# **Name: Abdurrahman Qureshi**

# **Roll no: 210451**

**Aim: Stack via array**

**CODE:**

#include <stdio.h>

#define size 10

int top = -1;

int stack\_array[size];

void popStackArray()

{

if (top == -1)

{

printf("\n\t Stack underflow!!!\n");

}

else

{

printf("\n\t Element %d - Popped from stack\n", stack\_array[top]);

top = top - 1;

}

}

void pushStackArray()

{

int stackData;

if (top == size)

{

printf("\n\t Stack overflow!!!\n");

}

else

{

printf("\n\t Enter data to be pushed:");

scanf("%d", &stackData);

top = top + 1;

stack\_array[top] = stackData;

printf("\n\t Element %d - Pushed into stack\n", stack\_array[top]);

}

}

void displayStackArray()

{

if (top == -1)

{

printf("\n\t Stack underflow!!!\n");

}

else

{

printf("\n\t");

printf("NULL");

for (int i = top; i >= 0; i--)

{

printf("<----%d", stack\_array[i]);

}

printf("\n");

}

}

void stackArray()

{

int choice;

printf("\n\tFollowing operations on stack can occur according to their respective number:\n");

while (1)

{

printf("\n\t 1 - Popping from the stack");

printf("\n\t 2 - Pushing in to the stack");

printf("\n\t 3 - Display the created stack");

printf("\n\t 4 - Exit");

printf("\n\n\t Enter your choice:");

scanf("%d", &choice);

switch (choice)

{

case 1:

popStackArray();

break;

case 2:

pushStackArray();

break;

case 3:

displayStackArray();

break;

case 4:

exit(0);

default:

printf("\n\t Invalid Choice!!!\n");

}

}

}

int main()

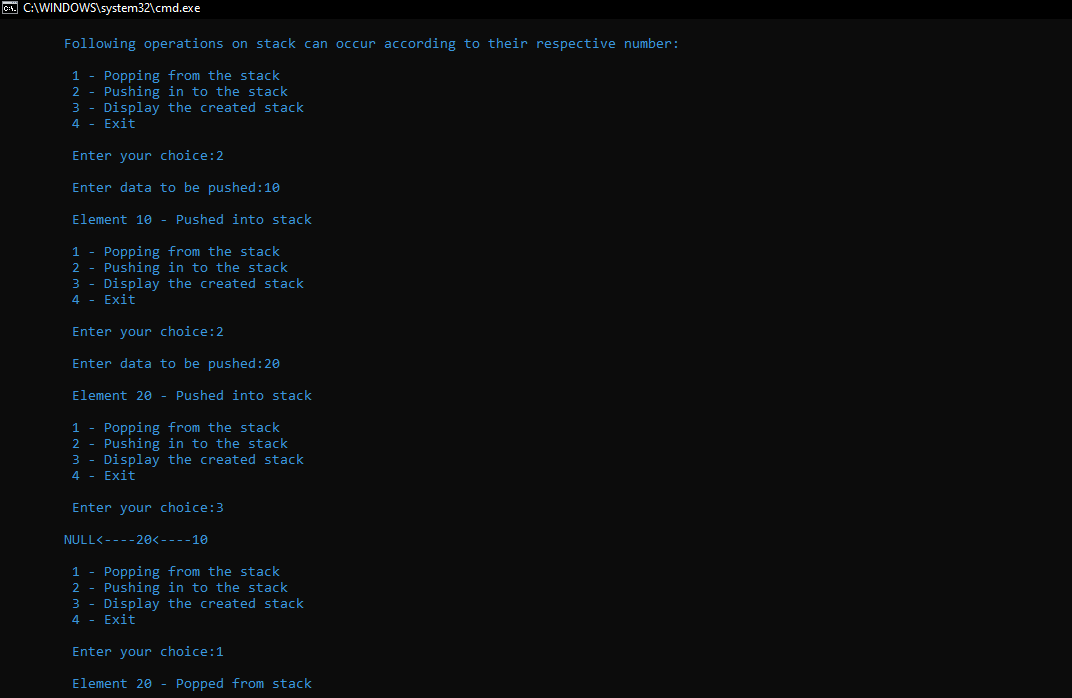
{

stackArray();

return 0;

}

**OUTPUT:**

****

**Aim: Stack via link list**

**CODE:**

#include<stdio.h>

#include<stdlib.h>

struct node{

int data;

struct node \*next;

};

struct node\* top = NULL;

int popStackLinkList(){

struct node \*poppinNode = top;

if(top == NULL){

printf("\n\n\t Stack underflow!!!\n");

}

else{

int temp\_data = top->data;

top = top->next;

printf("\n\n\t Number %d popped into the stack",poppinNode);

free(poppinNode);

return temp\_data;

}

}

void pushStackLinkList(){

int num;

struct node \*pushNewNode;

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

printf("\n\n\t Enter a number to push into the stack:");

scanf("%d",&num);

if(top == NULL){

pushNewNode->next = NULL;

}

else{

pushNewNode->data = num;

pushNewNode->next = top;

}

top = pushNewNode;

printf("\n\n\t Number %d pushed into the stack",num);

}

void displayStackLinkList(){

struct node \*temp = top;

if (top == NULL) {

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

return;

}

else{

printf("\n\t The stack is as follows:\n");

printf("\t");

printf("\n\t NULL ");

while(temp != NULL){

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

temp=temp->next;

}

//printf("%d---->NULL",temp->data);

}

}

void stackLinkList(){

int choice;

printf("\n\tFollowing operations on stack can occur according to their respective number:\n");

while(1){

printf("\n\t 1 - Popping from the stack");

printf("\n\t 2 - Pushing in to the stack");

printf("\n\t 3 - Display the created stack");

printf("\n\t 4 - Exit \n");

printf("\n\t Enter your choice:");

scanf("%d",&choice);

switch(choice){

case 1:

popStackLinkList();

break;

case 2:

pushStackLinkList();

break;

case 3:

displayStackLinkList();

break;

case 4:

exit(0);

default :

printf("\n\t Invalid Choice!!!\n");

}

}

}

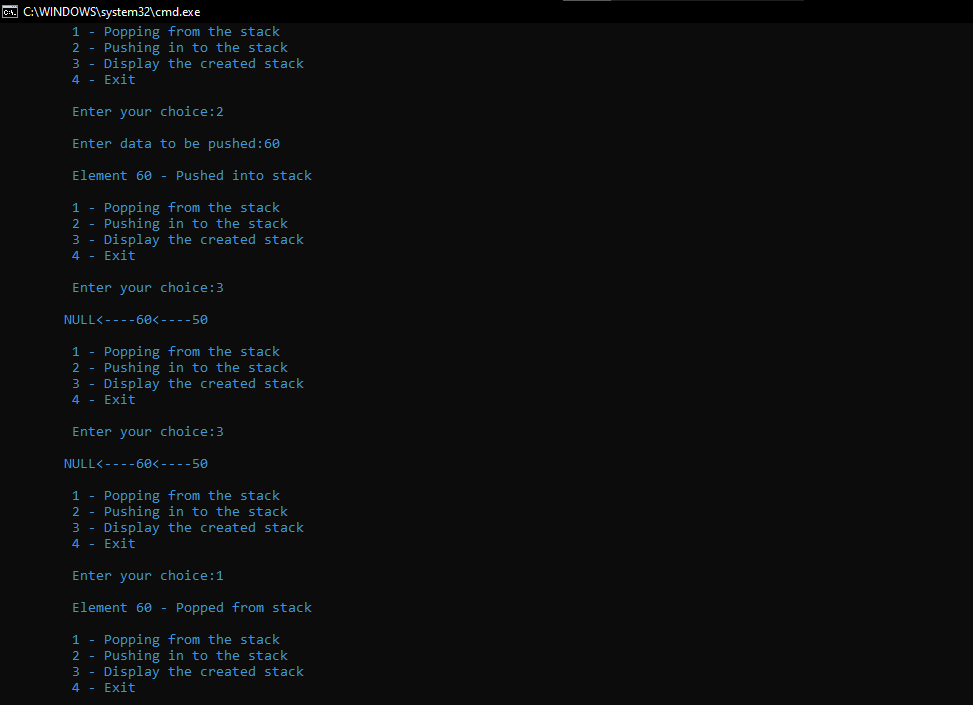
int main(){

stackLinkList();

return 0;

}

**OUTPUT:**

****