**Experiment No-6A**

**Aim**: - Write a program in C to implement stack using linked list.

**Theory**: -

A stack can be easily implemented through the linked list. In stack Implementation, a stack contains a top pointer. which is “head” of the stack where pushing and popping items happens at the head of the list. first node have null in link field and second node link have first node address in link field and so on and last node address in “top” pointer.

The main advantage of using linked list over an arrays is that it is possible to implements a stack that can shrink or grow as much as needed. In using array will put a restriction to the maximum capacity of the array which can lead to stack overflow. Here each new node will be dynamically allocate. so overflow is not possible.

Stack Operations:

1. [**push()**](https://www.geeksforgeeks.org/stack-push-and-pop-in-c-stl/)**:** Insert the element into linked list nothing but which is the top node of Stack.
2. [**pop()**](https://www.geeksforgeeks.org/stack-push-and-pop-in-c-stl/)**:** Return top element from the Stack and move the top pointer to the second node of linked list or Stack.
3. [**peek()**](https://www.geeksforgeeks.org/stack-peek-method-in-java/)**:** Return the top element.
4. **display():** Print all element of Stack.

**CODE:-**

#include<stdio.h>

#include<stdlib.h>

struct stack

{

int num;

struct stack \*next;

};

struct stack \*top=NULL;

void push(int e)

{

struct stack \*ptr;

ptr=(struct stack \*)malloc(sizeof(struct stack));

ptr->num=e;

if(top==NULL)

{

ptr->next=NULL;

top=ptr;

}

else

{

ptr->next=top;

top=ptr;

}

}

void pop()

{

struct stack \*ptr=top;

if(top==NULL)

printf("Stack is Empty\n");

else

{

top=top->next;

printf("The deleted element is %d\n",ptr->num);

free(ptr);

}

}

void peek()

{

if(top!=NULL)

printf("%d\n",top->num);

}

void display()

{

struct stack \*ptr=top;

if(top==NULL)

printf("Stack is Empty\n");

else

{

while(ptr!=NULL)

{

printf("%d\n",ptr->num);

ptr=ptr->next;

}

}

}

int main(void)

{

int ch,val;

do

{

printf("Enter your choice:\n");

printf("1.Push\n2.Pop\n3.Peek\n4.Display\n");

scanf("%d",&ch);

switch(ch)

{

case 1:

printf("Enter the element:\n");

scanf("%d",&val);

push(val);

break;

case 2:

pop();

break;

case 3:

peek();

break;

case 4:

display();

break;

default:

printf("Invalid Input\n");

}

}while(ch<5);

return 0;

}

**OUTPUT: -**





