**C program to implement Stack using array**

#include <stdio.h>

#include <stdlib.h>

#define MAX 10

int STACK[MAX],TOP;

/\* display stack element\*/

void display(int []);

/\* push (insert) item into stack\*/

void PUSH(int [],int);

/\* pop (remove) item from stack\*/

void POP (int []);

void main()

{

int ITEM=0;

int choice=0;

TOP=-1;

while(1)

{

/\*clrscr();\*/

printf("Enter Choice (1: display, 2: insert (PUSH), 3: remove(POP)), 4: Exit..:");

scanf("%d",&choice);

switch(choice)

{

case 1:

display(STACK);

break;

case 2:

printf("Enter Item to be insert :");

scanf("%d",&ITEM);

PUSH(STACK,ITEM);

break;

case 3:

POP(STACK);

break;

case 4:

exit(0);

            default:

printf("\nInvalid choice.");

break;

}

getch();

}// end of while(1)

}

/\* function : display(),

    to display stack elements.

\*/

void display(int stack[])

{

int i=0;

if(TOP==-1)

{

printf("Stack is Empty .\n");

return;

}

printf("%d <-- TOP ",stack[TOP]);

for(i=TOP-1;i >=0;i--)

{

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

}

printf("\n\n");

}

/\* function : PUSH(),

    to push an item into stack.

\*/

void PUSH(int stack[],int item)

{

if(TOP==MAX-1)

{

printf("\nSTACK is FULL CAN't ADD ITEM\n");

return;

}

TOP++;

stack[TOP]=item;

}

/\* function : POP(),

    to pop an item from stack.

\*/

void POP(int stack[])

{

int deletedItem;

if(TOP==-1)

{

printf("STACK is EMPTY.\n");

return;

}

deletedItem=stack[TOP];

TOP--;

printf("%d deleted successfully\n",deletedItem);

return;

}

**program to implement stack using array/ linear implementation of stack USING STACK STRUCTURE**

﻿#include< stdio.h >

#define MAX 100

/\* declaration\*/

typedef struct Stack

{

int ele[MAX];

int TOP;

}STACK;

STACK \*s;

/\* function: pushItem( ), to insert an item into stack. \*/

void pushItem(STACK \*s,int ITEM)

{

if(s->TOP==MAX-1)

{ printf("\nSTACK is FULL.\n"); return; }

s->ele[++s->TOP]=ITEM;

printf("\nITEM inserted successfully.\n");

}

/\* function: popItem( ), to delete an item from stack. \*/

int popItem(STACK \*s)

{

int itm;

if(s->TOP==-1) { printf("\nSTACK is empty.\n"); return; }

itm=s->ele[s->TOP];

s->TOP--;

printf("\nItem removed : %d\n",itm);

}

/\* function: dispItem( ), to display stack elements. \*/

void dispItems(STACK \*s)

{

int i;

if(s->TOP == -1)

{ printf("STACK IS EMPTY."); return; }

for(i=s->TOP;i>=0;i--)

printf("%d->",s->ele[i]);

printf("\n");

}

/\*\*\* main function \*\*/

void main()

{

int num; char dummy;

/\*\* initialisation\*\*/

s=(STACK\*)malloc(sizeof(STACK));

s->TOP=-1;

int choice=0;

again:

/\* display stack elements \*\*/

printf("\nSTACK ELEMENTS :"); dispItems(s);

printf("\nSTACK OPTIONS \n0: Exit\n1: Add item\n2: Remove item \nEnter choice :::");

scanf("%d",&choice);

switch(choice)

{ case 0:

exit(1);

break;

case 1:

printf("\nEnter an item to insert:");

scanf("%d",&num);

pushItem(s,num);

break;

case 2:

popItem(s);

break;

        default:

printf("\nAn Invalid Choice !!!");

break;

}

scanf("%c",&dummy);

goto again;

}