**Program 3**

**Aim :** To implement a stack and its basic operations.

Theory : Stack is a linear data structure which follows a particular order in which the operations are performed. The order may be LIFO(Last In First Out) or FILO(First In Last Out). There are many real-life examples of a stack. Consider an example of plates stacked over one another in the canteen. The plate which is at the top is the first one to be removed, i.e. the plate which has been placed at the bottommost position remains in the stack for the longest period of time. So, it can be simply seen to follow LIFO(Last In First Out)/FILO(First In Last Out) order.

In linked list implementation of stack, each new element is inserted at the front of the linked list. This way we just remove the head of the linked list to remove the top element.

**Algorithm :**

* Declare and define node structure with 2 data members – int data and struct node \* next
* Define a node pointer top and set it to NULL

Push(int value)

1. Create a newnode and set its data to value
2. Set newnode->next = top
3. Set top = newnode

Pop()

1. If stack is empty then

print(“Underflow”) and terminate the function

1. Set a node pointer temp = top
2. Set top = top->next
3. Free memory used by temp (free(temp))

Display(node\*top)

1. If stack is empty then

Print(“Stack is Empty”)

return

1. Repeat steps 3 and 4 while top!=NULL
2. Print(top->data)
3. Set top = top->next
4. return

Top(node \* top)

1. if stack is empty then

print(“Stack is Empty”)

terminate the function

1. Else return top->data

isEmpty(node\*top)

1. return top == NULL

**Program :**

#include<stdio.h>

#include<stdlib.h>

#include<limits.h>

struct node{

int data;

struct node\*next;

};

void pop(struct node \*\*);

void traverse(struct node \*);

int top(struct node \*);

int isEmpty(struct node \* );

void push(struct node \*\*,int value);

int main(){

struct node \* stack = NULL;

printf("isEmpty = %d\n",isEmpty(stack));

push(&stack,10);

push(&stack,22);

push(&stack,298);

push(&stack,23);

printf("%d popped from stack\n", top(stack));

pop(&stack);

printf("isEmpty = %d\n",isEmpty(stack));

printf("%d is at top of the stack\n",top(stack));

printf("Elements in stack : ");

traverse(stack);

return 0;

}

void pop(struct node \*\* stack){

if(isEmpty(\*stack)){

printf("Stack is empty\n");

return;

}

struct node \* temp = \*stack;

\*stack = (\*stack)->next;

free(temp);

}

void traverse(struct node \* top){

while(top){

printf("%d ",top->data);

top = top->next;

}

printf("\n");

}

int top(struct node \* top){

if(top == NULL){

printf("Stack is empty\n");

return INT\_MIN;

}

return top->data;

}

int isEmpty(struct node \* top){

return top == NULL;

}

void push(struct node \*\* stack, int value){

struct node \* newNode = (struct node \*)malloc(sizeof(struct node));

newNode->data = value;

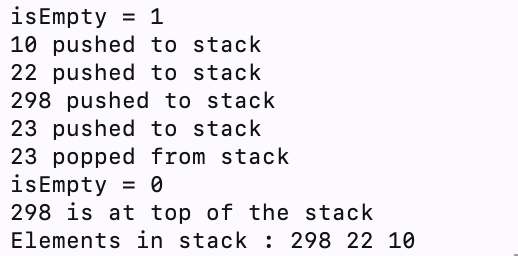
newNode->next = \*stack;

\*stack = newNode;

printf("%d pushed to stack\n", value);

}

**OUTPUT :**



Learning : We learnt about implementation of stack as a linked list and its basic operations.