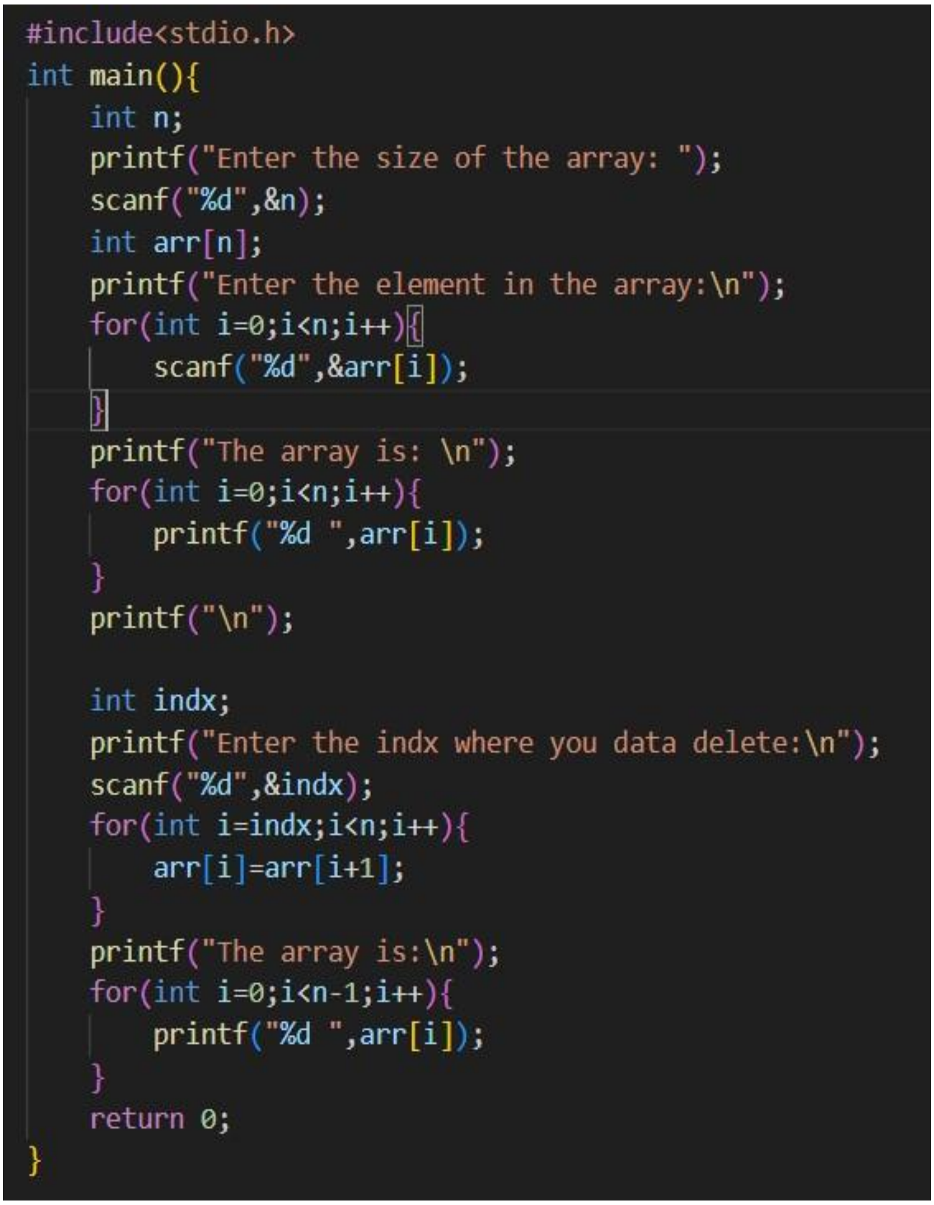


```
Enter the size of the array: 5
Enter the element in the array:
5 7 6 9 2
The array is:
5 7 6 9 2
Enter the element where you data insert:
13
Enter the indx where you data insert:
4
The arry is:
5 7 6 9 13 2
```

Question:2

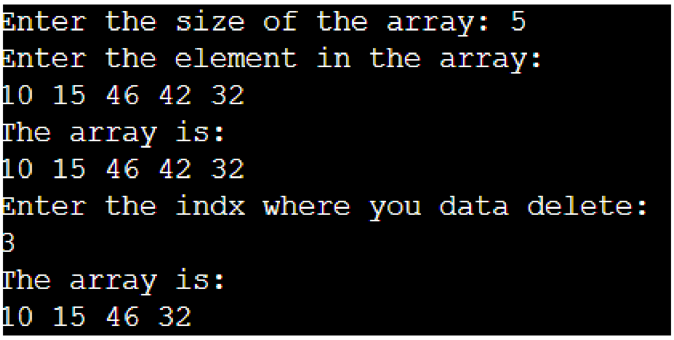
Statement: Write a program to delete an element at desired position in an array.

Source Code:



```c
#include<stdio.h>
int main(){
    int n;
    printf("Enter the size of the array: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter the element in the array:\n");
    for(int i=0;i<n;i++){
        scanf("%d",&arr[i]);
    }
    printf("The array is: \n");
    for(int i=0;i<n;i++){
        printf("%d ",arr[i]);
    }
    printf("\n");

    int indx;
    printf("Enter the indx where you data delete:\n");
    scanf("%d",&indx);
    for(int i=indx;i<n;i++){
        arr[i]=arr[i+1];
    }
    printf("The array is:\n");
    for(int i=0;i<n-1;i++){
        printf("%d ",arr[i]);
    }
    return 0;
}
```

Output:

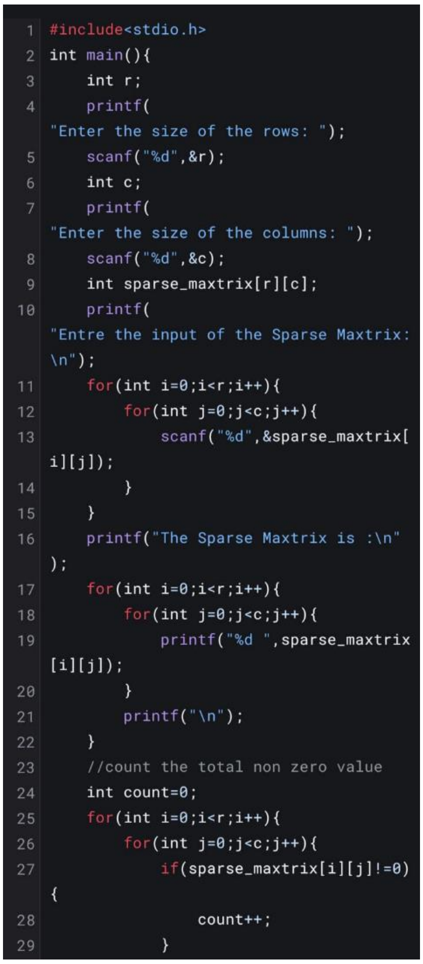


```
Enter the size of the array: 5
Enter the element in the array:
10 15 46 42 32
The array is:
10 15 46 42 32
Enter the indx where you data delete:
3
The array is:
10 15 46 32
```

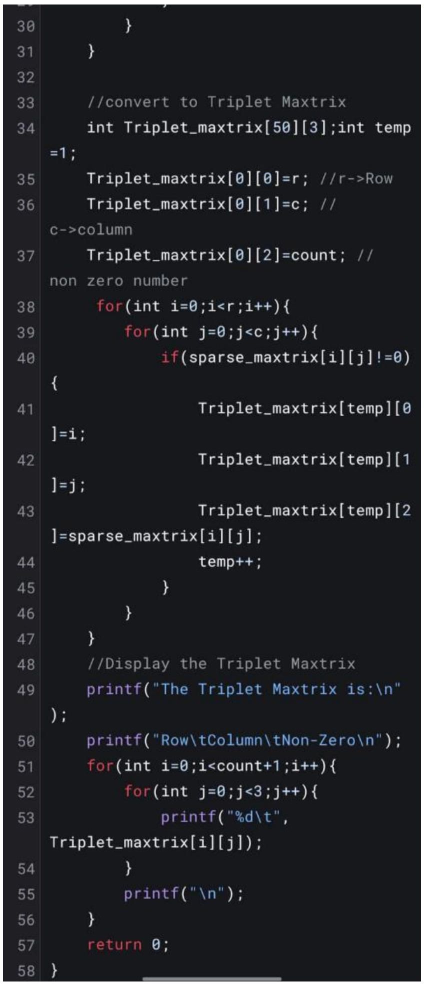
Question:3

Statement: Write a program a Sparse matrix as a user input and create Triplet.

Source Code:

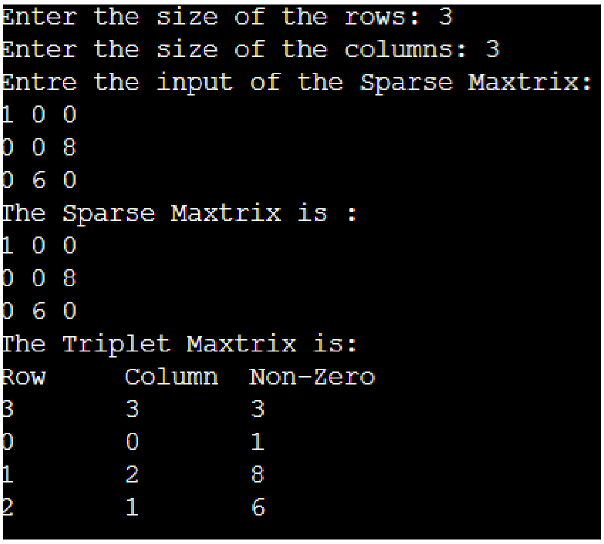


```c
#include<stdio.h>
int main(){
    int r;
    printf("Enter the size of the rows: ");
    scanf("%d",&r);
    int c;
    printf("Enter the size of the columns: ");
    scanf("%d",&c);
    int sparse_maxtrix[r][c];
    printf("Entre the input of the Sparse Maxtrix:\n");
    for(int i=0;i<r;i++){
        for(int j=0;j<c;j++){
            scanf("%d",&sparse_maxtrix[i][j]);
        }
    }
    printf("The Sparse Maxtrix is :\n");
    for(int i=0;i<r;i++){
        for(int j=0;j<c;j++){
            printf("%d ",sparse_maxtrix[i][j]);
        }
        printf("\n");
    }
    //count the total non zero value
    int count=0;
    for(int i=0;i<r;i++){
        for(int j=0;j<c;j++){
            if(sparse_maxtrix[i][j]!=0){
                count++;
            }
```



```c
        }
    }

    //convert to Triplet Maxtrix
    int Triplet_maxtrix[50][3];int temp=1;
    Triplet_maxtrix[0][0]=r; //r->Row
    Triplet_maxtrix[0][1]=c; //c->column
    Triplet_maxtrix[0][2]=count; //non zero number
     for(int i=0;i<r;i++){
        for(int j=0;j<c;j++){
            if(sparse_maxtrix[i][j]!=0){
                Triplet_maxtrix[temp][0]=i;
                Triplet_maxtrix[temp][1]=j;
                Triplet_maxtrix[temp][2]=sparse_maxtrix[i][j];
                temp++;
            }
        }
    }
    //Display the Triplet Maxtrix
    printf("The Triplet Maxtrix is:\n");
    printf("Row\tColumn\tNon-Zero\n");
    for(int i=0;i<count+1;i++){
        for(int j=0;j<3;j++){
            printf("%d\t",Triplet_maxtrix[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

Output

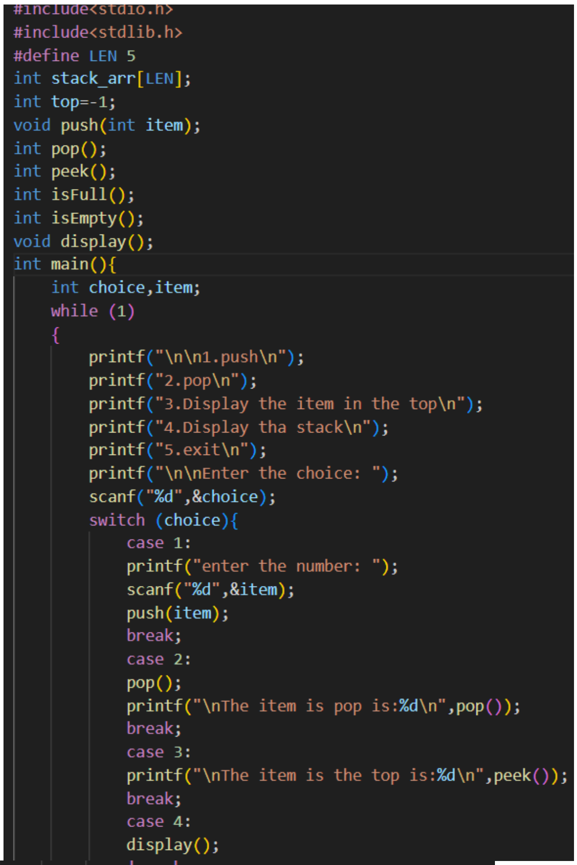


```
Enter the size of the rows: 3
Enter the size of the columns: 3
Entre the input of the Sparse Maxtrix:
1 0 0
0 0 8
0 6 0
The Sparse Maxtrix is :
1 0 0
0 0 8
0 6 0
The Triplet Maxtrix is:
Row     Column  Non-Zero
3       3       3
0       0       1
1       2       8
2       1       6
```

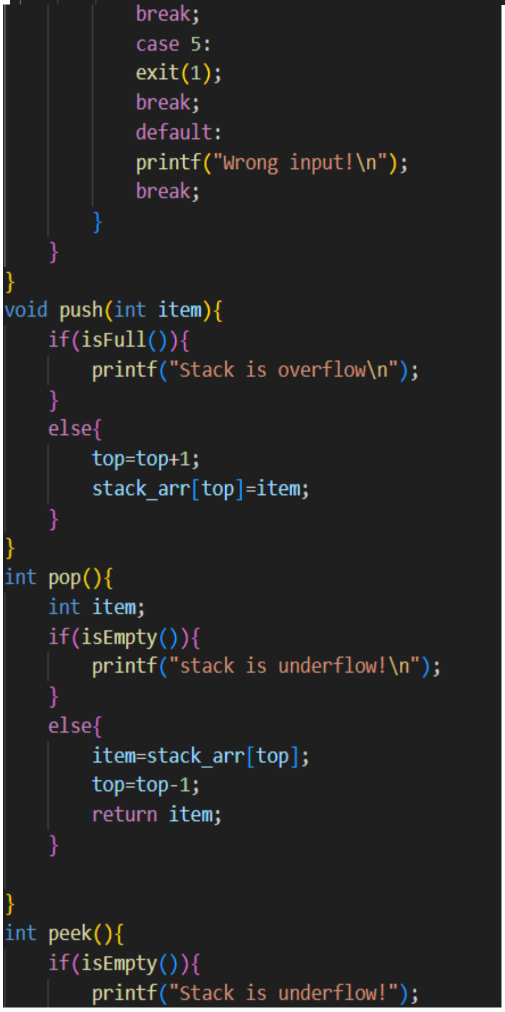
Question:4

Statement: Write a program to create a Stack and implement is operation push, pop, empty, full, peek and display.

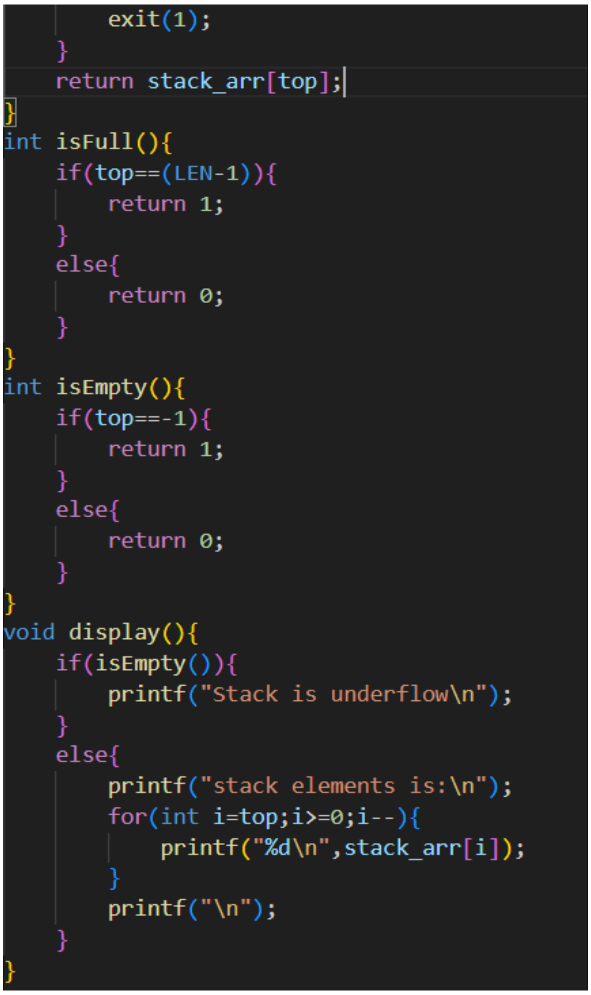
Source Code:



```
#include<stdio.h>
#include<stdlib.h>
#define LEN 5
int stack_arr[LEN];
int top=-1;
void push(int item);
int pop();
int peek();
int isFull();
int isEmpty();
void display();
int main(){
    int choice,item;
    while (1)
    {
        printf("\n\n1.push\n");
        printf("2.pop\n");
        printf("3.Display the item in the top\n");
        printf("4.Display tha stack\n");
        printf("5.exit\n");
        printf("\n\nEnter the choice: ");
        scanf("%d",&choice);
        switch (choice){
            case 1:
            printf("enter the number: ");
            scanf("%d",&item);
            push(item);
            break;
            case 2:
            pop();
            printf("\nThe item is pop is:%d\n",pop());
            break;
            case 3:
            printf("\nThe item is the top is:%d\n",peek());
            break;
            case 4:
            display();
```

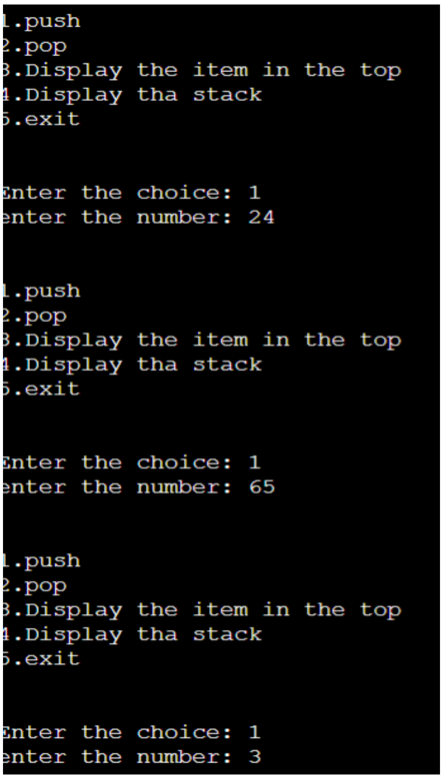


```
            break;
            case 5:
            exit(1);
            break;
            default:
            printf("Wrong input!\n");
            break;
        }
    }
}
void push(int item){
    if(isFull()){
        printf("Stack is overflow\n");
    }
    else{
        top=top+1;
        stack_arr[top]=item;
    }
}
int pop(){
    int item;
    if(isEmpty()){
        printf("stack is underflow!\n");
    }
    else{
        item=stack_arr[top];
        top=top-1;
        return item;
    }

}
int peek(){
    if(isEmpty()){
        printf("Stack is underflow!");
```

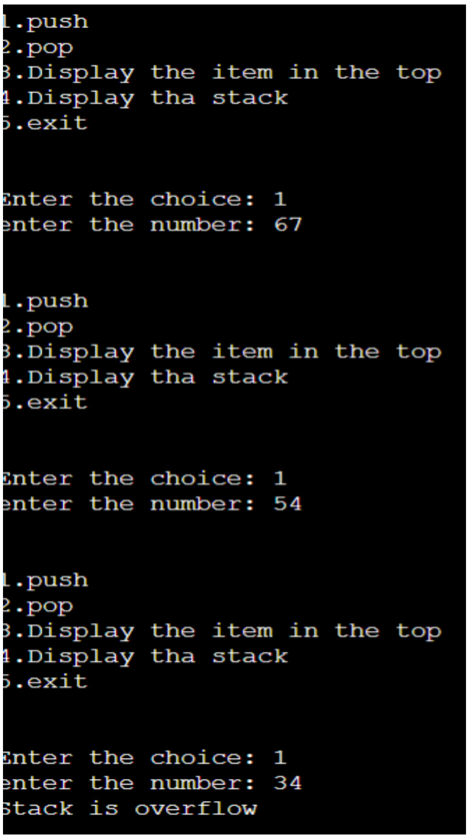


```
        exit(1);
    }
    return stack_arr[top];
}
int isFull(){
    if(top==(LEN-1)){
        return 1;
    }
    else{
        return 0;
    }
}
int isEmpty(){
    if(top==-1){
        return 1;
    }
    else{
        return 0;
    }
}
void display(){
    if(isEmpty()){
        printf("Stack is underflow\n");
    }
    else{
        printf("stack elements is:\n");
        for(int i=top;i>=0;i--){
            printf("%d\n",stack_arr[i]);
        }
        printf("\n");
    }
}
```

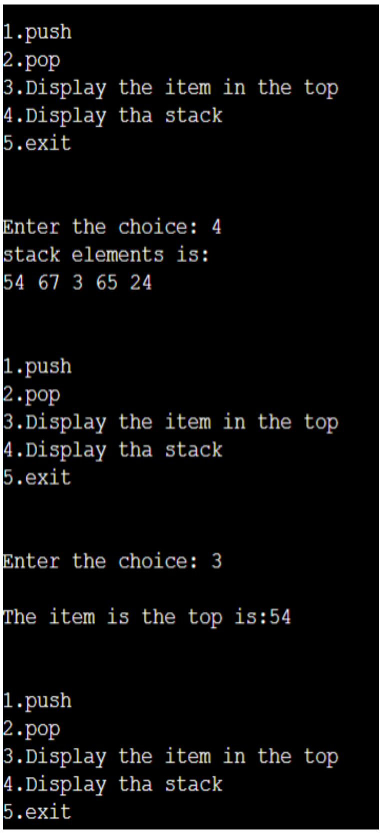
Output:



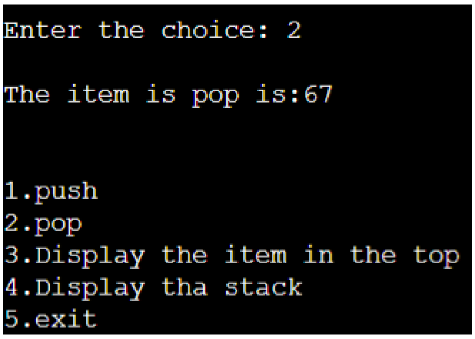
```
1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 1
enter the number: 24


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 1
enter the number: 65


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 1
enter the number: 3
```



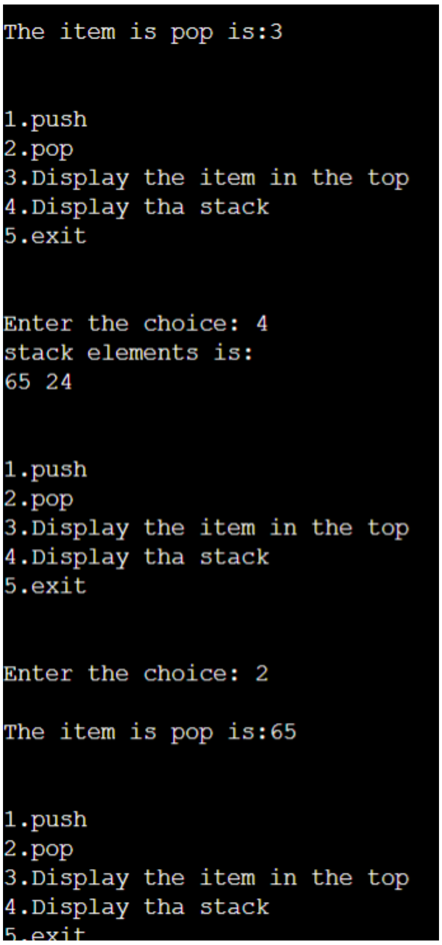
```
1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 1
enter the number: 67


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 1
enter the number: 54


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 1
enter the number: 34
Stack is overflow
```



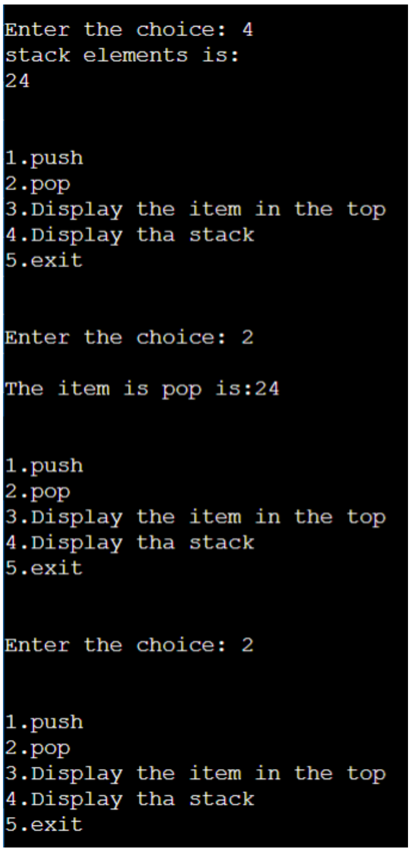
```
1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 4
stack elements is:
54 67 3 65 24


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 3

The item is the top is:54


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit
```



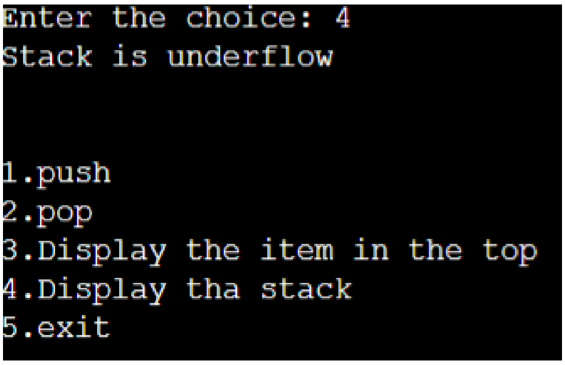
```
Enter the choice: 2

The item is pop is:67


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit
```



```
The item is pop is:3


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 4
stack elements is:
65 24


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 2

The item is pop is:65


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit
```



```
Enter the choice: 4
stack elements is:
24


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 2

The item is pop is:24


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit


Enter the choice: 2


1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit
```

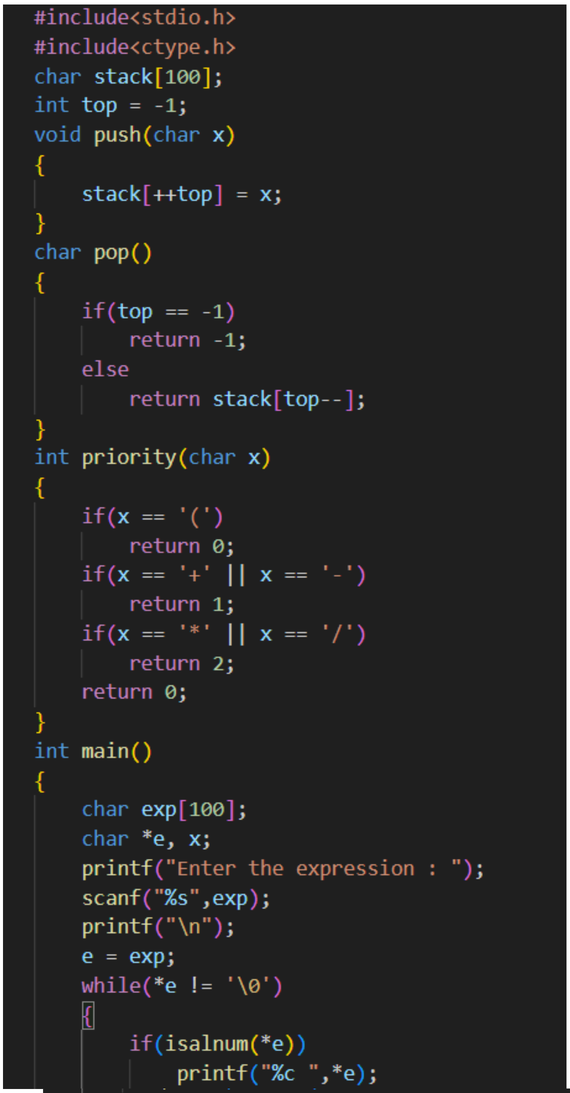


```
Enter the choice: 4
Stack is underflow



1.push
2.pop
3.Display the item in the top
4.Display tha stack
5.exit
```

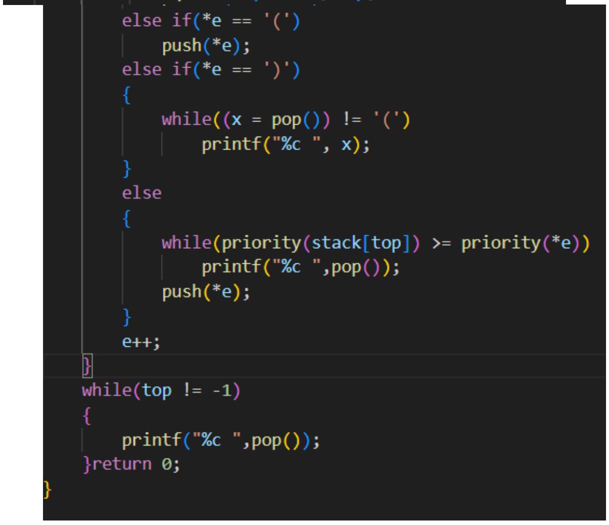
Question:5

Statement: Write a program infix to postfix.

Source Code:



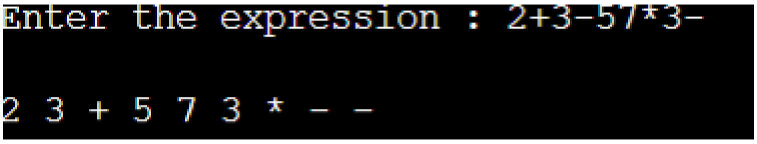
```c
#include<stdio.h>
#include<ctype.h>
char stack[100];
int top = -1;
void push(char x)
{
    stack[++top] = x;
}
char pop()
{
    if(top == -1)
        return -1;
    else
        return stack[top--];
}
int priority(char x)
{
    if(x == '(')
        return 0;
    if(x == '+' || x == '-')
        return 1;
    if(x == '*' || x == '/')
        return 2;
    return 0;
}
int main()
{
    char exp[100];
    char *e, x;
    printf("Enter the expression : ");
    scanf("%s",exp);
    printf("\n");
    e = exp;
    while(*e != '\0')
    {
        if(isalnum(*e))
            printf("%c ",*e);
```



```c
        else if(*e == '(')
            push(*e);
        else if(*e == ')')
        {
            while((x = pop()) != '(')
                printf("%c ", x);
        }
        else
        {
            while(priority(stack[top]) >= priority(*e))
                printf("%c ",pop());
            push(*e);
        }
        e++;
    }
    while(top != -1)
    {
        printf("%c ",pop());
    }return 0;
}
```

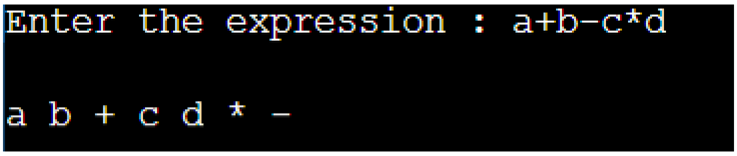
Output:

Case-1:



```
Enter the expression : 2+3-57*3-

2 3 + 5 7 3 * - -
```

Case-2:

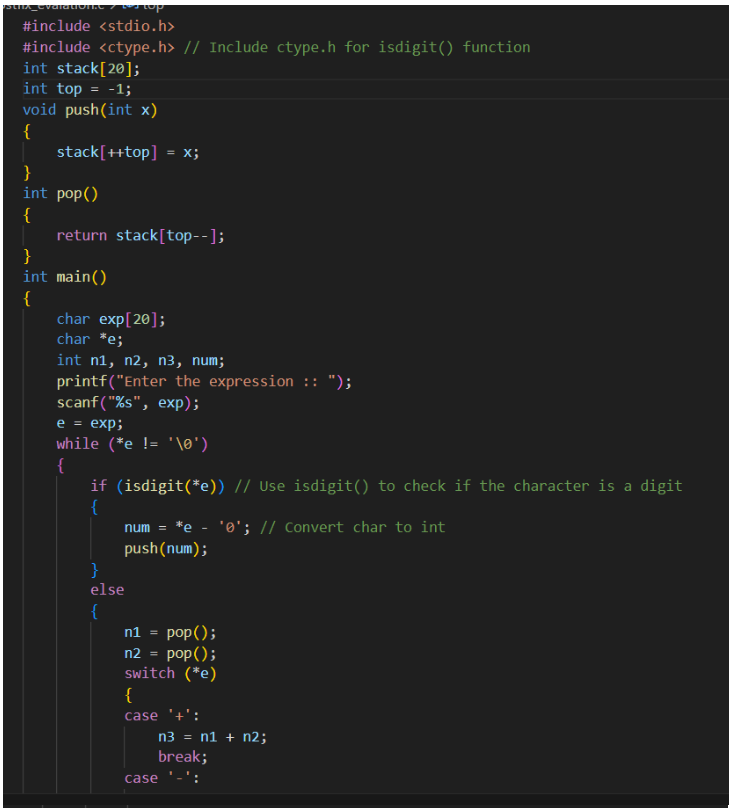


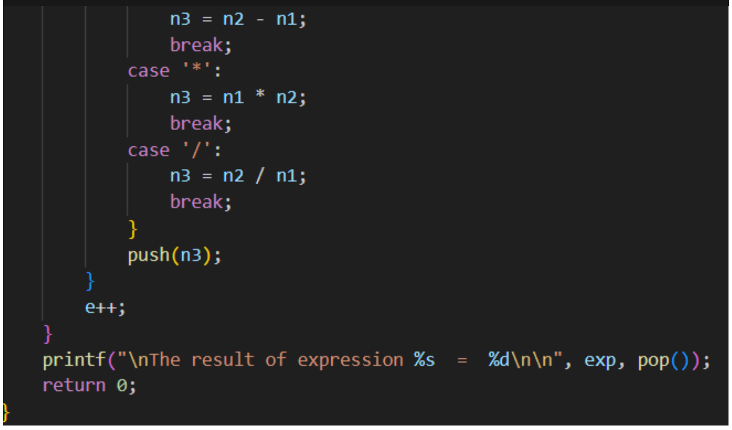
```
Enter the expression : a+b-c*d

a b + c d * -
```

Question:6

Statement: Evaluating postfix Expression in Stack.

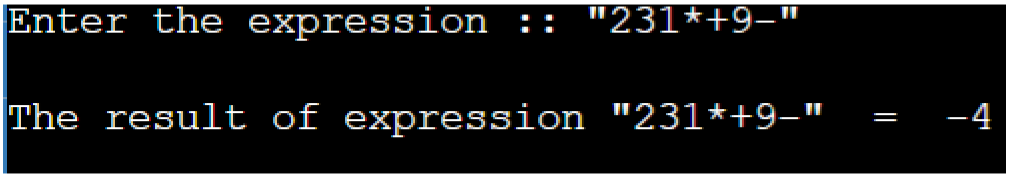
Source Code:





```c
#include <stdio.h>
#include <ctype.h> // Include ctype.h for isdigit() function
int stack[20];
int top = -1;
void push(int x)
{
    stack[++top] = x;
}
int pop()
{
    return stack[top--];
}
int main()
{
    char exp[20];
    char *e;
    int n1, n2, n3, num;
    printf("Enter the expression :: ");
    scanf("%s", exp);
    e = exp;
    while (*e != '\0')
    {
        if (isdigit(*e)) // Use isdigit() to check if the character is a digit
        {
            num = *e - '0'; // Convert char to int
            push(num);
        }
        else
        {
            n1 = pop();
            n2 = pop();
            switch (*e)
            {
            case '+':
                n3 = n1 + n2;
                break;
            case '-':
                n3 = n2 - n1;
                break;
            case '*':
                n3 = n1 * n2;
                break;
            case '/':
                n3 = n2 / n1;
                break;
            }
            push(n3);
        }
        e++;
    }
    printf("\nThe result of expression %s  =  %d\n\n", exp, pop());
    return 0;
}
```

Output:

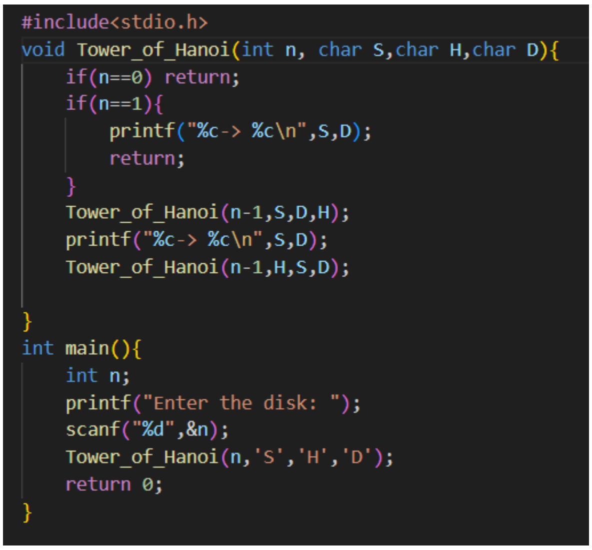


```
Enter the expression :: "231*+9-"

The result of expression "231*+9-"  =  -4
```

Question:7

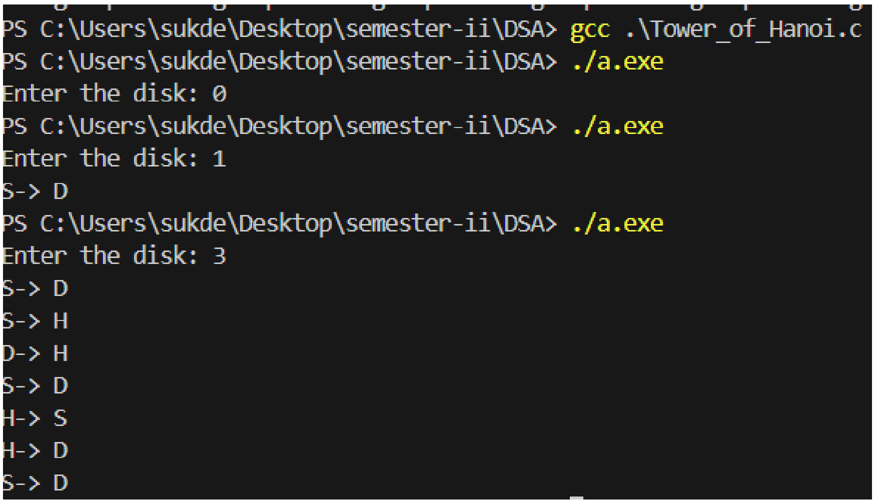
Statement: Write a program Tower of Hanoi.

Source Code:



```c
#include<stdio.h>
void Tower_of_Hanoi(int n, char S,char H,char D){
    if(n==0) return;
    if(n==1){
        printf("%c-> %c\n",S,D);
        return;
    }
    Tower_of_Hanoi(n-1,S,D,H);
    printf("%c-> %c\n",S,D);
    Tower_of_Hanoi(n-1,H,S,D);

}
int main(){
    int n;
    printf("Enter the disk: ");
    scanf("%d",&n);
    Tower_of_Hanoi(n,'S','H','D');
    return 0;
}
```

Output:

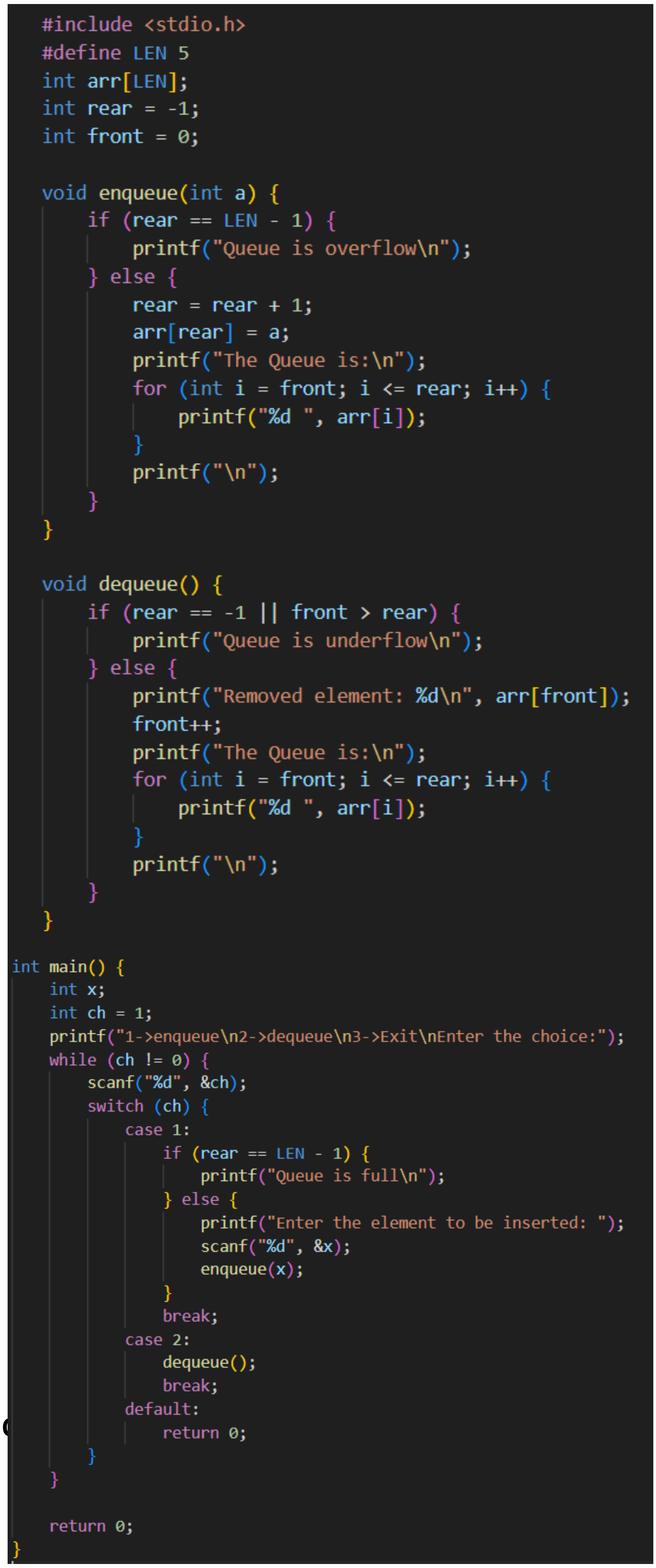


```
PS C:\Users\sukde\Desktop\semester-ii\DSA> gcc .\Tower_of_Hanoi.c
PS C:\Users\sukde\Desktop\semester-ii\DSA> ./a.exe
Enter the disk: 0
PS C:\Users\sukde\Desktop\semester-ii\DSA> ./a.exe
Enter the disk: 1
S-> D
PS C:\Users\sukde\Desktop\semester-ii\DSA> ./a.exe
Enter the disk: 3
S-> D
S-> H
D-> H
S-> D
H-> S
H-> D
S-> D
```

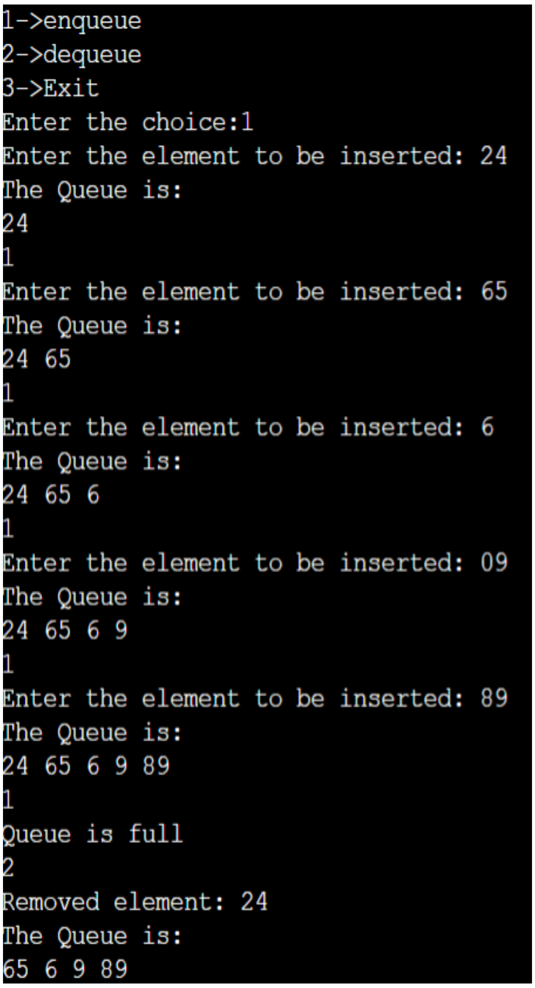
Question:8

Statement: Write a program a Linear Queue.

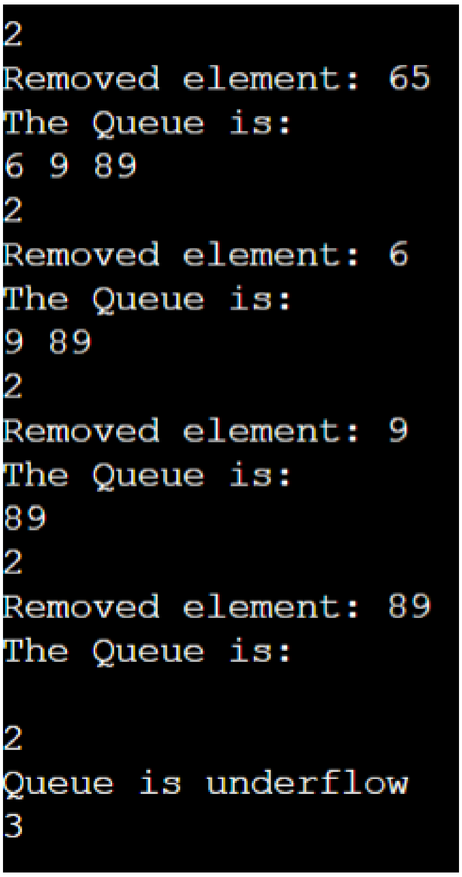
Source Code:



```c
#include <stdio.h>
#define LEN 5
int arr[LEN];
int rear = -1;
int front = 0;

void enqueue(int a) {
    if (rear == LEN - 1) {
        printf("Queue is overflow\n");
    } else {
        rear = rear + 1;
        arr[rear] = a;
        printf("The Queue is:\n");
        for (int i = front; i <= rear; i++) {
            printf("%d ", arr[i]);
        }
        printf("\n");
    }
}

void dequeue() {
    if (rear == -1 || front > rear) {
        printf("Queue is underflow\n");
    } else {
        printf("Removed element: %d\n", arr[front]);
        front++;
        printf("The Queue is:\n");
        for (int i = front; i <= rear; i++) {
            printf("%d ", arr[i]);
        }
        printf("\n");
    }
}

int main() {
    int x;
    int ch = 1;
    printf("1->enqueue\n2->dequeue\n3->Exit\nEnter the choice:");
    while (ch != 0) {
        scanf("%d", &ch);
        switch (ch) {
            case 1:
                if (rear == LEN - 1) {
                    printf("Queue is full\n");
                } else {
                    printf("Enter the element to be inserted: ");
                    scanf("%d", &x);
                    enqueue(x);
                }
                break;
            case 2:
                dequeue();
                break;
            default:
                return 0;
        }
    }

    return 0;
}
```



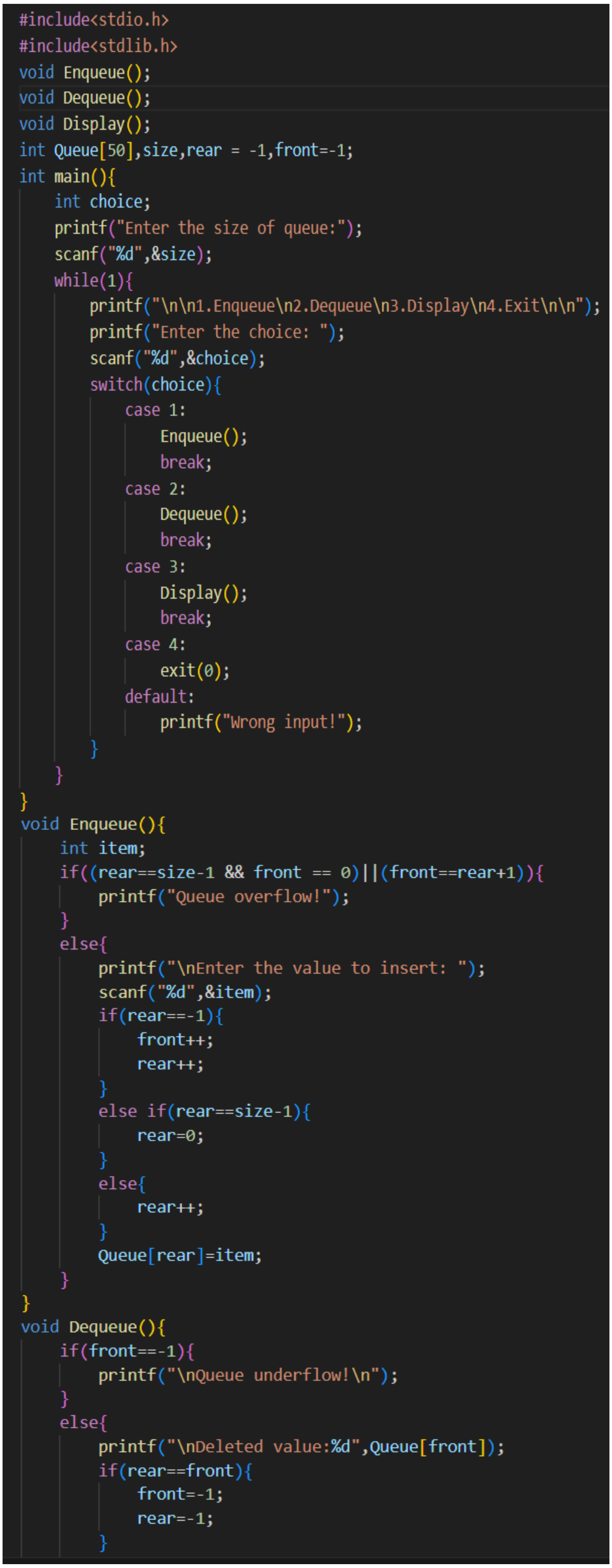
```
1->enqueue
2->dequeue
3->Exit
Enter the choice:1
Enter the element to be inserted: 24
The Queue is:
24
1
Enter the element to be inserted: 65
The Queue is:
24 65
1
Enter the element to be inserted: 6
The Queue is:
24 65 6
1
Enter the element to be inserted: 09
The Queue is:
24 65 6 9
1
Enter the element to be inserted: 89
The Queue is:
24 65 6 9 89
1
Queue is full
2
Removed element: 24
The Queue is:
65 6 9 89
```



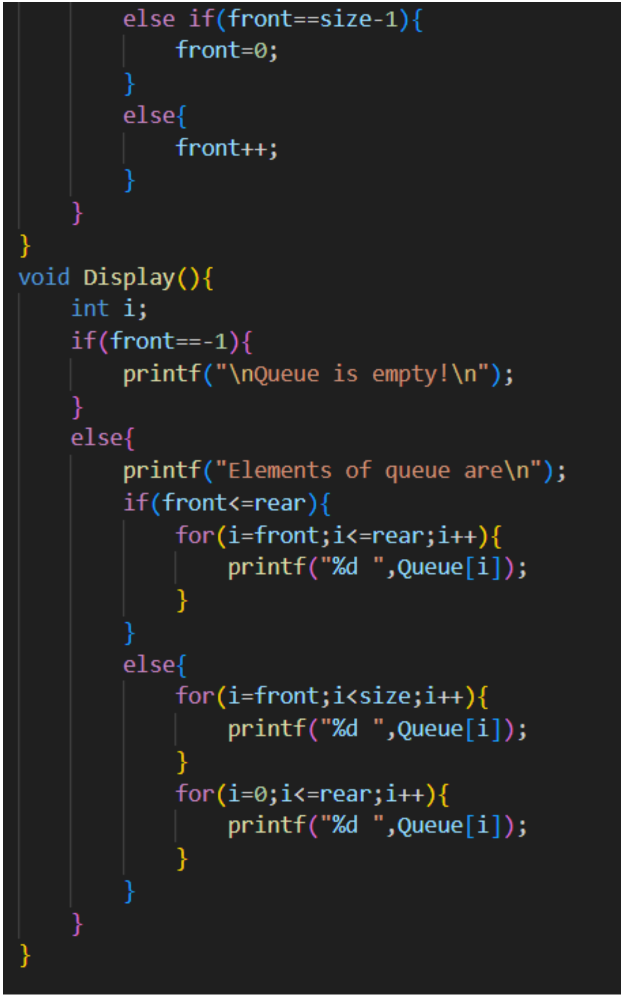
```
2
Removed element: 65
The Queue is:
6 9 89
2
Removed element: 6
The Queue is:
9 89
2
Removed element: 9
The Queue is:
89
2
Removed element: 89
The Queue is:

2
Queue is underflow
3
```

Statement: Write a program to implementation a Circular Queue.

Source Code:

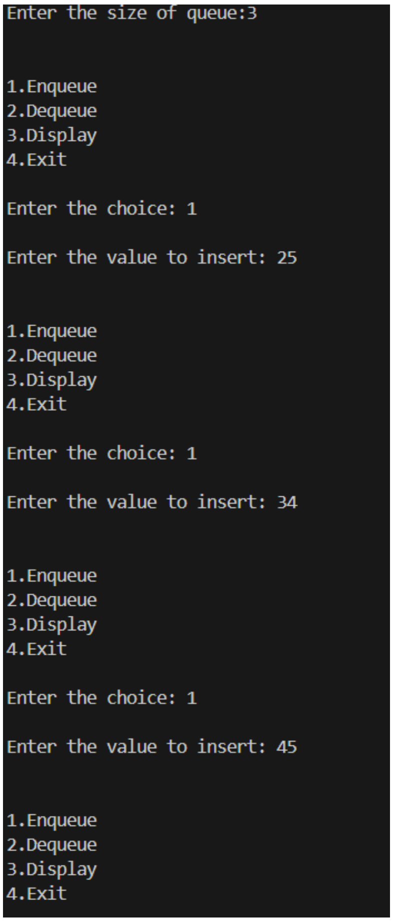


```
#include<stdio.h>
#include<stdlib.h>
void Enqueue();
void Dequeue();
void Display();
int Queue[50],size,rear = -1,front=-1;
int main(){
    int choice;
    printf("Enter the size of queue:");
    scanf("%d",&size);
    while(1){
        printf("\n\n1.Enqueue\n2.Dequeue\n3.Display\n4.Exit\n\n");
        printf("Enter the choice: ");
        scanf("%d",&choice);
        switch(choice){
            case 1:
                Enqueue();
                break;
            case 2:
                Dequeue();
                break;
            case 3:
                Display();
                break;
            case 4:
                exit(0);
            default:
                printf("Wrong input!");
        }
    }
}
void Enqueue(){
    int item;
    if((rear==size-1 && front == 0)||(front==rear+1)){
        printf("Queue overflow!");
    }
    else{
        printf("\nEnter the value to insert: ");
        scanf("%d",&item);
        if(rear==-1){
            front++;
            rear++;
        }
        else if(rear==size-1){
            rear=0;
        }
        else{
            rear++;
        }
        Queue[rear]=item;
    }
}
void Dequeue(){
    if(front==-1){
        printf("\nQueue underflow!\n");
    }
    else{
        printf("\nDeleted value:%d",Queue[front]);
        if(rear==front){
            front=-1;
            rear=-1;
        }
```

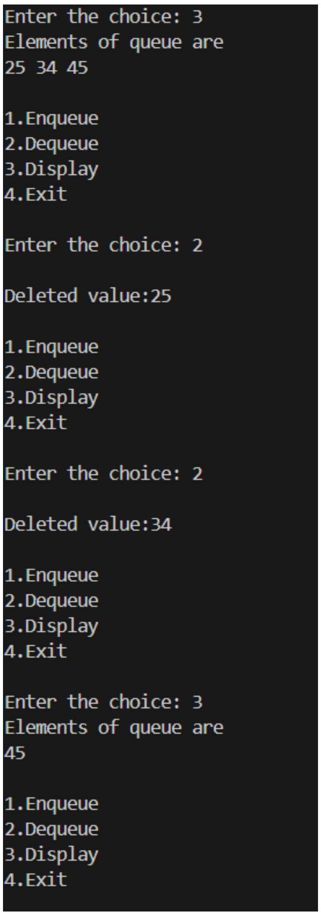


```
        else if(front==size-1){
            front=0;
        }
        else{
            front++;
        }
    }
}
void Display(){
    int i;
    if(front==-1){
        printf("\nQueue is empty!\n");
    }
    else{
        printf("Elements of queue are\n");
        if(front<=rear){
            for(i=front;i<=rear;i++){
                printf("%d ",Queue[i]);
            }
        }
        else{
            for(i=front;i<size;i++){
                printf("%d ",Queue[i]);
            }
            for(i=0;i<=rear;i++){
                printf("%d ",Queue[i]);
            }
        }
    }
}
```

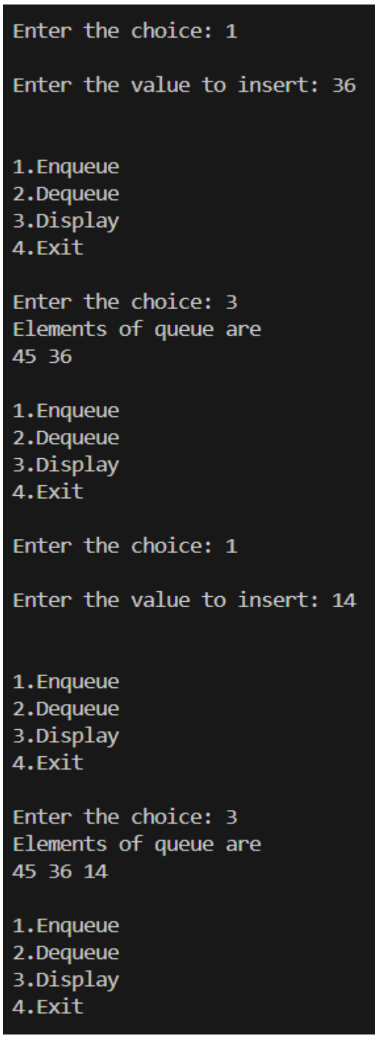
Output:



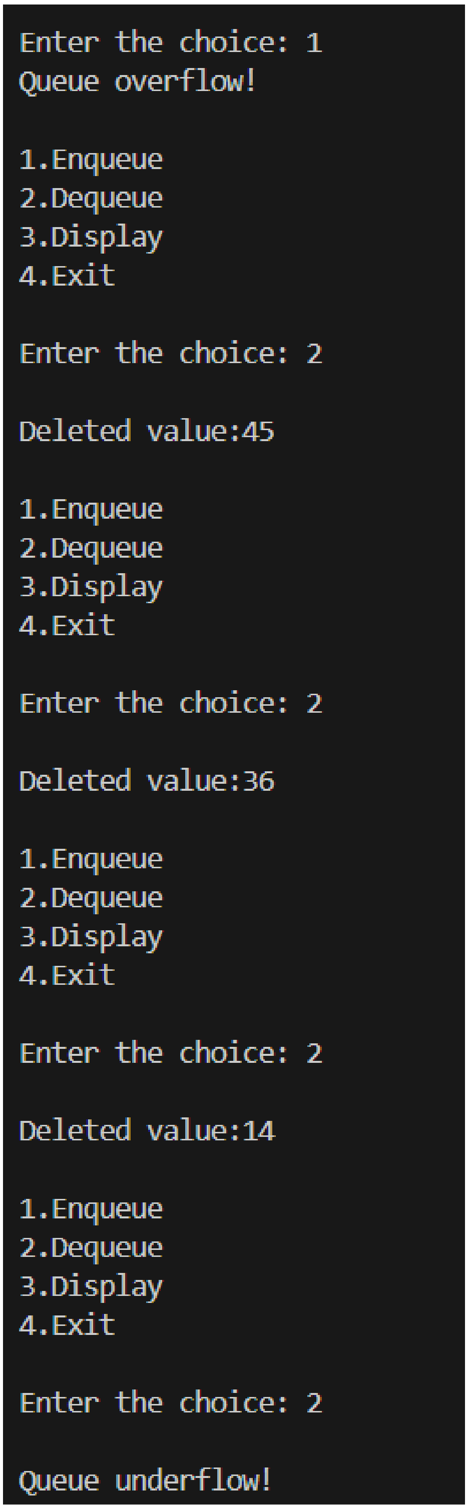
```
Enter the size of queue:3


1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 1

Enter the value to insert: 25


1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 1

Enter the value to insert: 34


1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 1

Enter the value to insert: 45


1.Enqueue
2.Dequeue
3.Display
4.Exit
```



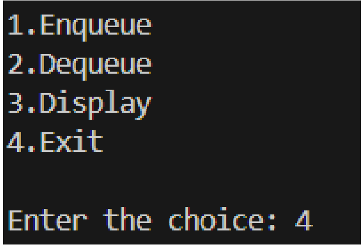
```
Enter the choice: 3
Elements of queue are
25 34 45

1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 2

Deleted value:25

1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 2

Deleted value:34

1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 3
Elements of queue are
45

1.Enqueue
2.Dequeue
3.Display
4.Exit
```



```
Enter the choice: 1

Enter the value to insert: 36


1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 3
Elements of queue are
45 36

1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 1

Enter the value to insert: 14


1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 3
Elements of queue are
45 36 14

1.Enqueue
2.Dequeue
3.Display
4.Exit
```



```
Enter the choice: 1
Queue overflow!

1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 2

Deleted value:45

1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 2

Deleted value:36

1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 2

Deleted value:14

1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 2

Queue underflow!
```

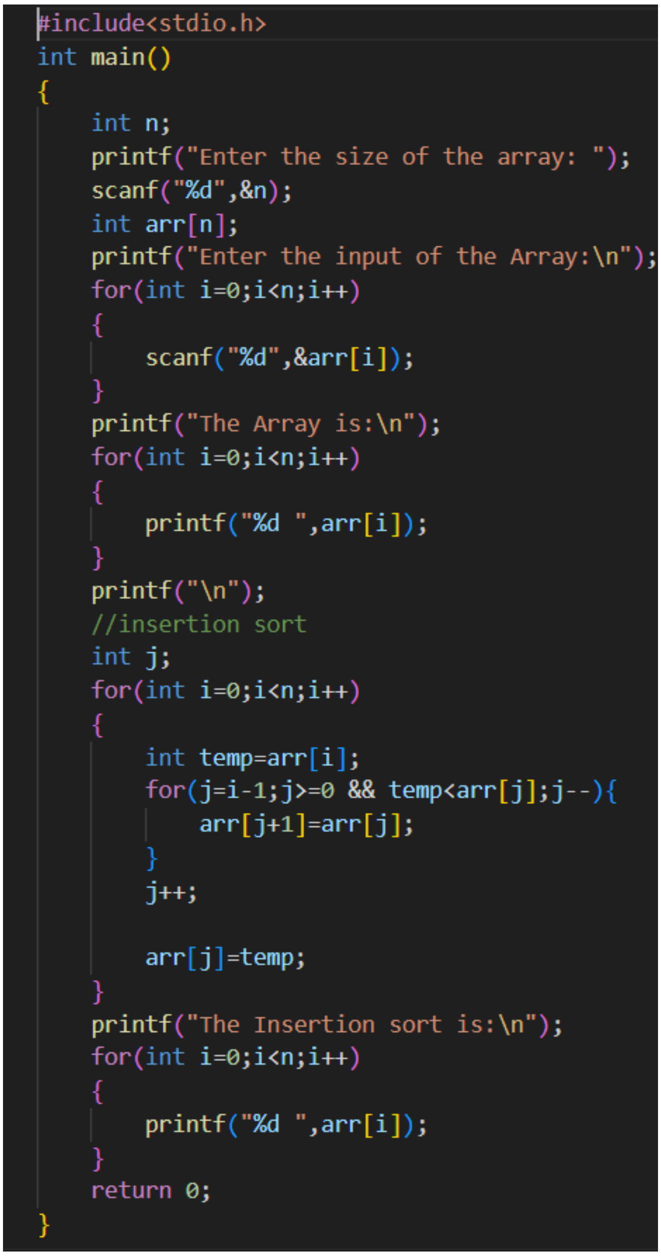


```
1.Enqueue
2.Dequeue
3.Display
4.Exit

Enter the choice: 4
```

Question:10

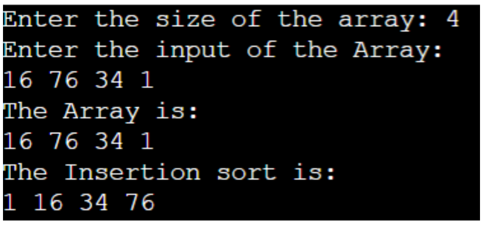
Statement: Write a program to implement of insertion Sort.

Source Code:



```
#include<stdio.h>
int main()
{
    int n;
    printf("Enter the size of the array: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter the input of the Array:\n");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("The Array is:\n");
    for(int i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
    //insertion sort
    int j;
    for(int i=0;i<n;i++)
    {
        int temp=arr[i];
        for(j=i-1;j>=0 && temp<arr[j];j--){
            arr[j+1]=arr[j];
        }
        j++;

        arr[j]=temp;
    }
    printf("The Insertion sort is:\n");
    for(int i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    return 0;
}
```

Output:

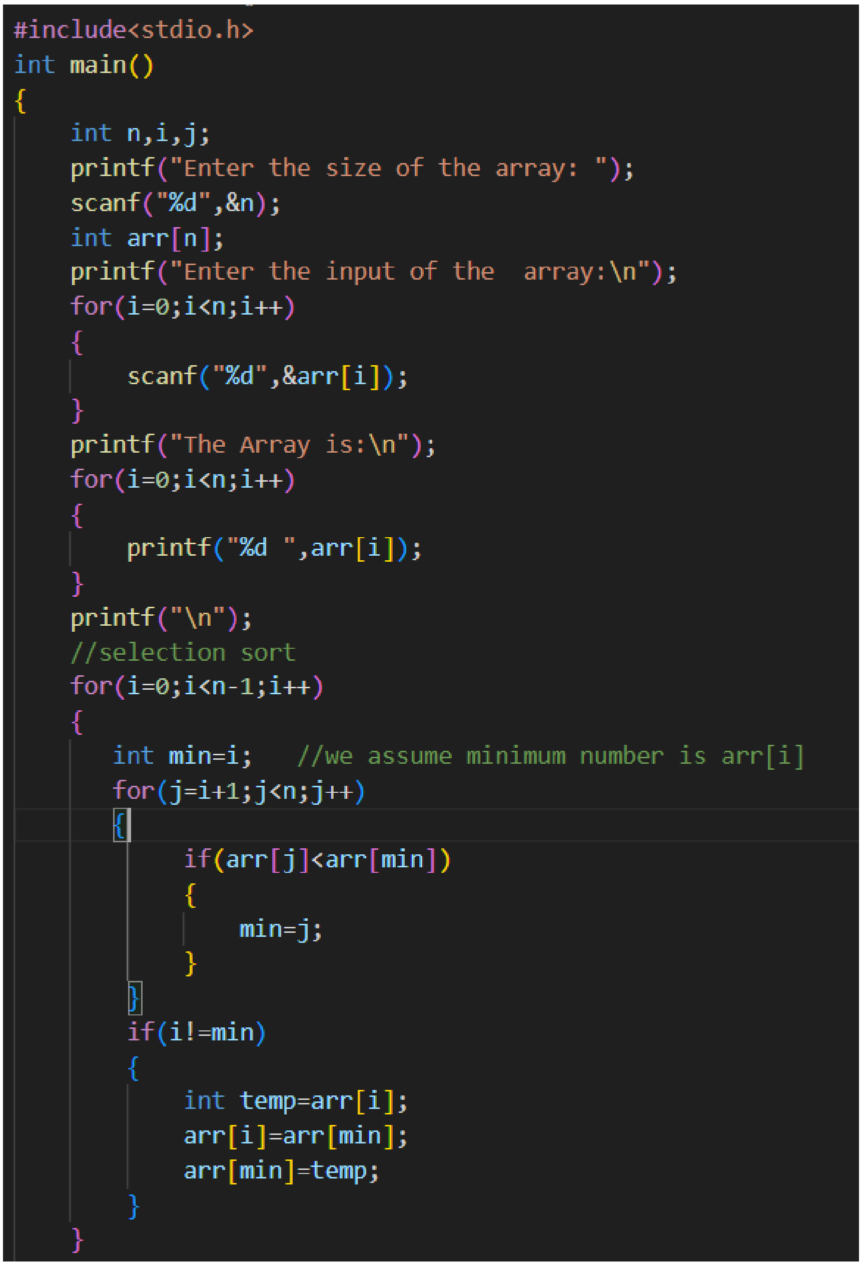


```
Enter the size of the array: 4
Enter the input of the Array:
16 76 34 1
The Array is:
16 76 34 1
The Insertion sort is:
1 16 34 76
```

Question:11

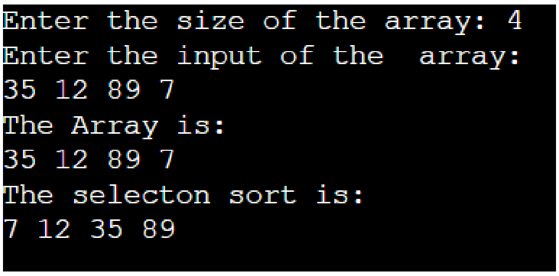
Statement: Write a program to implement of Selection Sort.

Source Code:



```
#include<stdio.h>
int main()
{
    int n,i,j;
    printf("Enter the size of the array: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter the input of the  array:\n");
    for(i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("The Array is:\n");
    for(i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
    //selection sort
    for(i=0;i<n-1;i++)
    {
       int min=i;   //we assume minimum number is arr[i]
       for(j=i+1;j<n;j++)
       {
            if(arr[j]<arr[min])
            {
                min=j;
            }
        }
        if(i!=min)
        {
            int temp=arr[i];
            arr[i]=arr[min];
            arr[min]=temp;
        }
    }
```

Output:

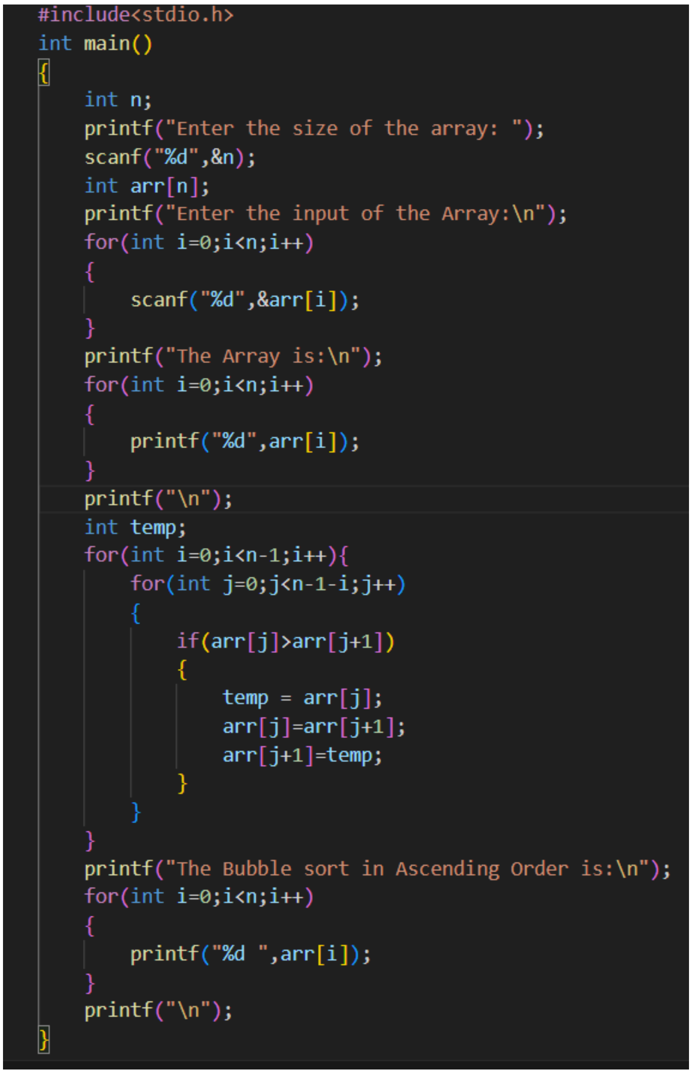


```
Enter the size of the array: 4
Enter the input of the  array:
35 12 89 7
The Array is:
35 12 89 7
The selecton sort is:
7 12 35 89
```

Question:12

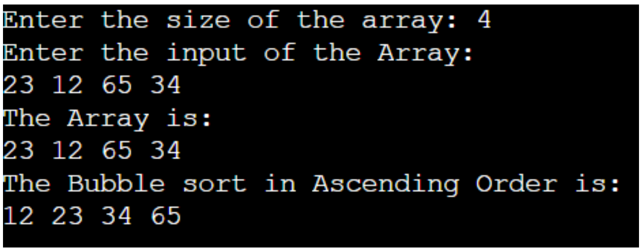
Statement: Write a program to implement of Bubble Sort.

Source Code:



```
#include<stdio.h>
int main()
{
    int n;
    printf("Enter the size of the array: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter the input of the Array:\n");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    printf("The Array is:\n");
    for(int i=0;i<n;i++)
    {
        printf("%d",arr[i]);
    }
    printf("\n");
    int temp;
    for(int i=0;i<n-1;i++){
        for(int j=0;j<n-1-i;j++)
        {
            if(arr[j]>arr[j+1])
            {
                temp = arr[j];
                arr[j]=arr[j+1];
                arr[j+1]=temp;
            }
        }
    }
    printf("The Bubble sort in Ascending Order is:\n");
    for(int i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
}
```

Output:

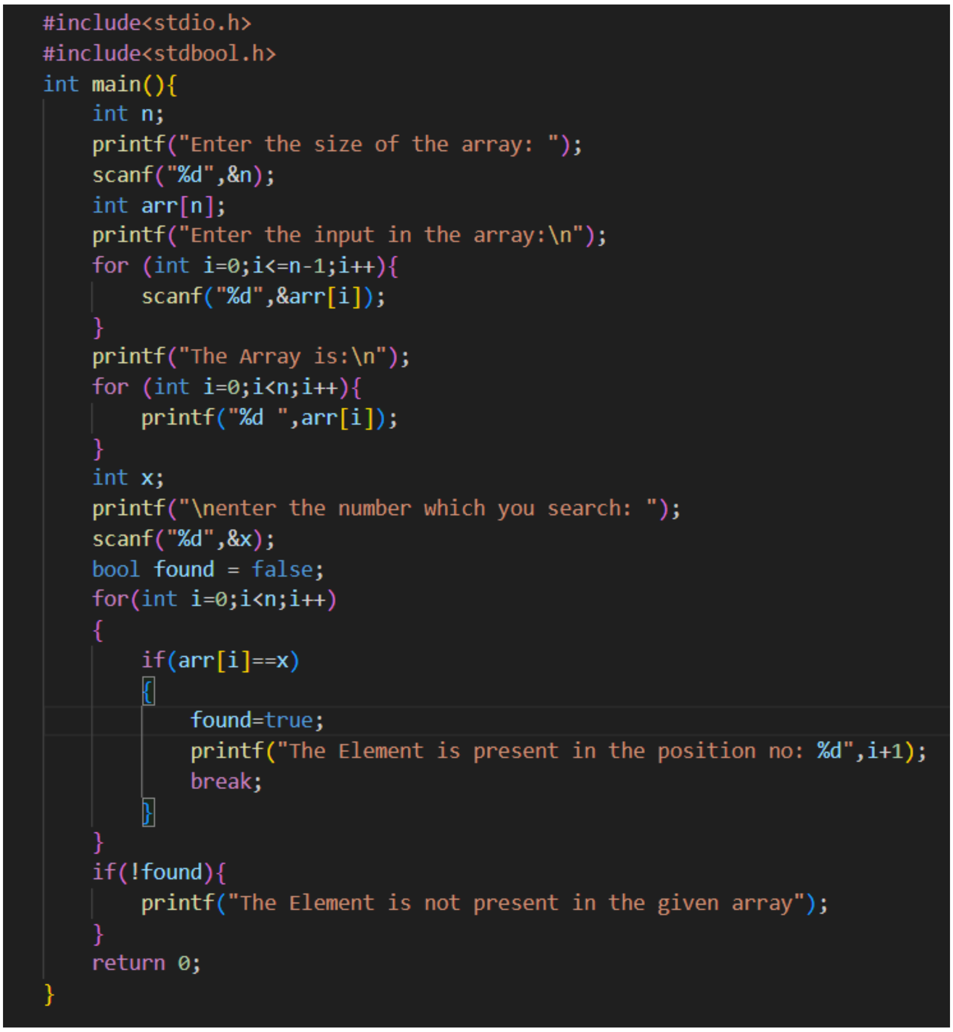


```
Enter the size of the array: 4
Enter the input of the Array:
23 12 65 34
The Array is:
23 12 65 34
The Bubble sort in Ascending Order is:
12 23 34 65
```

Question:13

Statement: Write a program to implement of Linear Search.

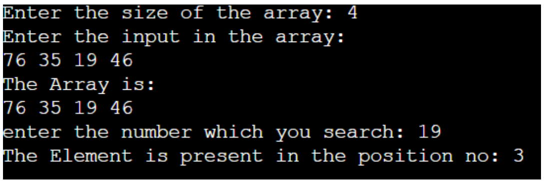
Source Code:



```
#include<stdio.h>
#include<stdbool.h>
int main(){
    int n;
    printf("Enter the size of the array: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter the input in the array:\n");
    for (int i=0;i<=n-1;i++){
        scanf("%d",&arr[i]);
    }
    printf("The Array is:\n");
    for (int i=0;i<n;i++){
        printf("%d ",arr[i]);
    }
    int x;
    printf("\nenter the number which you search: ");
    scanf("%d",&x);
    bool found = false;
    for(int i=0;i<n;i++)
    {
        if(arr[i]==x)
        {
            found=true;
            printf("The Element is present in the position no: %d",i+1);
            break;
        }
    }
    if(!found){
        printf("The Element is not present in the given array");
    }
    return 0;
}
```

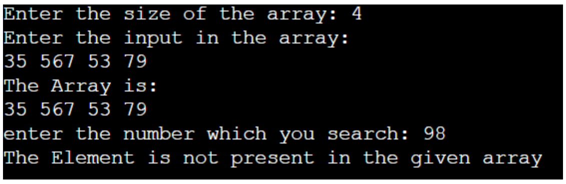
Output:

Case-1:



```
Enter the size of the array: 4
Enter the input in the array:
76 35 19 46
The Array is:
76 35 19 46
enter the number which you search: 19
The Element is present in the position no: 3
```

Case-2:

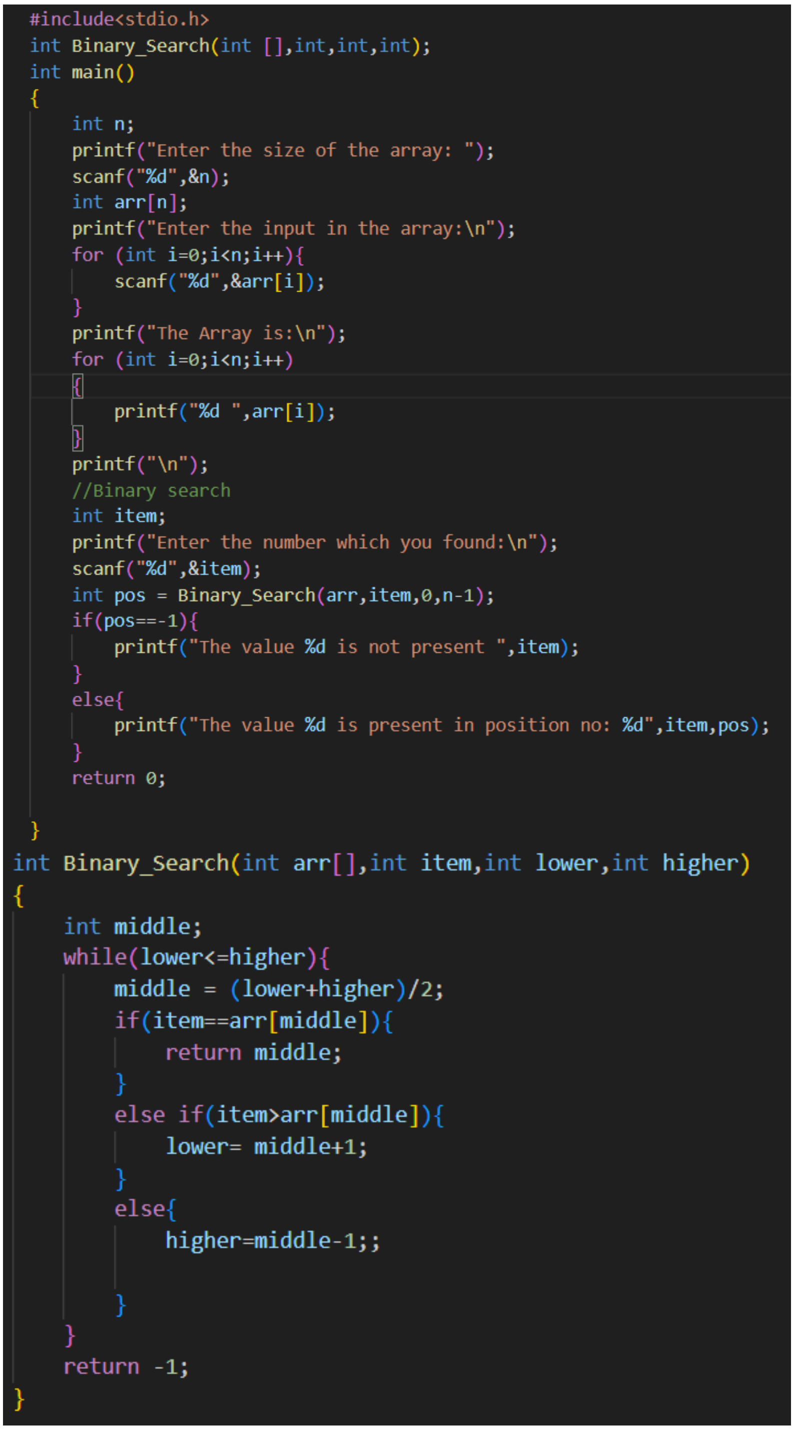


```
Enter the size of the array: 4
Enter the input in the array:
35 567 53 79
The Array is:
35 567 53 79
enter the number which you search: 98
The Element is not present in the given array
```

Question:14

Statement: Write a program to implement of Binary Search.

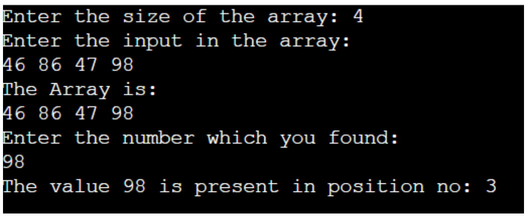
Source Code:



```c
#include<stdio.h>
int Binary_Search(int [],int,int,int);
int main()
{
    int n;
    printf("Enter the size of the array: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter the input in the array:\n");
    for (int i=0;i<n;i++){
        scanf("%d",&arr[i]);
    }
    printf("The Array is:\n");
    for (int i=0;i<n;i++)
    {
        printf("%d ",arr[i]);
    }
    printf("\n");
    //Binary search
    int item;
    printf("Enter the number which you found:\n");
    scanf("%d",&item);
    int pos = Binary_Search(arr,item,0,n-1);
    if(pos==-1){
        printf("The value %d is not present ",item);
    }
    else{
        printf("The value %d is present in position no: %d",item,pos);
    }
    return 0;

}
int Binary_Search(int arr[],int item,int lower,int higher)
{
    int middle;
    while(lower<=higher){
        middle = (lower+higher)/2;
        if(item==arr[middle]){
            return middle;
        }
        else if(item>arr[middle]){
            lower= middle+1;
        }
        else{
            higher=middle-1;;

        }
    }
    return -1;
}
```

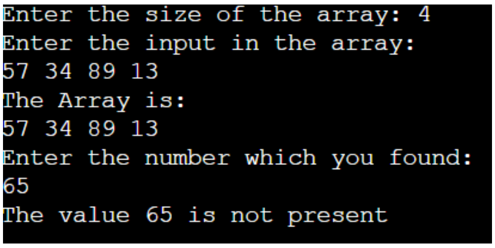
Output:

Case-1:



```
Enter the size of the array: 4
Enter the input in the array:
46 86 47 98
The Array is:
46 86 47 98
Enter the number which you found:
98
The value 98 is present in position no: 3
```

Case-2:

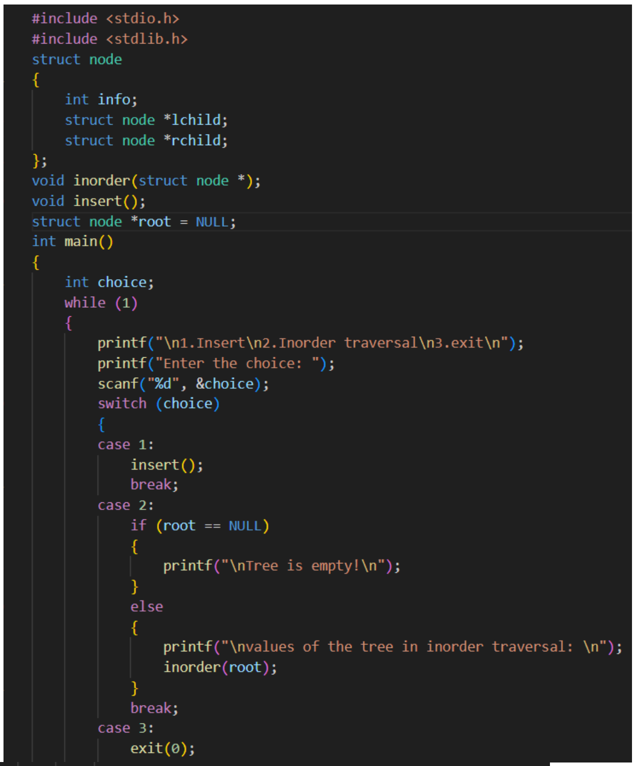


```
Enter the size of the array: 4
Enter the input in the array:
57 34 89 13
The Array is:
57 34 89 13
Enter the number which you found:
65
The value 65 is not present
```

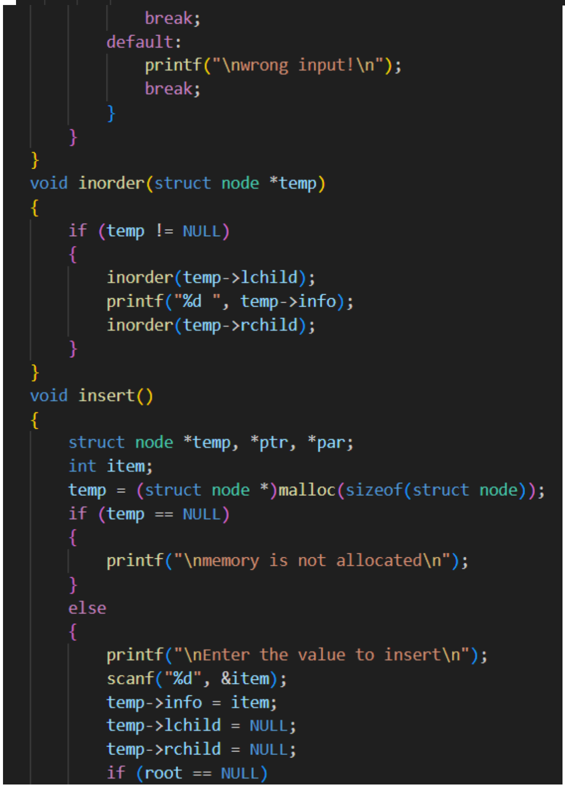
Question:15

Statement: Create a Binary Tree and perfrom a Inorder Traversal.

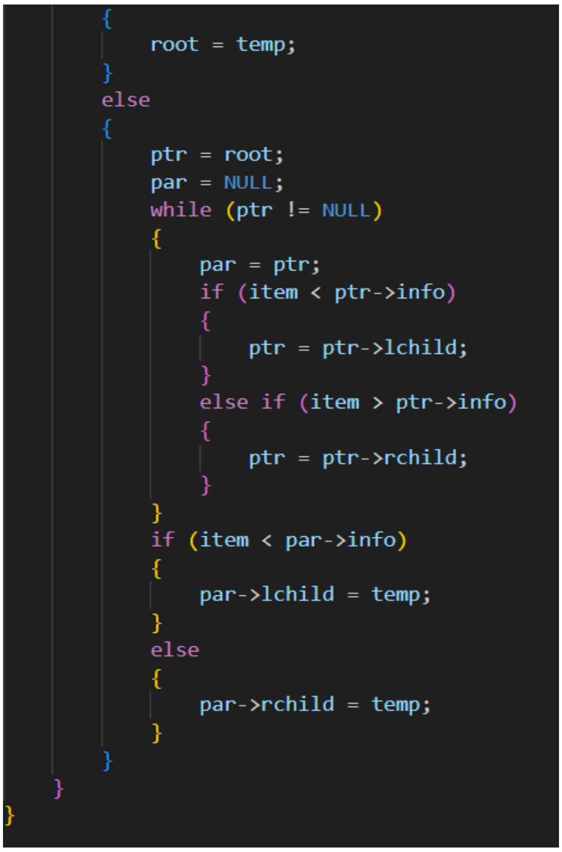
Source Code:



```
#include <stdio.h>
#include <stdlib.h>
struct node
{
    int info;
    struct node *lchild;
    struct node *rchild;
};
void inorder(struct node *);
void insert();
struct node *root = NULL;
int main()
{
    int choice;
    while (1)
    {
        printf("\n1.Insert\n2.Inorder traversal\n3.exit\n");
        printf("Enter the choice: ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            insert();
            break;
        case 2:
            if (root == NULL)
            {
                printf("\nTree is empty!\n");
            }
            else
            {
                printf("\nvalues of the tree in inorder traversal: \n");
                inorder(root);
            }
            break;
        case 3:
            exit(0);
```

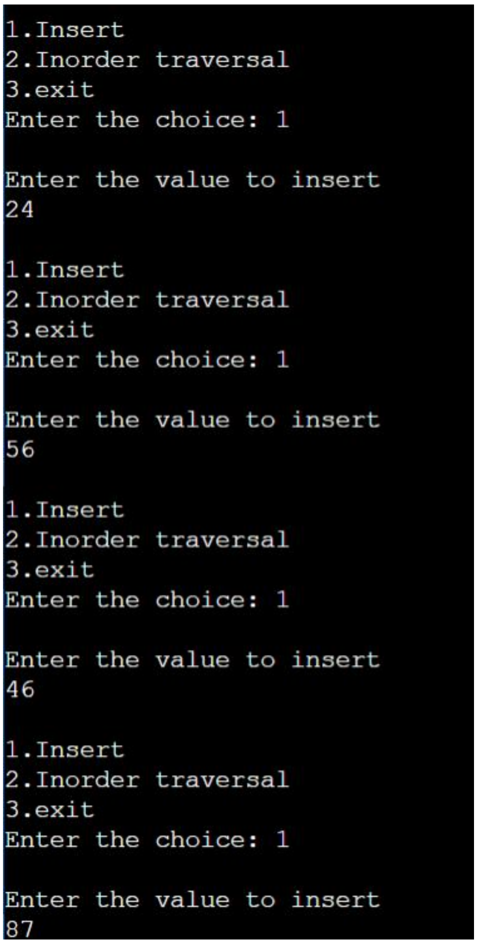


```
            break;
        default:
            printf("\nwrong input!\n");
            break;
        }
    }
}
void inorder(struct node *temp)
{
    if (temp != NULL)
    {
        inorder(temp->lchild);
        printf("%d ", temp->info);
        inorder(temp->rchild);
    }
}
void insert()
{
    struct node *temp, *ptr, *par;
    int item;
    temp = (struct node *)malloc(sizeof(struct node));
    if (temp == NULL)
    {
        printf("\nmemory is not allocated\n");
    }
    else
    {
        printf("\nEnter the value to insert\n");
        scanf("%d", &item);
        temp->info = item;
        temp->lchild = NULL;
        temp->rchild = NULL;
        if (root == NULL)
```

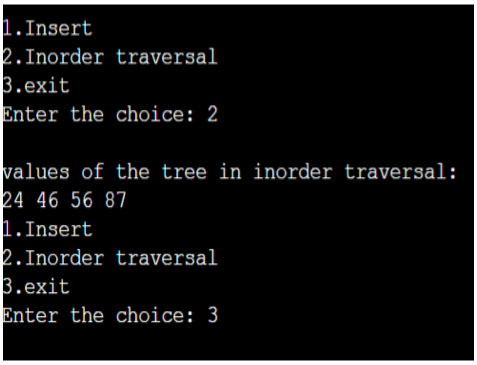


```
        {
            root = temp;
        }
        else
        {
            ptr = root;
            par = NULL;
            while (ptr != NULL)
            {
                par = ptr;
                if (item < ptr->info)
                {
                    ptr = ptr->lchild;
                }
                else if (item > ptr->info)
                {
                    ptr = ptr->rchild;
                }
            }
            if (item < par->info)
            {
                par->lchild = temp;
            }
            else
            {
                par->rchild = temp;
            }
        }
    }
}
```

Output:



```
1.Insert
2.Inorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
24

1.Insert
2.Inorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
56

1.Insert
2.Inorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
46

1.Insert
2.Inorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
87
```

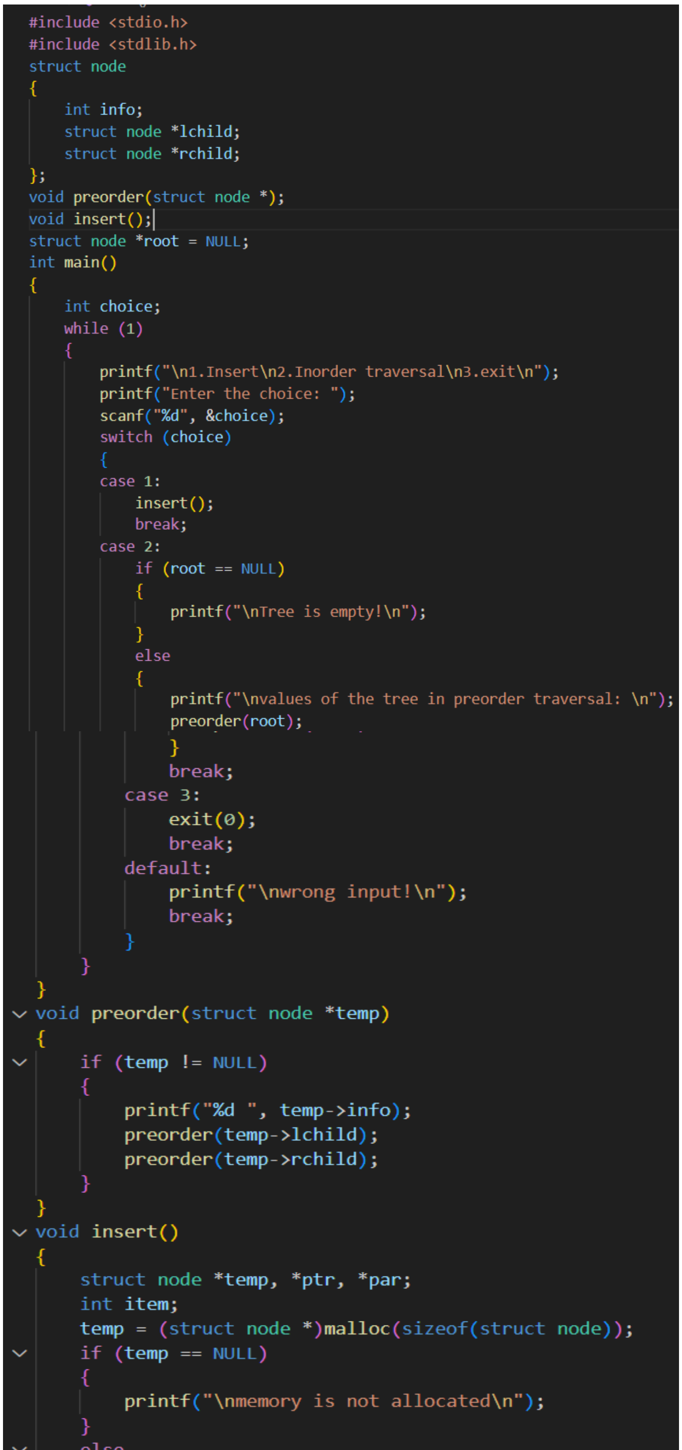


```
1.Insert
2.Inorder traversal
3.exit
Enter the choice: 2

values of the tree in inorder traversal:
24 46 56 87
1.Insert
2.Inorder traversal
3.exit
Enter the choice: 3
```

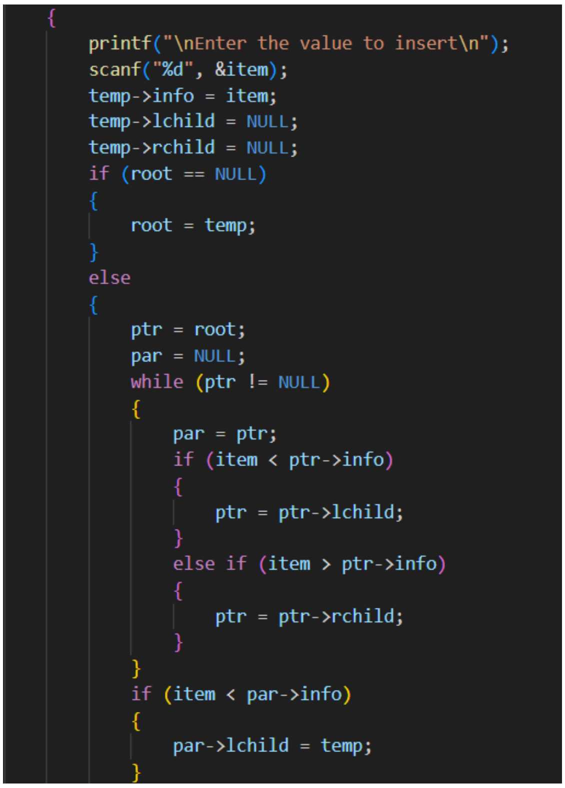
Question:16

Statement: Create a Binary Tree and perfrom a Preorder Traversal.

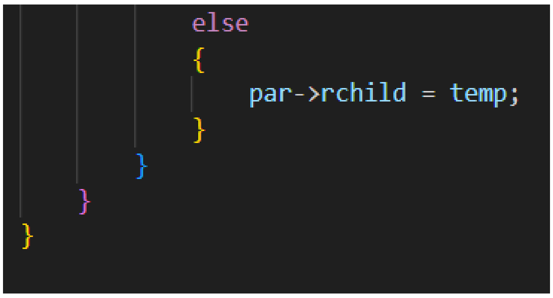
Source Code:



```c
#include <stdio.h>
#include <stdlib.h>
struct node
{
    int info;
    struct node *lchild;
    struct node *rchild;
};
void preorder(struct node *);
void insert();
struct node *root = NULL;
int main()
{
    int choice;
    while (1)
    {
        printf("\n1.Insert\n2.Inorder traversal\n3.exit\n");
        printf("Enter the choice: ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            insert();
            break;
        case 2:
            if (root == NULL)
            {
                printf("\nTree is empty!\n");
            }
            else
            {
                printf("\nvalues of the tree in preorder traversal: \n");
                preorder(root);
            }
            break;
        case 3:
            exit(0);
            break;
        default:
            printf("\nwrong input!\n");
            break;
        }
    }
}
void preorder(struct node *temp)
{
    if (temp != NULL)
    {
        printf("%d ", temp->info);
        preorder(temp->lchild);
        preorder(temp->rchild);
    }
}
void insert()
{
    struct node *temp, *ptr, *par;
    int item;
    temp = (struct node *)malloc(sizeof(struct node));
    if (temp == NULL)
    {
        printf("\nmemory is not allocated\n");
    }
    else
```

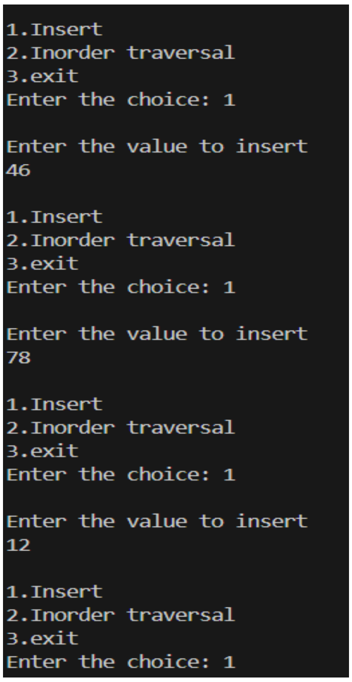


```
    {
        printf("\nEnter the value to insert\n");
        scanf("%d", &item);
        temp->info = item;
        temp->lchild = NULL;
        temp->rchild = NULL;
        if (root == NULL)
        {
            root = temp;
        }
        else
        {
            ptr = root;
            par = NULL;
            while (ptr != NULL)
            {
                par = ptr;
                if (item < ptr->info)
                {
                    ptr = ptr->lchild;
                }
                else if (item > ptr->info)
                {
                    ptr = ptr->rchild;
                }
            }
            if (item < par->info)
            {
                par->lchild = temp;
            }
```

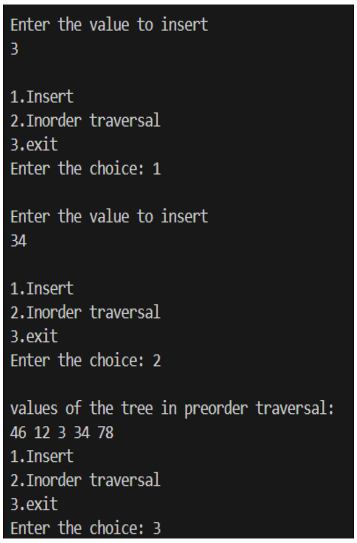


```
            else
            {
                par->rchild = temp;
            }
        }
    }
}
```

Output:



```
1.Insert
2.Inorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
46

1.Insert
2.Inorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
78

1.Insert
2.Inorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
12

1.Insert
2.Inorder traversal
3.exit
Enter the choice: 1
```

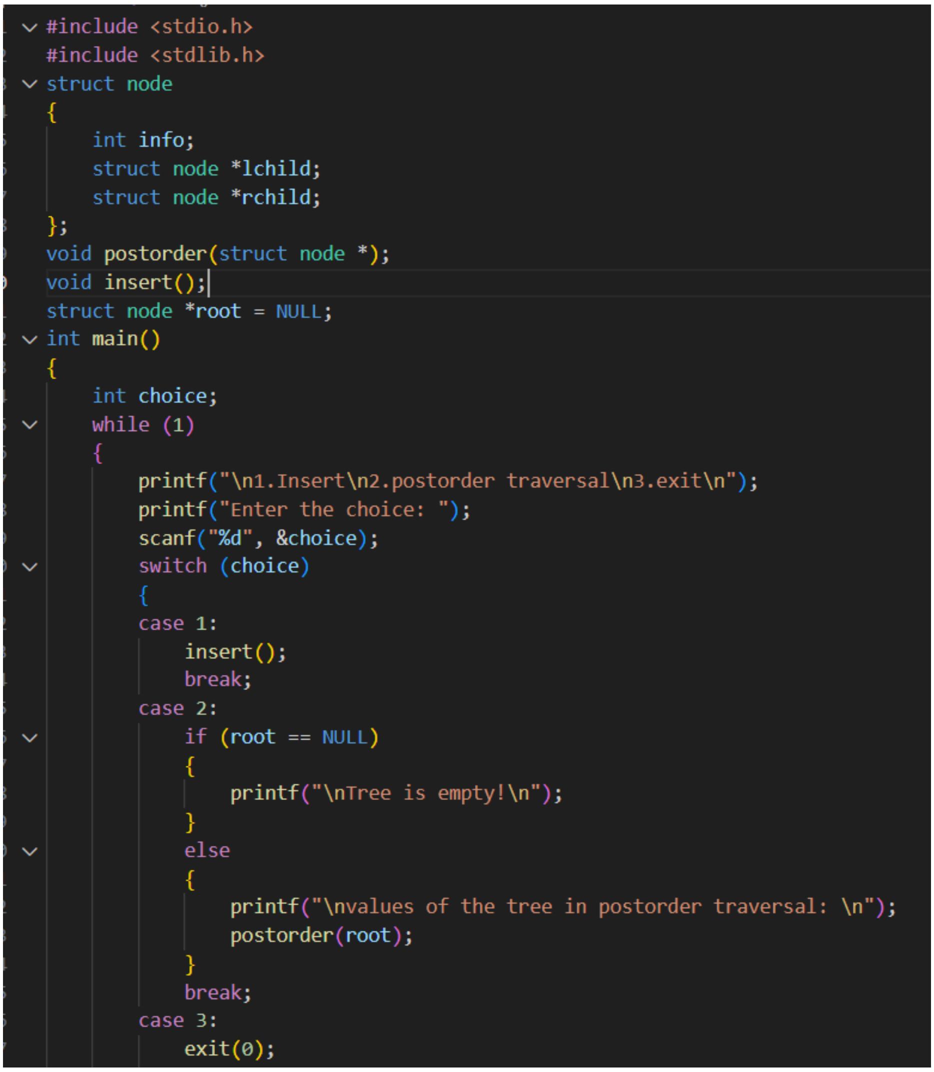


```
Enter the value to insert
3

1.Insert
2.Inorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
34

1.Insert
2.Inorder traversal
3.exit
Enter the choice: 2

values of the tree in preorder traversal:
46 12 3 34 78
1.Insert
2.Inorder traversal
3.exit
Enter the choice: 3
```

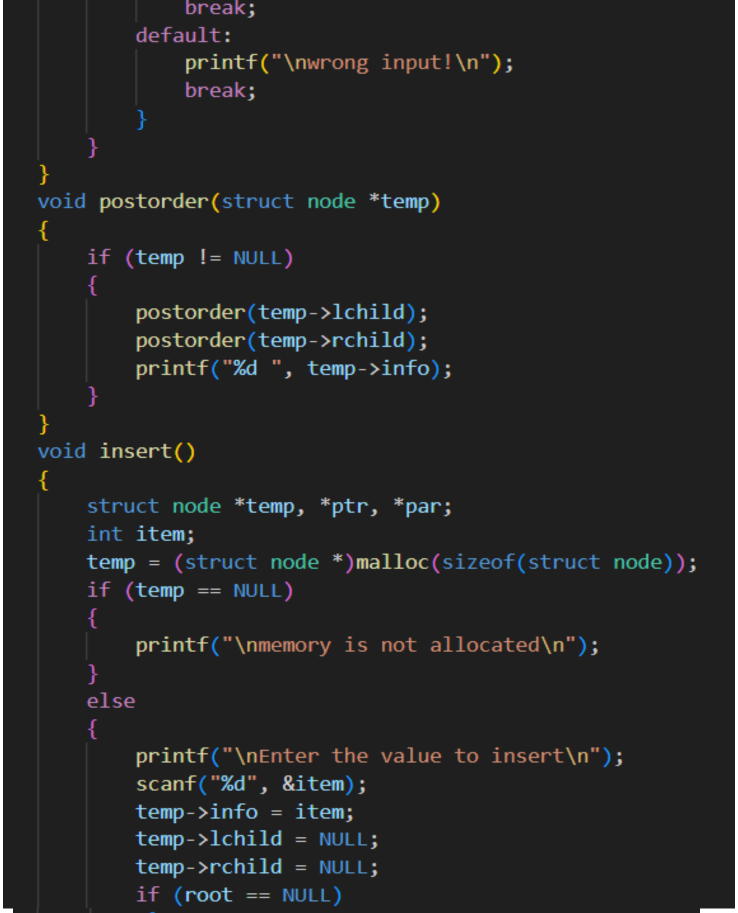
Question:17

Statement: Create a Binary Tree and perfrom a postorder Traversal.

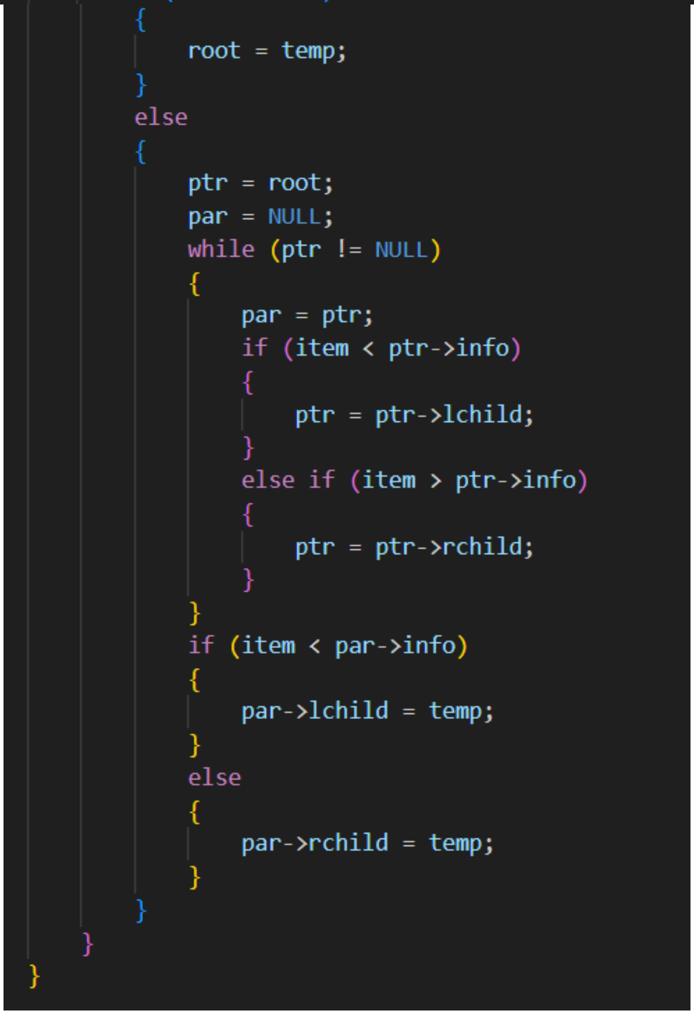
Source Code:



```
#include <stdio.h>
#include <stdlib.h>
struct node
{
    int info;
    struct node *lchild;
    struct node *rchild;
};
void postorder(struct node *);
void insert();
struct node *root = NULL;
int main()
{
    int choice;
    while (1)
    {
        printf("\n1.Insert\n2.postorder traversal\n3.exit\n");
        printf("Enter the choice: ");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            insert();
            break;
        case 2:
            if (root == NULL)
            {
                printf("\nTree is empty!\n");
            }
            else
            {
                printf("\nvalues of the tree in postorder traversal: \n");
                postorder(root);
            }
            break;
        case 3:
            exit(0);
```

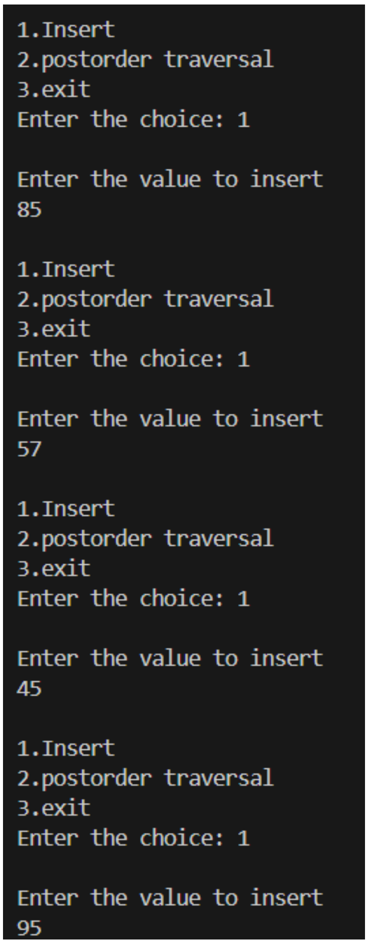


```c
            break;
        default:
            printf("\nwrong input!\n");
            break;
        }
    }
}
void postorder(struct node *temp)
{
    if (temp != NULL)
    {
        postorder(temp->lchild);
        postorder(temp->rchild);
        printf("%d ", temp->info);
    }
}
void insert()
{
    struct node *temp, *ptr, *par;
    int item;
    temp = (struct node *)malloc(sizeof(struct node));
    if (temp == NULL)
    {
        printf("\nmemory is not allocated\n");
    }
    else
    {
        printf("\nEnter the value to insert\n");
        scanf("%d", &item);
        temp->info = item;
        temp->lchild = NULL;
        temp->rchild = NULL;
        if (root == NULL)
```

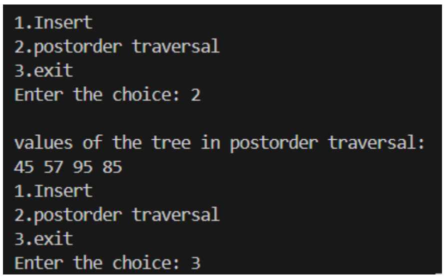


```c
        {
            root = temp;
        }
        else
        {
            ptr = root;
            par = NULL;
            while (ptr != NULL)
            {
                par = ptr;
                if (item < ptr->info)
                {
                    ptr = ptr->lchild;
                }
                else if (item > ptr->info)
                {
                    ptr = ptr->rchild;
                }
            }
            if (item < par->info)
            {
                par->lchild = temp;
            }
            else
            {
                par->rchild = temp;
            }
        }
    }
}
```

Output:



```
1.Insert
2.postorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
85

1.Insert
2.postorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
57

1.Insert
2.postorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
45

1.Insert
2.postorder traversal
3.exit
Enter the choice: 1

Enter the value to insert
95
```



```
1.Insert
2.postorder traversal
3.exit
Enter the choice: 2

values of the tree in postorder traversal:
45 57 95 85
1.Insert
2.postorder traversal
3.exit
Enter the choice: 3
```