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：202200400053 | 姓名： 王宇涵 | | 班级： 22级2班 |
| 实验题目：栈 | | | |
| 实验学时：2 | | 实验日期： 2023-10-25 | |
| 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：  Vscode | | | |
| 1. 实验内容   题目描述：  创建栈类（采用数组描述）计算数学表达式的值。对于输入的数学表达式，  输出表达式的计算结果。数学表达式由单个数字和运算符“+”、“-”、“\*”、  “/”、“(”、“)”构成，例如2+3\*(4+5)–6/4。  输入输出格式：  输入：第一行输入一个不超过100 的正整数n，代表表达式的个数。接下来  n 行，每行输入一个表达式，保证表达式内的数字为单个整数，表达式内各  运算符和数字间没有空格，表达式的长度不超过2000，且表达式合法。  输出：每行输出一个浮点数（保留两位小数），表示一个表达式的计算结果。   1. 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）   **栈的实现**  代码中使用了 ArrayStack 类来实现栈，其中有 ArrayStack<double> num 用于存储操作数，和 ArrayStack<char> op 用于存储操作符。  **中缀表达式转后缀表达式**  通过循环遍历输入的字符串 s，逐个处理字符串中的字符。  如果遇到数字字符，就将其解析为操作数，并将操作数压入 num 栈中。  如果遇到左括号 '('，将其压入 op 栈中。  如果遇到右括号 ')'，则弹出 op 栈中的操作符并计算，直到遇到左括号 '('。这样处理可以确保正确的运算顺序。  如果遇到操作符，需要比较其与 op 栈顶操作符的优先级。如果 op 栈顶操作符的优先级大于等于当前操作符，就弹出 op 栈顶操作符并计算，然后将当前操作符压入 op 栈中。  **eval 函数**  根据当前操作符从 op 栈中弹出两个操作数，并执行相应的计算操作。  计算结果再次压入 num 栈中。  这一过程重复进行，直到整个表达式的计算完成。  最后num栈中剩余的元素就是最终的计算结果   1. 测试结果（测试输入，测试输出）   输入:  3  1+6/1\*7+2\*1\*4+9/1+2\*0\*9+9+7/(9\*5)-1\*6-0\*8-7-9\*2+6-(0-5-2\*8-7-9\*5\*(6-5\*5\*2\*6-2-7-5+6\*7+6\*9-1\*0\*0+3\*0+2/1-6/6+5))  0-4-1/6\*(1-(6/7)-4+6+2+6\*1)-1\*7+2-8\*2+0-(4+6-6\*1+(3-8\*6/4-6-5)\*6/4/8+7-1\*4/9\*5)-0/6+1-0-2+7-2+6\*4-3\*6+2/8+6+1\*6\*2  5-3\*9+5/1\*5-9+1\*8-6-8-4\*1+5-2+9/3\*2-2/5/(2-6)\*2/7-9\*0-2+4/6\*6\*7\*8-8-8\*6+8\*9\*(3+0\*1/5/2\*7\*8+0-8\*8-5+8/5\*2-0)  输出  -9197.84  -3.47  -4362.57   1. 分析与探讨（结果分析，若存在问题，探讨解决问题的途径）   问题:  1 一开始没有注意到精度问题,所以需要对精度进行修改.数据类型为double  2 无法将连续的字符如”9””0”转化为整数90,后来自己查询资料实现了功能  3 一开始将数字和操作符放入同一个栈中,结果实现起来比较复杂且难以通过,因此切换思路,将数字和操作符分别放入两个栈中,从而实现了要求   1. 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<iostream>  using namespace std;  template<class T>  class stack  {  public:  virtual ~stack(){}  virtual bool empty() const =0;  virtual int size()const =0;  virtual T top()=0;  virtual void pop()=0;  virtual void push(const T& theElement)=0;  };  #ifndef myExceptions\_  #define myExceptions\_  #include <string>  #include<iostream>  using namespace std;  // illegal parameter value  class illegalParameterValue  {  public:  illegalParameterValue(string theMessage = "Illegal parameter value")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  // illegal input data  class illegalInputData  {  public:  illegalInputData(string theMessage = "Illegal data input")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  // illegal index  class illegalIndex  {  public:  illegalIndex(string theMessage = "Illegal index")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  // matrix index out of bounds  class matrixIndexOutOfBounds  {  public:  matrixIndexOutOfBounds  (string theMessage = "Matrix index out of bounds")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  // matrix size mismatch  class matrixSizeMismatch  {  public:  matrixSizeMismatch(string theMessage =  "The size of the two matrics doesn't match")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  // stack is empty  class stackEmpty  {  public:  stackEmpty(string theMessage =  "Invalid operation on empty stack")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  // queue is empty  class queueEmpty  {  public:  queueEmpty(string theMessage =  "Invalid operation on empty queue")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  // hash table is full  class hashTableFull  {  public:  hashTableFull(string theMessage =  "The hash table is full")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  // edge weight undefined  class undefinedEdgeWeight  {  public:  undefinedEdgeWeight(string theMessage =  "No edge weights defined")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  // method undefined  class undefinedMethod  {  public:  undefinedMethod(string theMessage =  "This method is undefined")  {message = theMessage;}  void outputMessage() {cout << message << endl;}  private:  string message;  };  #endif  #include"unordered\_map"  template<class T>  class ArrayStack :public stack<T>  {  public:  ArrayStack(int theCapacity=10);  ~ArrayStack(){delete[] stack;}  bool empty() const{return stackTop==-1;}  int size()const {return stackTop+1;}  T top()  {  if(stackTop==-1)  throw stackEmpty();  return stack[stackTop];  }  void pop()  {  if(stackTop==-1)  throw stackEmpty();  stack[stackTop--].~T();  }  void push(const T& theElement);  void clear()  {  stackTop=-1;  }  private:  int stackTop;  int arrayLength;  T\* stack;  };  template<class T>  void changeLength(T\*& a,int oldLength,int newLength)  {  if(newLength<0)  throw illegalParameterValue("the newlength must >0");  T \*temp=new T[newLength];  int size=min(oldLength,newLength);  copy(a,a+size,temp);  delete[]a;  a=temp;  }  template <class T>  inline ArrayStack<T>::ArrayStack(int theCapacity)  {  if(theCapacity<1)  throw illegalParameterValue("Capacity must >0");  arrayLength=theCapacity;  stack=new T[theCapacity];  stackTop=-1;  }  template <class T>  inline void ArrayStack<T>::push(const T &theElement)  {  if(stackTop+1==arrayLength)  {  changeLength(stack,arrayLength,arrayLength\*2);  arrayLength\*=2;  }  stack[++stackTop]=theElement;  }  int n;  ArrayStack<double>num;  ArrayStack<char>op;  unordered\_map<char,int>mp;  string s;  //进行计算  void eval()  {  double result=0;  double b=num.top();num.pop();  double a=num.top();num.pop();  char c=op.top();op.pop();  if(c=='+')  result=a+b;  else if(c=='-')  result=a-b;  else if(c=='\*')  result=a\*b;  else  result=a/b;  num.push(result);  }  int main()  {  cin>>n;  mp.insert({'+',1});mp.insert({'-',1});  mp.insert({'\*',2});mp.insert({'/',2});  while(n--)  {  num.clear();  op.clear();  cin>>s;  for(int i=0;i<s.size();i++)  {  char c=s[i];  if(isdigit(c))  {  int j=i;  double x=0;  while(j<s.size()&&isdigit(s[j]))  {  x=x\*10+s[j]-'0';  j++;  }  i=j-1;  num.push(x);  }  else if(c=='(')  {  op.push(c);  }  else if(c==')')  {  while(op.top()!='(')  eval();  op.pop();//删除(  }  else  {  while(op.size()&&mp[op.top()]>=mp[c])  eval();  op.push(c);  }  }  while(op.size())  eval();  printf("%.2f\n",num.top());  }  return 0;  } | | | |
|  | | | |