**DSA Lab Assignment 1**

**Stack**

Dibyajyoti Mondal

Roll No: 001910701029

1. **Fixed memory implementation of Stack**

**Code:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<math.h>

#include<stdbool.h>

/\* Author : Dibyajyoti Mondal

B.E. ETCE UG2 DSA LAB\*/

/// defining static array

/// I have kept the stack size small so that it can be tested easily for overflow condition.

#define MAX\_SIZE 10

int stack[MAX\_SIZE];

/// top denoting top of stack

int top=-1;

/// push function

void push(int val)

{

if(top==MAX\_SIZE-1)

{

printf("Stack Overflow!!!\n");

return;

}

else

{

top++;

stack[top]=val;

print\_stack();

return;

}

}

/// pop function

void pop()

{

if(top==-1)

{

printf("Stack Underflow!!!\n");

return;

}

else

{

top--;

print\_stack();

return;

}

}

/// print contents of stack

void print\_stack()

{

for(int i=0;i<=top;i++)

printf("%d ",stack[i]);

printf("\n");

}

int main()

{

printf("====================Welcome to Stack Using Array==================\n\n\n");

printf("Push: 1 number\nPop: 0\n");

while(true)

{

int x,y;

printf("Enter a number to perform operation\n");

scanf("%d",&x);

if(x==1)

{

scanf("%d",&y);

push(y);

}

else if(x==0)

pop();

else

printf("Invalid Input !!!\n");

}

return 0;

}

1. **Dynamic memory implementation of Stack**

**Code:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<math.h>

#include<stdbool.h>

/\* Author : Dibyajyoti Mondal

B.E. ETCE UG2 DSA LAB\*/

/// implementing stack using linked list

/// push means insertion of new head node and pop means deletion of head node

struct Node{

int val;

struct Node \*next;

};

typedef struct Node Node;

/// head is null initially

Node \*head=NULL;

/// push into stack

void push(int value)

{

/// dynamic memory allocation

Node \*New\_Node=(Node \*)malloc(sizeof(Node));

if(!New\_Node)

{

printf("Stack Overflow!!! ERROR: No more heap space available\n");

return;

}

New\_Node->val=value;

New\_Node->next=NULL;

if(head==NULL)

head=New\_Node;

else

{

New\_Node->next=head;

head=New\_Node;

}

print\_stack();

return;

}

/// pop the topmost element

void pop()

{

/// deleting the current head and redefining it

if(head==NULL)

{

printf("Stack Underflow!!!\n");

return;

}

else

{

Node \*temp\_node=head;

head=head->next;

free(temp\_node);

print\_stack();

return;

}

}

/// printing the stack

void print\_stack()

{

if(head==NULL)

{

printf("Stack Empty!!!\n");

}

else

{

Node \*temp=head;

while(temp!=NULL)

{

printf("%d ",temp->val);

temp=temp->next;

}

printf("\n");

}

return;

}

int main()

{

printf("====================Welcome to Stack Using Linked List==================\n\n\n");

printf("Push: 1 number\nPop: 0\n");

while(true)

{

int x,y;

printf("Enter a number to perform operation\n");

scanf("%d",&x);

if(x==1)

{

scanf("%d",&y);

push(y);

}

else if(x==0)

pop();

else

printf("Invalid Input !!!\n");

}

return 0;

}