n.c

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

typedef struct node{
  int data;
  struct node *link;
}node;

node *top=NULL;



void push()
{
 node *temp;
temp=(node *)malloc(sizeof(node));
 printf("Enter node element\n");
 scanf("%d",&temp->data);
 temp->link=NULL;

 if(top==NULL)
 {
     top=temp;
 }
 else
 {
  temp->link=top;
  top=temp;
 }
}

void pop()
{

node *temp;

```

```
}

void pop()
{

node *temp;

if(top==NULL)
{
  printf("Stack is empty\n");
}

else
{
 temp=top;
top=temp->link;
temp->link=NULL;
free(temp);
}

}


void display()
{
 node *temp=top;
 if(temp==NULL)
 {
   printf("Stack is empty\n");
 }
 else
 {
   while(temp!=NULL)
   {
     printf("%d  ",temp->data);
     temp=temp->link;
```

n.c

```
}


void display()
{
 node *temp=top;
 if(temp==NULL)
 {
   printf("Stack is empty\n");
 }
 else
 {
   while(temp!=NULL)
   {
     printf("%d  ",temp->data);
     temp=temp->link;
   }
   printf("\n");
 }
}

int main()
{

 int op,len;
   printf("\n1.Push\n2.Pop\n3.Display\n4.Exit\n");
 while(1)
{ printf("Enter the operation ");

  scanf("%d",&op);
  switch (op)
  {
  case 1:push();
    break;
  case 2: pop();
    break;
```

ain.c

```
 }
 else
 {
   while(temp!=NULL)
   {
     printf("%d  ",temp->data);
     temp=temp->link;
   }
   printf("\n");
 }
}

int main()
{

 int op,len;
   printf("\n1.Push\n2.Pop\n3.Display\n4.Exit\n");
 while(1)
{ printf("Enter the operation ");

  scanf("%d",&op);
  switch (op)
  {
  case 1:push();
    break;
  case 2: pop();
    break;
  case 3: display();
    break;
  case 4: exit(0);
     break;
  default: printf("No such operation\n");
  }
 }
return 0;
}
```

input

```
1.Push
2.Pop
3.Display
4.Exit
Enter the operation 1
Enter node element
12
Enter the operation 1
Enter node element
23
Enter the operation 1
Enter node element
34
Enter the operation 1
Enter node element
6
Enter the operation 3
6  34  23  12
Enter the operation 2
Enter the operation 2
Enter the operation 2
Enter the operation 3
12
Enter the operation 2
Enter the operation 3
Stack is empty
Enter the operation
```