**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 top variable to keep a track of the index of stack, a choice variable

**S3 :** Call the functions in a switch statement according to the choice value

**S4 :** In push function of type void check if stack is FULL if not the increment the top value and with that as index put the value taken by user in array

**S5 :** In pop function check if stack is EMPTY if not the decrement top value

**S6 :** In display function we display the stack by decrementing for loop starting from top to least index

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

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

**S9 :** Stop

**Code :**

#include<stdio.h>

# define MAX 5

int stack[5];

int top = -1;

int i,ch;

int isfull()

{

if(top == MAX-1)

{

printf("Stack is FULL\n");

return(1);

}

else

{

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

return(0);

}

}

int isempty()

{

if(top == -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);

top = top + 1;

stack[top] = var;

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

}

}

void pop()

{

if(!isempty())

{

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

top = top - 1;

}

}

void display()

{

for(i=top;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\tchoice : ");

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");

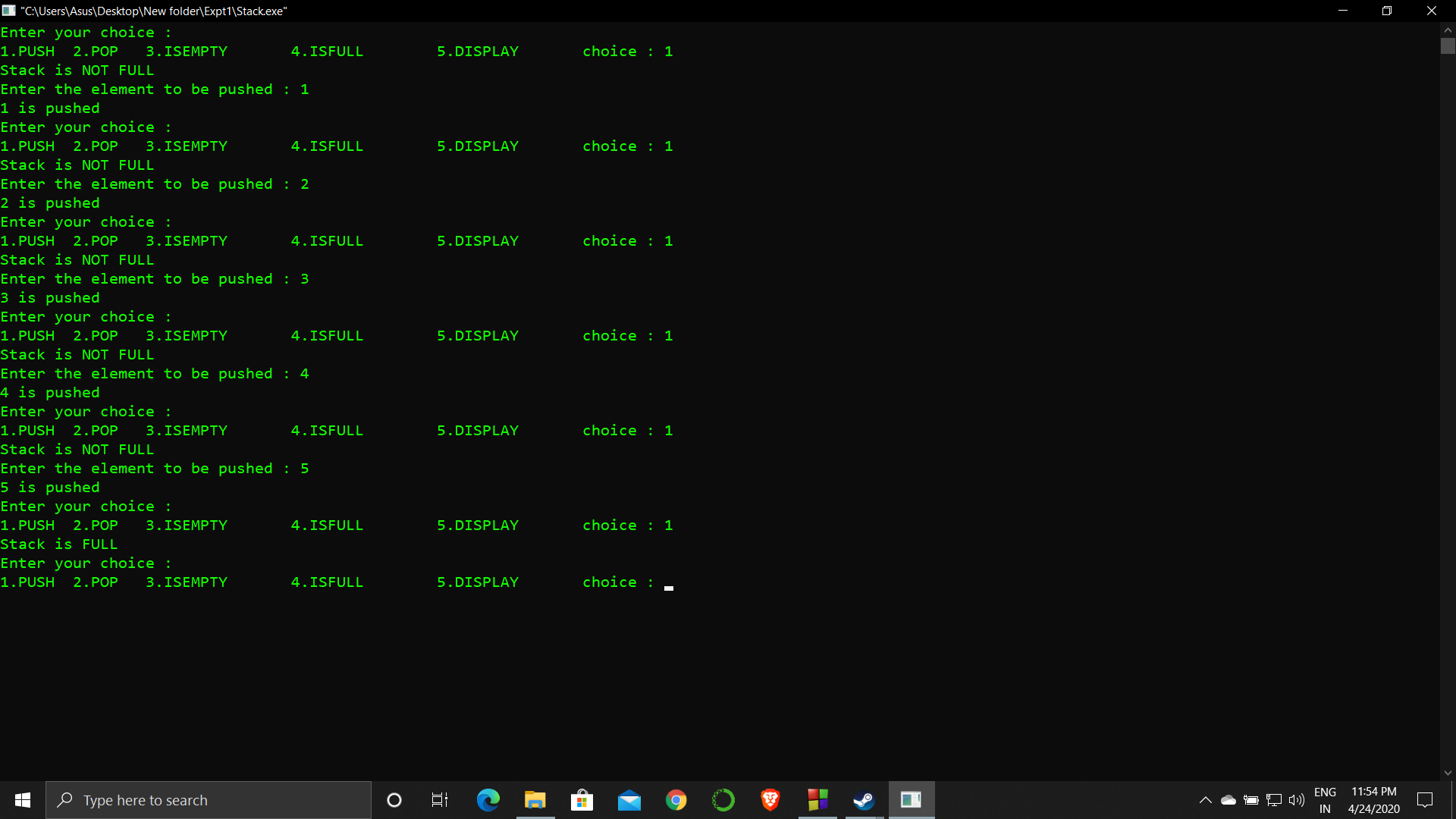
break;

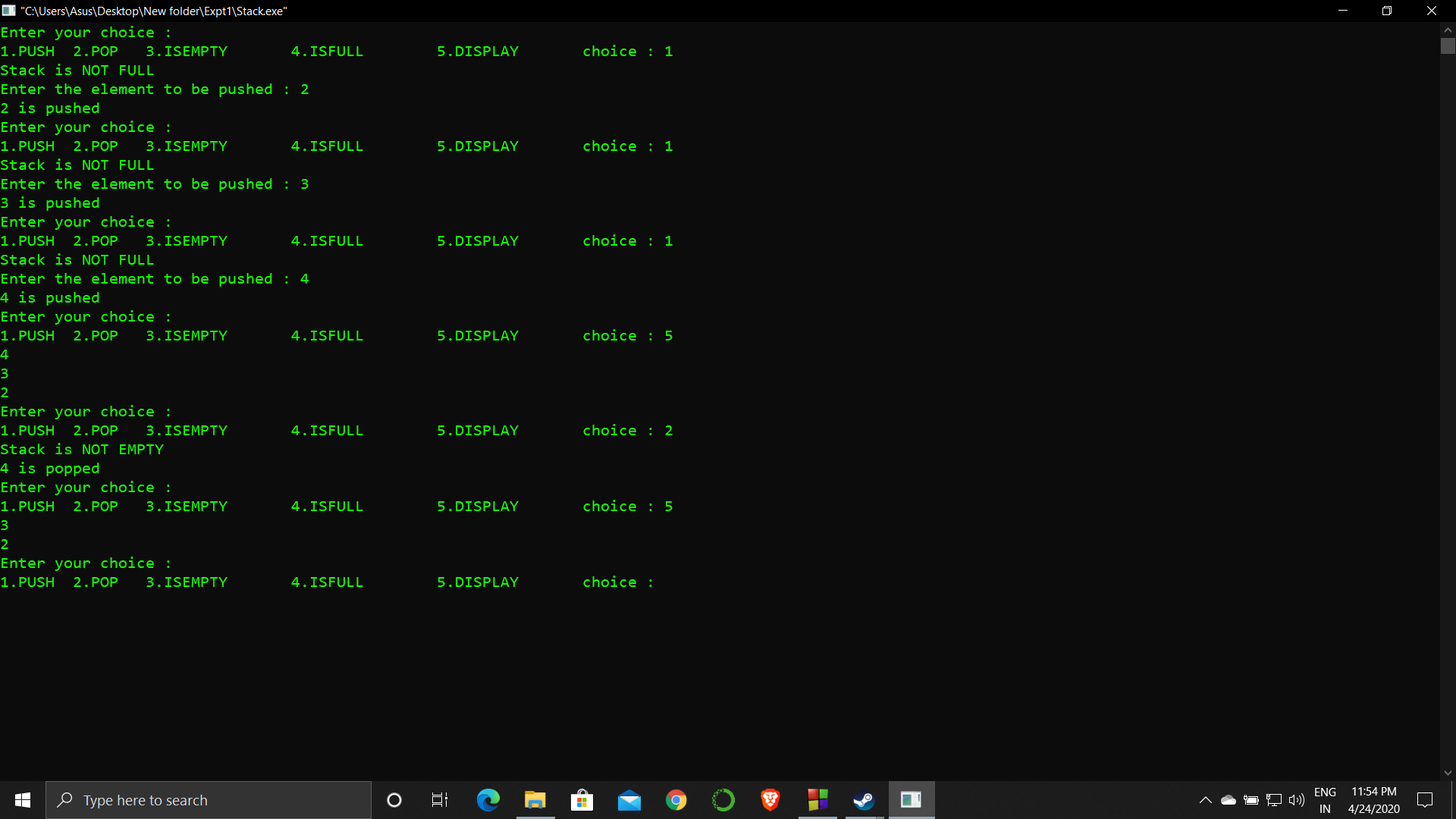
}

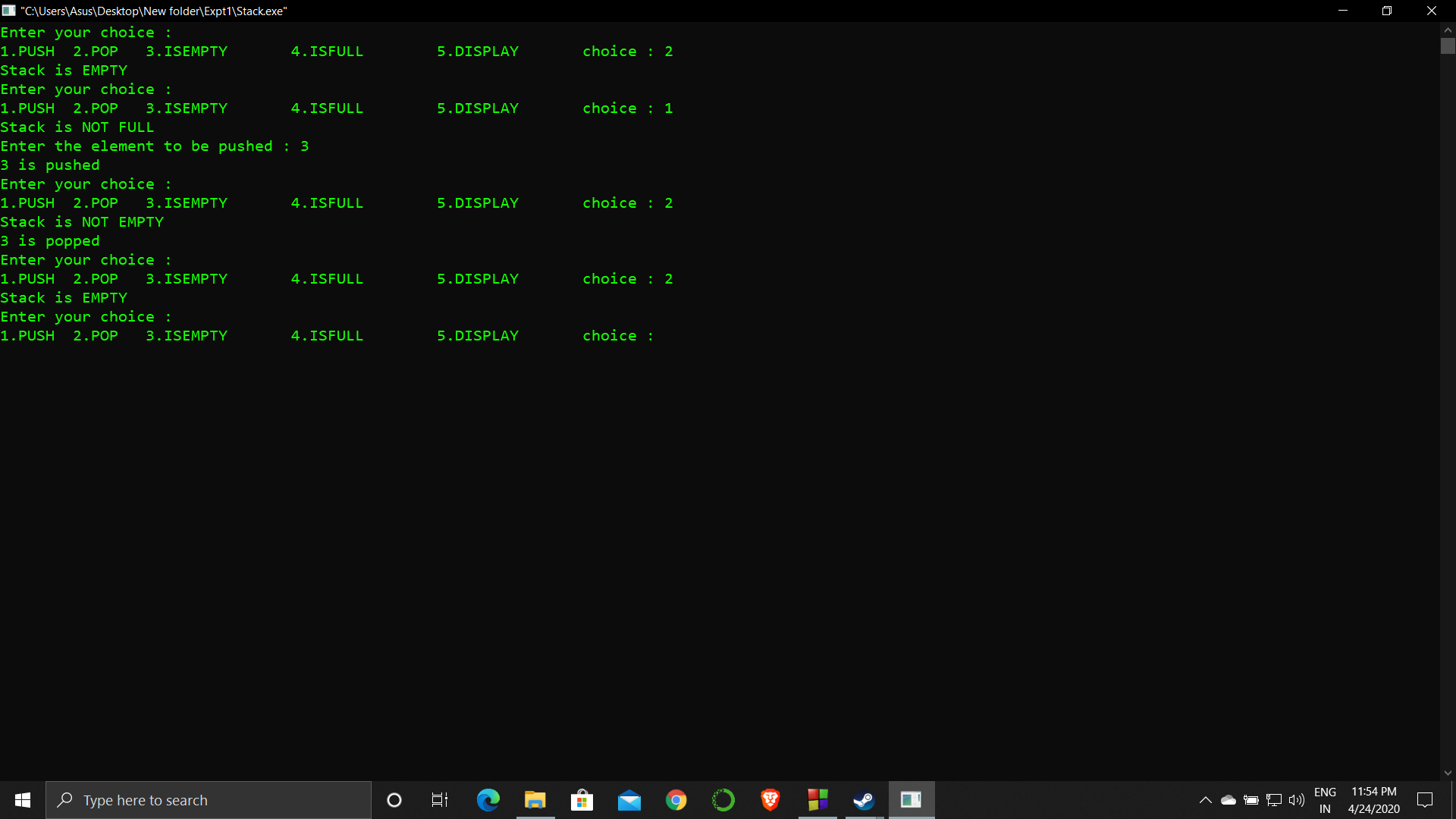
goto start;

}

**Output :**

****

****

****

**Analysis :**

The stack size is limited which makes the operations related to it also limited.

If the top value gets compromised the stack gives and error since it is the only value that controls the operation of stack.