#include <stdio.h>

#include <stdlib.h>

struct Stiva {

int elemente[10];

int top;

};

int full\_stack(struct Stiva\* stiva, int max) {

if (stiva->top == max) {

return 1;

}

return 0;

}

int empty\_stack(struct Stiva\* stiva) {

if (stiva->top == 0) {

return 1;

}

return 0;

}

void push(struct Stiva\* stiva, int max) {

if (stiva->top != max) {

printf("Introduceti numarul dorit\n");

int val;

scanf("%d", &val);

stiva->elemente[stiva->top] = val;

stiva->top++;

} else {

printf("Stack overflow\n");

}

}

void pop(struct Stiva\* stiva) {

if (stiva->top != 0) {

stiva->top--;

} else {

printf("Stack underflow\n");

}

}

void initializare(struct Stiva\* stiva) {

stiva->top = 0;

}

int main() {

struct Stiva stiva;

int max = 10;

initializare(&stiva);

if (full\_stack(&stiva, max) == 1) {

printf("Stiva este plina\n");

}

if (empty\_stack(&stiva) == 1) {

printf("Stiva este goala\n");

}

while (full\_stack(&stiva, max) != 1) {

push(&stiva, max);

}

pop(&stiva);

for (int i = 0; i < stiva.top; i++) {

printf("%d ", stiva.elemente[i]);

}

return 0;

}

/\*

struct Nod{

struct Nod\* succ;

int info;

};

struct stiva{

struct Nod\* inceput;

struct Nod\* top;

};

struct stiva\* initializare(){

struct stiva\* lista=(struct stiva\*)malloc(sizeof(struct stiva));

lista->inceput=NULL;

lista->top=NULL;

return lista;

}

struct Nod\* init(int val){

struct Nod\* nod\_nou=(struct Nod\*)malloc(sizeof(struct Nod));

nod\_nou->info=val;

nod\_nou->succ=NULL;

return nod\_nou;

};

struct stiva\* inserare(struct stiva\* lista, int val){

if(lista->inceput==NULL){

lista->top=init(val);

lista->inceput=lista->top;

}else{

struct Nod\* nod\_nou=init(val);

struct Nod\* tmp=lista->inceput;

while(tmp->succ!=NULL){

tmp=tmp->succ;

}

tmp->succ=nod\_nou;

nod\_nou->succ=NULL;

lista->top=nod\_nou;

}

return lista;

}

void testare(struct stiva\* lista){

if(lista->inceput==NULL){

printf("Stiva este goala\n");

}

else{

printf("Stiva nu este goala\n");

}

}

struct stiva\* pop(struct stiva\* lista){

if(lista->top==NULL){

printf("Stiva nu mai are nici-un element\n");

}

else if(lista->inceput->succ==NULL){

struct Nod\* tmp=lista->inceput;

lista->inceput=NULL;

free(tmp);

}

else{

struct Nod\* tmp=lista->inceput;

while(tmp->succ->succ!=NULL){

tmp=tmp->succ;

}

struct Nod\* temp=tmp->succ;

tmp->succ=NULL;

free(temp);

lista->top=tmp;

}

return lista;

}

void reset(struct stiva\* lista){

if(lista->inceput==NULL){

printf("Stiva este goala\n");

}

else{

while(lista!=NULL){

lista= pop(lista);

}

lista->top=NULL;

lista=NULL;

}

}

void adunare(struct stiva\* S1, struct stiva\* S2) {

struct Nod\* tmp1=S1->inceput;

struct Nod\* tmp2=S2->inceput;

int N1[10];

int N2[10];

int i=0;

while(tmp1!=NULL){

N1[i]=tmp1->info;

i++;

tmp1=tmp1->succ;

}

int j=0;

while(tmp2!=NULL){

N2[j]=tmp2->info;

j++;

tmp2=tmp2->succ;

}

int nr1=0;

for(int k=0; k<i; k++){

nr1=nr1+pow(10, k)\*N1[i-k-1];

}

int nr2=0;

for(int k=0; k<j; k++){

nr2=nr2+pow(10, k)\*N2[j-k-1];

}

printf("%d\n", nr1);

printf("%d\n", nr2);

printf("%d", nr1+nr2);

}

int main(){

struct stiva\* lista=initializare();

lista=inserare(lista, 3);

lista=pop(lista);

printf("%d\n", lista->inceput->info);

testare(lista);

struct stiva\* S1=initializare();

struct stiva\* S2=initializare();

S1=inserare(S1, 5);

S1=inserare(S1, 2);

S2=inserare(S2, 5);

S2=inserare(S2, 5);

adunare(S1, S2);

}\*/