山东大学 计算机科学与技术 学院

数据结构与算法 课程实验报告

学号：202000130143	姓名： 郑凯饶	班级： 计科 20.1
实验题目：栈		
实验学时：2	实验日期：1103	

实验目的：

1. 掌握栈结构的定义与实现；
2. 掌握栈结构的使用。

软件开发环境：

Windows 10 家庭中文版 64 位（10.0，版本 18363）

Dev-C++ IDE

1. 实验内容

创建栈类，采用数组描述，计算数学表达式的值。数学表达式由单个数字和运算符“+”，“-”，“*”，“/”，“(”，“)”构成。

2. 数据结构与算法描述 （整体思路描述，所需要的数据结构与算法）

```
template<class T>
class arrayStack { // : public stack<T>
    public:
        arrayStack(int iniCap = 1000);
        ~arrayStack() { delete [] stack; }
        bool empty() const { return Top == -1; }
        int size() const { return Top + 1; }
        T& top() {
            if (Top == -1) {

            }
            return stack[Top];
        }
        void pop() {
            stack[Top--].~T();      //  3221226356
//          Top--;
        }
        void push(const T& tar);
        void clear() {
            while (!(this -> empty())) {
                this -> pop();
            }
        }
    private:
        int Top;
        int Length;
        T* stack;
};
```

以上为栈的定义与实现。

Top：栈顶元素的索引

Length：栈的容量

Stack：栈元素的储存空间

表达式计算的思路：

维护两个栈 Dig 和 Sym，前者储存数字，后者储存运算符。扫描一遍表达式，遇到数字直接入栈（Dig）；遇到运算符（加减乘除）将其优先级和栈顶（Sym）的元素进行比较，若 Top 的优先级大于等于当前运算符，进行一次 Cal()，直到栈空或者 Top 优先级小于当前运算符。若遇到“(”，直接入栈；遇到“)”，不断 Cal()直到遇到“(”。

Cal()函数就是进行一次运算，将 Dig 栈顶的两个元素弹出，将 Sym 栈顶的一个元素弹出，进行运算，结果入栈（Dig）。

3. 测试结果（测试输入，测试输出）
在 OJ 平台上成功提交。

4. 分析与探讨（结果分析，若存在问题，探讨解决问题的途径）
一开始设计算法的时候，忽略了栈顶和当前运算符是同级别时也要进行 Cal()，像 3/2*5，计算顺序应该是 3/2=1.5 -> 1.5*5=7.5，而不是 2*5=10 -> 3/10=0.3.
另一方面，在实现算法时，写错了一些程序逻辑和细节。如将 pop 误输入为 top，导致最后未将栈清空，而题目有多组样例输入，引发越界错误，程序返回异常值 3221225477。

5. 附录：实现源代码（本实验的全部源程序代码，程序风格清晰易理解，有充分的注释）

```
#include<bits/stdc++.h>
using namespace std;

template<class T>
class arrayStack { // : public stack<T>
 public:
  arrayStack(int iniCap = 1000);
  ~arrayStack() { delete [] stack; }
  bool empty() const { return Top == -1; }
  int size() const { return Top + 1; }
  T& top() {
   if (Top == -1) {

   }
   return stack[Top];
  }
  void pop() {
   stack[Top--].~T();  //  3221226356
//   Top--;
  }
  void push(const T& tar);
  void clear() {
   while (!(this -> empty())) {
    this -> pop();
   }
  }
 private:
  int Top;
  int Length;
  T* stack;
};

template<class T>
arrayStack<T>::arrayStack(int iniCap) {
```

```
 Length = iniCap;
 stack = new T[Length];
 Top = -1;
}

template<class T>
void arrayStack<T>::push(const T& tar) {
 // changeLength1D()
 stack[++Top] = tar;
}

arrayStack<double> Dig;
arrayStack<int> Sym;

string operand[] = {"+", "-", "*", "/"};

void Cal() {
 double b = Dig.top(); Dig.pop();
 double a = Dig.top(); Dig.pop();
 int op = Sym.top(); Sym.pop();

 double Res;
 switch(op) {
  case 0:
   Res = a + b;
   break;
  case 1:
   Res = a - b;
   break;
  case 2:
   Res = a * b;
   break;
  case 3:
   Res = a / b;
   break;
 }
// cout << a << " " << operand[op] << " " << b << " = " << Res << '\n';
 Dig.push(Res);
}

void solve() {

 int n; cin >> n;
 string Exp;
 for (int i = 1; i <= n; i++) {
//  Dig.clear(); Sym.clear();  // 3221225477
```

```
  cin >> Exp;
  int len = Exp.length();
  // Cal
  for (int j = 0; j < len; j++) {
   if (Exp[j] <= '9' && Exp[j] >= '0') Dig.push(Exp[j] - '0');
   else {
    if (Exp[j] == '+') {
     while (!Sym.empty() && (Sym.top() <= 3)) {
      Cal();
     }
     Sym.push(0);
    }
    else if (Exp[j] == '-') {
     while (!Sym.empty() && (Sym.top() <= 3)) {
          Cal();
        }
        Sym.push(1);
       }
       else if (Exp[j] == '*') {
        while (!Sym.empty() && (Sym.top() == 2 || Sym.top() == 3)) {
         Cal();
        }
        Sym.push(2);
       }
       else if (Exp[j] == '/' ) {
        while (!Sym.empty() && (Sym.top() == 2 || Sym.top() == 3)) {
         Cal();
        }
        Sym.push(3);
       }
       else if (Exp[j] == '(') {
        Sym.push(4);
       }
       else if (Exp[j] == ')') { // 5
        while (Sym.top() != 4) {
         Cal();
        }
        Sym.pop();
       }
      }
     }
     // More
     while (Dig.size() > 1) {
   //   cout << "Cal" << '\n';
      Cal();

```

```
132.      double Ans = Dig.top(); Dig.pop();
133.      cout << Ans << '\n';
134.     }
135.    }
136.    int main(){
137.     cout << setiosflags(ios::fixed) << setprecision(2);
138.     solve();
139.     return 0;
140.    }
```