**AIM:-Design and implement a stack using an array and demonstrate its working with necessary input, display the appropriate output**

**algorithm**

Step 1: Include all the header files which are used the program and define a constant “size” with a specific value.

Step 2: Declare all the functions used in stack implementation.

Step 3: Create a one-dimensional array with fixed size (int stack size).

Step 4: Define a integer variable “top” and initialize with -1 (int top=-1).

Step 5: In main method, display, menu with list of operations and make a suitable function calls to perform opereation selected by the user on the stack.

TO PUSH:

Step 1: Check whether stack is full (top==size-1).

Step 2: If it is full, then display “stack is full insertion is not possible” and terminate the function.

Step 3: If it is not full, then increment top value by one (top++) and set stack [top] to value (stack [top]=value).

TO POP:

Step 1: Check whether the stack is empty(top==-1).

Step 2: If it is empty, then display “stack is empty” and terminate the function.

Step 3: If it is not empty, then delete stack[top] and decrement top value by one [top--].

TO DISPLAY:

Step 1: Check whether the stack is empty (top==-1).

Step 2: If empty display “stack is empty” and terminate.

Step 3: If not empty, then define a variable ‘I’ and initialize with top. Display stack [1] value and decrement I value by one (i—0).

Step 4: Repeat above stop until I value becomes ‘0’.

program: -

#include<stdio.h>//standard input output header file//

#include<stdlib.h>//standard library header file//

#define size 5//initializing the array size to 5//

void push();//push function//

void pop();//pop function//

void display();//display function//

int top=-1,inp\_arr[size],i;//initializing the top value to the -1//

int main()//main function//

{

int choice;//declaring the choice//

while(1)//checks the condition//

{

printf("operation performed by the stack");//prints the statement//

printf("\n 1.push \n 2.pop \n 3.display \n 4.exit \n");//prints the functions//

printf("enter the operation performed by the stack\n");//print the statement//

scanf("%d",&choice);//scans the input and places the value into the choice//

switch(choice)//scans the condition and enters into the case when it satisfies//

{

case 1:push();//prints the case 1 when the condition is 1//

break;

case 2:pop();//prints the case 2 when the condition is 2//

break;

case 3:display();//prints the case 3 when the condition is 3//

break;

case 4:exit(0); //come out of the loop//

default:printf("invalid entry\n");//prints the value//

}

}

return 0;

}

void push()

{

int x;//declaring the x//

if(top==size-1)//checks the condition//

{

printf("overflow\n");//prints the value when it satisfies//

}

else

{

printf("enter the value to be inserted\n");

scanf("%d",&x);

top=top+1;//increment by 1//

inp\_arr[top]=x;

}

}

void pop()

{

if(top==-1)//checks the condition//

{

printf("underflow\n");

}

else

{

printf("popped element is %d\n",inp\_arr[top]);

top=top-1;//decrement//

}

}

void display()

{ int i;

if(top>-1)//checks the condition//

{

printf("elementes in the stack r \n");

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

{

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

}

}

else

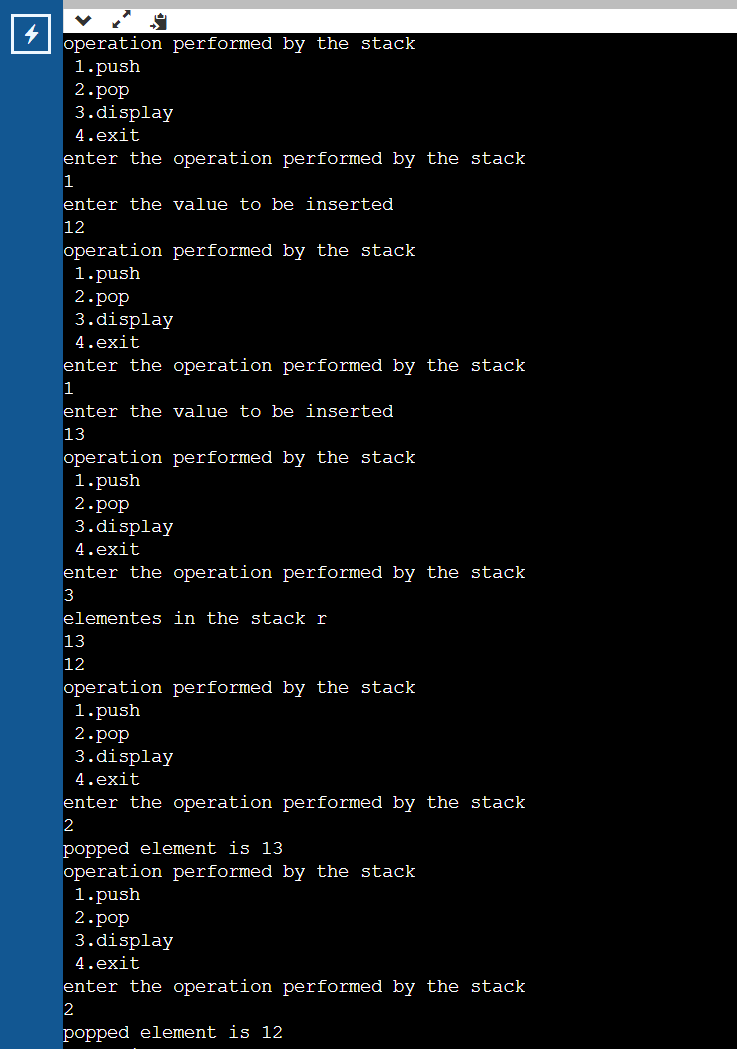
{

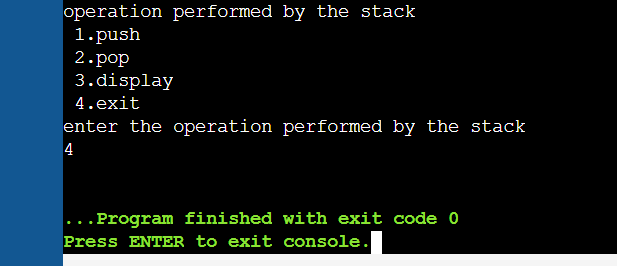
printf("stack is empty\n");

}

}

Screenshot of output: -





Link of the program in git hub: -

https://github.com/Koushik123123/datastructures-1/blob/main/19\_12\_22\_1NT21IS109\_program%201.docx