// IMPLEMENTATION OF STACK

#include <stdio.h>

#include <stdlib.h>

#define MAX 50 //Max size of stack

int stack[MAX]; //Defining stack

int top; //Defining top

//Function declaration

void initialize();

int isEmpty();

int isFull();

int size();

void push(int );

void pop(int\*);

void peek();

void display();

int main() {

int num, elem;

int popped, peeked;

initialize();

//Enter choices

while(1) {

printf("\n");

printf("\* 1 -> PUSH");

printf("\n\* 2 -> POP");

printf("\n\* 3 -> PEEK");

printf("\n\* 4 -> DISPLAY");

printf("\n\* 5 -> SIZE");

printf("\n\* 6 -> EXIT\n");

scanf("%d", &num);

printf("\n\t");

if(num == 1) { //PUSH

printf("\n--Enter a number to push-- ");

scanf("%d", &elem);

push(elem);

} else if(num == 2) { //POP

pop(&popped);

} else if(num == 3) { //PEEK

peek(&peeked);

} else if(num == 4) { //DISPLAY

display();

} else if (num == 5) { //SIZE

printf("\n--Currently, size of stack is => %d", size());

} else if (num == 6) { //QUIT

printf("\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

break;

} else { //DEFAULT

printf("\nINVALID INPUT");

}

}

return 0;

}

void initialize() {

top = -1;

}

int isEmpty() {

if (top == -1)

return 1;

return 0;

}

int size() {

return top+1;

}

int isFull() {

if (top == MAX-1)

return 1;

return 0;

}

void push(int elem) {

if(isFull()) {

printf("\nOVERFLOW");

return;

}

top++;

stack[top] = elem;

}

void pop(int\* popped) {

if(isEmpty()) {

printf("\nUNDERFLOW");

return;

}

\*popped = stack[top];

top--;

printf("--Popped element is => %d", \*popped);

return;

}

void peek(int\* peeked) {

if(isEmpty()) {

printf("\nUNDERFLOW");

return;

}

\*peeked = stack[top];

printf("--Top value is => %d", \*peeked);

return;

}

void display() {

int i;

if(isEmpty()) {

printf("\nEMPTY");

} else {

printf("\nElements in Stack are : ");

for (i=0 ; i<=top ; i++) {

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

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//OUTPUT



