PROGRAM TO CREATE A STACK OF INTEGERS AND IMPLEMENT PUSH, POP AND DISPLAY OPERATIONS

```
#include<stdio.h>
#include<stdlib.h>
#define SIZE 5
int s[SIZE],top=-1; //global variables stack and top
void push(int);  // function prototype for insertion
int pop();  // function prototype for deletion
void display(); // function prototype for displaying the contents of stack
void main()
{
 int ch,item,del;
 for(;;) // infinte loop to ask user choices for different operations
 {
  printf("\n1.Push 2.Pop 3.Display 4.Exit\n");
  printf("Enter your choice:");
  scanf("%d",&ch);
  switch(ch)
  {
    case 1:printf("Enter the item to be inserted\n");
           scanf("%d",&item);  // read element to be inserted to the stack
           push(item);  // call push function to insert element
           break;

    case 2:del=pop(); // call pop function to delete top most element of the stack
           if(del!=-1)
                   printf("Deleted element is %d\n",del); // print deleted element
           break;

    case 3:display();  // call display function to see the contents of stack
           break;

    case 4:exit(0);
  }
 }
}

void push(int ele)
{
 if(top==SIZE-1)  //check if stack is full
 {
   printf("Stack overflow\n");
   return;
 }
 s[++top] = ele;  // increment top and insert the element (pre increment)
}

int pop()
{
 if(top==-1) // check if stack is empty
 {
   printf("Stack underflow\n");
   return -1;
 }
 return s[top--]; //delete the element and decrement top (post decrement)
}

void display()
{
 int i;
 if(top==-1) // check if stack is empty
 {
   printf("Stack is empty\n");
   return;
 }
 printf("Elements in stack are as follows:\n");
 for(i=0;i<=top;i++)  // print all elements of the stack from index 0 to top
   printf("%d ",s[i]);
}
```

NOTE:

- **Try modifying the same program for stacks of type float and char**
- **Use local variables for ‘*stack array*’ and ‘*top*’. Pass ‘*stack array*’ and ‘*top*’ as arguments to all functions. As ‘*top*’ value gets updated in push and pop function, it has to be passed by reference**