**Experiment No. 1**

**Title :** Implementation of Stack Linear Data Structure

**Problem Statement :** Implementing linear data structure stack using array with functions

Push()

Pop()

Display()

isFull()

isEmpty()

**Algorithm :**

**S1 :** Start

**S2 :** Declare an array along with the size, a variable X is declared to keep a track of the index of stack, a choice variable

**S3 :** Call the functions in a switch statement using choice variable.

**S4 :** In push function check if stack is FULL if not the increment the value of X.Now, with that as index put the value taken by user in array

**S5 :** In pop function check if stack is EMPTY if not then decrement value of X.

**S6 :** In display function display the content of stack.

**S7 :** In isFull function check if X value is equal to that of size of stack.

**S8 :** In isEmpty function check if X value is equal to -1

**S9 :** Stop

**Code :**

#include<stdio.h>

# define MAX 5

int stack[5]; //Declare array stack

int X = -1;

int i,ch;

int isfull() //isfull function

{

if(X == MAX-1)

{

printf("Stack is FULL\n");

return(1);

}

else

{

printf("Stack is NOT FULL\n");

return(0);

}

}

int isempty()

{

if(X == -1)

{

printf("Stack is EMPTY\n");

return(1);

}

else

{

printf("Stack is NOT EMPTY\n");

return(0);

}

}

void push()

{

int var;

if(!isfull())

{

printf("Enter the element to be pushed : ");

scanf("%d",&var);

X = X + 1;

stack[X] = var;

printf("%d is pushed\n",stack[X]);

}

}

void pop()

{

if(!isempty())

{

printf("%d is popped\n",stack[X]);

X = X - 1;

}

}

void display()

{

for(i=X;i>-1;--i)

{

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

}

}

int main()

{

start :

printf("Enter your choice :\n1.PUSH\t2.POP\t3.isEmpty\t4.isFull\t5.DISPLAY\nchoice : ");

scanf("%d",&ch);

switch(ch)

{

case 1 :

push();

break;

case 2 :

pop();

break;

case 3 :

isempty();

break;

case 4 :

isfull();

break;

case 5 :

display();

break;

default :

printf("Enter the correct choice\n");

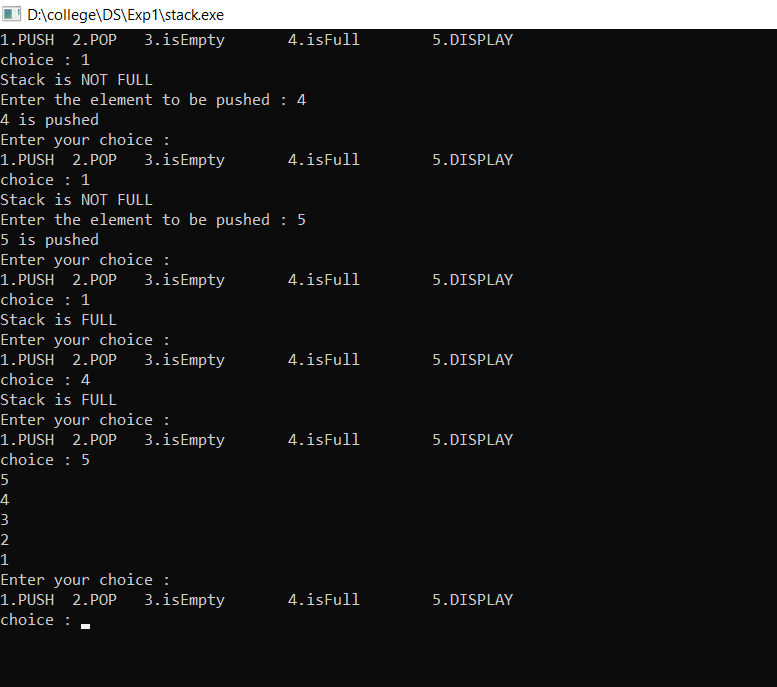
break;

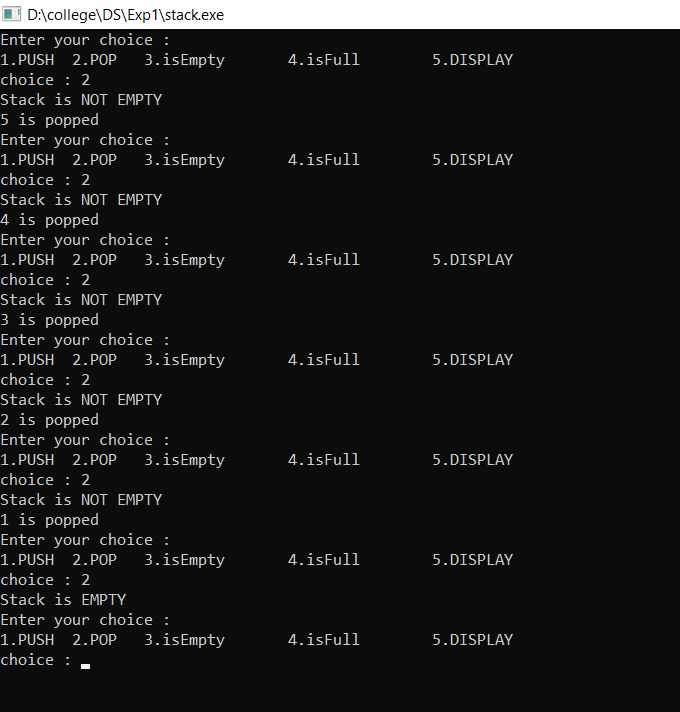
}

goto start;

}

**Output :**

****

****

**Analysis :**

* The stack size is limited hence operations is also limited.
* Operations of the stack is controlled by only one variable X.