EX 4: Implementation of stack using array and linked list

Stack using array

```
#include <stdio.h>
#include <stdlib.h>
#define Size 25
int Stack[Size];
int Top=-1;


void Push(int);
int Pop();
void Display();
int IsFull();
int IsEmpty();
int Peak();

int IsFull(){
	if (Top==Size-1)
		return 1;
	else
		return 0;
}

int IsEmpty(){
	if (Top==-1)
		return 1;
	else
		return 0;
}

void Push(int val){
	if (!IsFull())
		{
			Top=Top+1;
			Stack[Top]=val;
		}
	else
	{
		printf("Stack Overflow");

	}
```

```
}

int Pop(){
	if (!IsEmpty()){
		int del=Stack[Top];
		Top=Top-1;
		return del;
	}
	else
	{
		printf("Stack Underflow");
		return -1;
	}
}


void Display(){
	int a=Top;
	if (!IsEmpty()){
		for (int i=a;i>=0;i--)
			printf("%d ",Stack[i]);
	}
	else
	{
		printf("Stack Underflow");
		return ;
	}
}

int Peak(){
	if (!IsEmpty())
		return Stack[Top];
	else
	{
		printf("Stack Underflow");
		return -1 ;
	}
}


int main(){

	int choice,t=1,n;
	while (t==1)
```

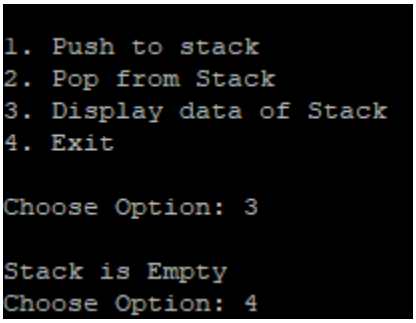
```
    {
        printf("\n\nMENU FOR STACK IMPLEMENTATION USING ARRAY:");
        printf("\n1.Push an element.\n2.Pop an element.\n3.Return Top most
element.\n4.Display.\n5.EXIT\n");
        printf("\nEnter your choice:");
        scanf("%d",&choice);
        switch (choice)
        {
            case 1:

                printf("Enter an element:");
                scanf("%d",&n);
                Push(n);
            break;


            case 2:
        {
            n=Pop();
            printf("%d",n);
            break;
        }
        case 3:
        {
            printf("%d",Peak());
            break;
        }
        case 4:
        {
            Display();
            break;
        }
        case 5:
        {
            t=0;
            break;
        }
        default:
        {
   printf("INVALID CHOICE");
            break;
        }
    }
    }
```

}

OUTPUT:



Stack using linked list

```
#include<stdio.h>
#include<stdlib.h>

struct node
{
   int data;
   struct node *link;
}*first=NULL;


void push(int);
void pop();
void Top();
void display1();

void push(int data)
{
   struct node *newnode;
   newnode=(struct node*)malloc(sizeof(struct node));
   newnode->data=data;
   if(first==NULL){
      newnode->link=NULL;
      first=newnode;
   }
   else
   {
      newnode->link=first;
      first=newnode;
   }
```

```
    printf("Data inserted\n");
}

void pop()
{
    struct node *temp=NULL;
    temp=first;
    if(first==NULL){
    printf("INVALID OPERATION");
    }
    else{
        printf("\n%d is the popped element",temp->data);
        first=temp->link;
        free(temp);
        temp=NULL;
    }
}

void Top()
{
    if(first!=NULL)
    printf("%d is the top element",first->data);
    else
    printf("\nNo data inside");

}

void display1()
{
{
    struct node *temp=NULL;
    temp=first;
    if(temp!=NULL){
        while(temp!=NULL)
    {
        printf("%d ",temp->data);
        temp=temp->link;
    }
}
else{
    printf("\nNo data inside");
}
}
}
```

```
int main()
{
    int ch,n;
     printf("MENU DRIVEN PROGRAM");
     printf("0. Exit\n");
    printf("1. Push\n");
    printf("2. Pop\n");
    printf("3. Return Top element\n");
    printf("4. Display\n");

    while(1){
    printf("\nEnter your choice :  ");
    scanf("%d",&ch);
    switch (ch)
    {
    case 1:
    printf("\nEnter data to push : ");
    scanf("%d",&n);
    push_beg(n);
    break;

    case 2:
    pop_beg();
    break;

    case 3:
    top_elemt();
    break;

    case 4:
    display1();
    break;

    default:
    printf("\nMENU EXITED");
    break;
    }
    if(ch==0){
        break;
    }
    else{
    continue;
    }
```

```
printf("\nProgram exited");
}
}
```

OUTPUT: