```
//============================================================================
// Name        : fds10.cpp
// Author      : Aastha Bisen
// Roll No.    : COSA23
// Description : Assignment No.10
//============================================================================

#include<iostream>
using namespace std;

#define MAX 20
class stack
{
int top,topeval;
char infix[MAX], postfix[MAX],stk[MAX];
float stkeval[MAX];
public:
stack();
void push(char);
void pusheval(float);
void read();
int IsEmpty ();
int IsEmptyeval ();
char pop();
float popeval();
void covert_infix_postfix();
int priority(char x);
float evaluatepostfix();
};

stack :: stack ()
{
top=-1;
topeval=-1;
}

int stack :: IsEmpty ()
{
if (top == -1)
return (-1);
else
return (1);
}
int stack :: IsEmptyeval ()
```

```
{
if (topeval == -1)
return (-1);
else
return (1);
}

void stack :: push(char temp)
{
top ++;
if (top == MAX)
cout<<"Stack is Full";
else
{
stk [top] = temp;
}

}
void stack :: pusheval(float temp)
{
topeval ++;
if (topeval == MAX)
cout<<"Stack is Full";
else
{
stkeval [topeval] = temp;
}

}

char stack :: pop()
{
int status;
char temp;
status = IsEmpty ();
if (status == -1) {
return ('@');
}
else {
temp = stk [top--];
return (temp);
}
}
```

```
float stack :: popeval()
{
int status;
float temp;
status = IsEmptyeval ();
if (status == -1) {
return (-999);
}
else {
temp = stkeval [topeval--];
return (temp);
}
}
void stack :: read()
{
cout<<"Enter infix expression ";
cin>>infix;
}


void stack :: covert_infix_postfix()
{
int i,k=0;
char sop;
for(i=0;infix[i]!='\0';i++)
{
if(infix[i]>='0' && infix[i]<='9')
postfix[k++] = infix[i];
else if(infix[i]=='(')
push(infix[i]);
else if(infix[i]==')')
{
while((sop=pop())!='(')
postfix[k++]=sop;
}
else
{
while(priority(infix[i]) <= priority(sop=pop()))
{
postfix[k++] = sop;
if(top==-1)
break;
}
if(priority(infix[i]) > priority(sop))
```

```
{
push(sop);
}
push(infix[i]);
}
}
while(top!=-1)
{
sop = pop();
postfix[k++] = sop;
}
postfix[k-1]='\0';
cout<<endl<<"The postfix is : "<<postfix;
}

int stack :: priority(char x)
{
switch(x)
{
case '@':
return -1;
case '(':
return 0;
case '+':
return 1;
case '-':
return 1;
case '*':
return 2;
case '/':
return 2;
}
}

float stack :: evaluatepostfix()
{
float value,operand1,operand2,result;
for(int i=0;postfix[i]!='\0';i++)
{
if(postfix[i]>='0' && postfix[i]<='9')
{
value = postfix[i] - 48;
pusheval(value);
}
```

```
else
{
operand2=popeval();
operand1=popeval();
switch(postfix[i])
{
case '+':
result=operand1+operand2;
break;
case '*':
result=operand1*operand2;
break;
case '-':
result=operand1-operand2;
break;
case '/':
result=operand1/operand2;
break;
}
pusheval(result);
}
}
result=popeval();
return result;
}
int main()
{
stack s;
float answer;
s.read();
s.covert_infix_postfix();
answer=s.evaluatepostfix();
cout<<endl<<"The answer is : "<<answer;
return 0;
}
```