1. Pila.h

```
#ifndef PILA_H
#define PILA_H

template<typename T>
class Pila
{
    public:
        Pila();
        class Nodo{
            public:
                Nodo();
                Nodo(T dato);
                T dato;
                Nodo *siguiente;
            private:
        };
        void agregar(T dato);
        bool esVacia();
        T desapilar();
    protected:
    private:
        Nodo * cabeza;
};

template<typename T>
Pila<T>::Nodo::Nodo(){
    siguiente = nullptr;
}
template<typename T>
Pila<T>::Nodo::Nodo(T dato){
    this->dato = dato;
}
template<typename T>
Pila<T>::Pila(){
    cabeza = nullptr;
}
template<typename T>
void Pila<T>::agregar(T dato){
    Pila<T>::Nodo * nuevo = new Nodo(dato);
    if(cabeza == nullptr){
        cabeza = nuevo;
    }
    else{
        nuevo->siguiente = cabeza;
        cabeza = nuevo;
    }
}
template<typename T>
bool Pila<T>::esVacia(){
    if(cabeza == nullptr)
        return true;
    return false;
}
template<typename T>
T Pila<T>::desapilar(){
    if(esVacia()){
        return 0;
```

```
58      }
59      else{
60          T result = cabeza->dato;
61          cabeza = cabeza->siguiente;
62          return result;
63      }
64  }
65
66
67
68  #endif // PILA_H
```

2. main.cpp

```
1
2   #include <iostream>
3   #include <Pila.h>
4   #include <math.h>
5
6   using namespace std;
7
8   float convertirNumero(string numero){
9       double resultado = 0;
10      auto iter = numero.end();
11      iter--;
12      double contador = 0;
13      for(iter; iter!= numero.begin(); iter--){
14          if((*iter) == '.'){
15              resultado /= pow(10,contador);
16              contador = -1;
17          }
18          else{
19              resultado += pow(10,contador) * ((*iter) - 48);
20          }
21          contador++;
22      }
23      resultado += pow(10,contador) * ((*iter) - 48);
24      return resultado;
25  }
26  int main()
27  {
28      Pila<float> memoria;
29      string expresion;
30      string temp;
31      cin>>expresion;
32      for(int i = 0; i < expresion.size(); i++){
33          if(expresion[i] == '|'){
34              memoria.agregar(convertirNumero(temp));
35              temp.clear();
36          }
37          else if((expresion[i] >= 48 and expresion[i] <= 57)
38                  or expresion[i] == '.'){
39              temp.insert(temp.end(),expresion[i]);
40          }
41          else{
```

```
42              switch(expresion[i]){
43                  case '+':
44                      memoria.agregar(memoria.desapilar()
45                                      + convertirNumero(temp));
46                      temp.clear();
47                      break;
48                  case '-':
49                      memoria.agregar(memoria.desapilar()
50                                      - convertirNumero(temp));
51                      temp.clear();
52                      break;
53                  case '*':
54                      memoria.agregar(memoria.desapilar()
55                                      * convertirNumero(temp));
56                      temp.clear();
57                      break;
58                  case '/':
59                      memoria.agregar(memoria.desapilar()
60                                      / convertirNumero(temp));
61                      temp.clear();
62                      break;
63                  default:
64                      cout<<"El caracter "<<expresion[i]
65                          <<" no se puede reconocer";
66                      return 0;
67              }
68          }
69      }
70      cout<<memoria.desapilar();
71  }
```