```
//StackIf.h
#include<stdio.h>
typedef struct{
        int top,size;
        char a[14];
        }stack;
void initialize(stack *s);
void push(stack *s,char x);
char pop(stack *s);
int isFull(stack *s);
int isEmpty(stack *s);
void display(stack *s);

//StackImpl.h
#include<stdio.h>
#define max 40
void initialize(stack *s){
        s->top=-1;
        s->size=0;
        //s->exp=NULL;
}

int isFull(stack *s){
        if(s->top==max-1)
                return 1;
        else
                return 0;
}
```

```
int isEmpty(stack *s){
        if(s->top==-1)
                return 1;
        else
                return 0;
}

void push(stack *s,char x){
    if(isFull(s))
    {
        printf("\n\tStack is over flow");
    }
    else
    {
        s->top++;
        s->a[s->top]=x;
       s->size++;
    }
}

char pop(stack *s){
    if(isEmpty(s))
    {
        printf("\n\tStack empty");
    }
    else
    {
       s->size--;
       return s->a[s->top--];
```

```
	}
}

void display(stack *s){
	int i;
	if(isEmpty(s))
		printf("\nThe Stack is empty");

	else{
		printf("\nThe elements in Stack \n");
		for(i=0; i<s->size; i++)
			printf("%c\n",s->a[i]);
	}
}

//StackV1.c
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include "StackIf.h"
#include "StackImpl.h"

void main(){
		int i;
		char exp[],ch;
		stack *s;
		s=(stack *)malloc(sizeof(stack));

		initialize(s);
```

```
	printf("Enter expression: ");
	scanf("%[^\n]",exp);

	for(i=0;i<strlen(exp);i++)
			push(s,exp[i]);
	display(s);

	for(i=0;i<strlen(exp);i++){
		ch=pop(s);
		printf("Popping %c\n",ch);}
}

//StackV2.c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

#include "StackIf.h"
#include "StackImpl.h"

void toPostfix(stack *s,char in[]);
void evaluate(stack *s,char po[]);

void main(){
	char infix[40],postfix[40];
	stack*s=(stack *)malloc(sizeof(stack));
	initialize(s);
```

```
	printf("Enter infix expression: ");
	scanf("%s", infix);
	toPostfix(s,infix);
}

void toPostfix(stack *s,char in[]){
	int i,j=0;
	char po[40],ch;

	for(i=0;i<strlen(in);i++){
		if(isdigit(in[i])){
			po[j]=in[i];
			j++;
		}

		else{
			if(in[i]=='+'||in[i]=='-'){

	if(s->a[s->top]=='+'||s->a[s->top]=='-'||s->a[s->top]=='*'||s->a[s->top]=='/'){
						po[j]=pop(s);
						j++;

						if(s->a[s->top]=='+'||s->a[s->top]=='-'){
				po[j]=pop(s);
				j++;
			}

			push(s,in[i]);
```

```
                }
                else
                        push(s,in[i]);
            }

            else if(in[i]=='*'||in[i]=='/'){
                if(s->a[s->top]=='*'||s->a[s->top]=='/'){
              po[j]=pop(s);
                        j++;
                        push(s,in[i]);
                }
                else
                        push(s,in[i]);
            }
        }//end if-else
    } //end for

    if(!isEmpty(s)){
        do{
            ch=pop(s);
            po[j]=ch;
            j++;
          }while(!isEmpty(s));
    }
po[j]='\0';

    printf("Postfix form:");
    for(i=0;i<strlen(po);i++)
        printf("%c",po[i]);
```

```
	stack *p=(stack *)malloc(sizeof(stack));
	initialize(p);
	evaluate(p,po);

}//end toPostfix

void evaluate(stack *s,char po[]){
	int res,a,b;
	int i,j=0;
	//printf("\nstrlen of postfix %d",strlen(po));
	for(i=0;i<strlen(po);i++){
		if(isdigit(po[i])){

				//printf("\nhi\n");
			push(s,(int)po[i]-48);

			//printf("%d",s->top);
		}
		else if(po[i]=='+'||po[i]=='-'||po[i]=='*'||po[i]=='/')
		{
				b=pop(s);
				a=pop(s);

				if(po[i]=='+')
					res=a+b;
				if(po[i]=='-')
					res=a-b;
				if(po[i]=='*')
```

```
                    res=a*b;
                if(po[i]=='/')
                    res=a/b;
                //printf("%d",res);
                push(s,res);
            }
        }

        printf("\nResult: ");
        printf("%d",s->a[s->top]);
}

/*
OUTPUT:
Enter infix expression: 3+5/5*2-2
Postfix form:355/2*+2-
Result: 3
*/
```