山东大学 软件学院 学院

软件测试 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201800301202 | 姓名：李成 | | 班级： 2018级软件4班 |
| 实验题目：堆栈的应用 | | | |
| 实验学时：4 | | 实验日期2019-11-14 | |
| 实验目的:  掌握堆栈的使用。 | | | |
| 硬件环境：  1、笔记本PC | | | |
| 软件环境：   1. Windows10 2. Eclipse 3. Java | | | |
| 实验步骤与内容：  实验内容：  1、输入一个数学表达式（假定表达式输入格式合法），计算表达式结果  并输出。  2、数学表达式由单个数字和运算符“+”、“-”、“\*”、“/”、“(、) ” 构成，例如 2+3\*(4+5)-6/4。  3、变量、输出采用整数，只舍不入。  实验步骤：   1. 先写一个计算没有括号的多项式的程序，用来计算。   int calculate(int num[] ,int n, char sym[],int s){  int result=num[0];    for(int i = 1 ; i < s ; i++){      if(sym[i-1]=='\*'){    result=result\*num[i];    }    if(sym[i-1]=='/'){    result=result/num[i];  }      if((sym[i]=='+'||sym[i]=='-')&&sym[i-1]=='+') {    result = result + num[i];    }    if((sym[i]=='+'||sym[i]=='-')&&sym[i-1]=='-') {    result = result - num[i];    }    if(sym[i]=='\*'&&(sym[i-1]=='+'||sym[i-1]=='-')){    num[i+1]=num[i]\*num[i+1];  sym[i]=sym[i-1];    }    if(sym[i]=='/'&&(sym[i-1]=='+'||sym[i-1]=='-')){    num[i+1]=num[i]/num[i+1];  sym[i]=sym[i-1];    }    if(i+2>=n) {        if(sym[i]=='\*')  result = result\* num[i+1];  if(sym[i]=='/')  result = result/ num[i+1];  if(sym[i]=='+')  result = result+ num[i+1];  if(sym[i]=='-')  result = result- num[i+1];    }      }  if(n<=2){  if(sym[0]=='\*')  result = result\* num[1];  if(sym[0]=='/')  result = result/ num[1];  if(sym[0]=='+')  result = result+ num[1];  if(sym[0]=='-')  result = result- num[1];      }    return result;  }  2.定义一个Node 用来进行链表操作，为以后的链表堆栈做准备。  class Node{    public:    char symbol;  int value;  Node \*next;    Node(char& s){    this->symbol = s;  this->value = NULL;  this->next = NULL;  }    Node(int& s){    this->symbol = NULL;  this->value = s;  this->next = NULL;  }    Node(char& s, Node \*n){    this->symbol = s;  this->value = NULL;  this->next = n;    }    Node(int& v , Node \*n ){    this->symbol=NULL;  this->value = v;  this->next = n;    }  };  3.堆栈  class stack{    private:    Node \*bottom;  Node \*top;  Node \*firstNode;    Node \*p;    int Size;      public:    stack(){    top = NULL;  bottom = NULL;  firstNode = NULL;    Size = 0;    }    bool empty(){    if(bottom==NULL)    return true ;    else    return false;    }    void push(char& s){    Node \*n = new Node(s);    if(bottom==NULL){    bottom = n;  top = bottom;    }    else{    n->next = top;    top = n;    firstNode = top;    }    Size++;  }    void push(int v){    Node \*n = new Node(v);    if(bottom==NULL){    bottom = n;  top = bottom;  }    else{    n->next = top;    top = n;    firstNode = top;    }    Size++;  }    void pop(){    if(top!=bottom)  {    top = top->next;    delete firstNode;    firstNode = top;    }  else {      delete top;    top = bottom = firstNode =NULL;    }      Size--;    }    Node\* read(){    if(top!=NULL){    return top;  }    else return NULL;    }      int work(){    Node \*p = top;    if(top==bottom){    return top->value;  }    int a = Size/2+1;  int s = Size/2;    cout<<a<<"~"<<endl;    int c1=0;  int c2=0;//计数器    int arr[a];  char sy[s];    for(int i = 0 ; i < a ; i++){    arr[i]=NULL;    }    for(int i = 0 ; i < s ; i++){    sy[i]=NULL;    }    p = firstNode;    int k = Size;    for(int i = 0 ; i < k;i++){    if(p->symbol==NULL){  arr[a-1-c1] = p->value;    p = p->next;    c1++;    }    else{    sy[s-1-c2] = p->symbol;    p = p->next;    c2++;  }    }  return calculate( arr , a , sy , s );  }  };  4.main（）  int main(){    stack polynomial;    char in[20];    int nn=19 ,ss=19,ii=0;//计数器    int num[nn];    char symbol[ss];    int number = 0;    for(int j = 0 ; j < 20 ; j ++ ){    in[j]=NULL;//标示符    }    cout<<"Input"<<endl;    cin>>in;    while(in[number]!=NULL){    number++;  }    int begin,end,c;    for(int i = 0 ; in[i]!=NULL ; i++){    nn=19;  ss=19;    while(in[i]!=NULL&&in[i]!=')'){    if(judge(in[i])){//是数字    c = in[i]-48;    polynomial.push(c);  }  else  {    polynomial.push(in[i]);    }    i++;  }    //a[i]==')'未入    while(polynomial.empty()==false&&polynomial.read()->symbol!='('){    ii++;    if(polynomial.read()->symbol==NULL){//数字    num[nn]=polynomial.read()->value;  polynomial.pop();  nn--;    }  else{    symbol[ss]=polynomial.read()->symbol;  polynomial.pop();  ss--;  }    }  if(polynomial.empty()==false) {    polynomial.pop();    }    int num\_n = (ii+1)/2;  int sym\_n = ii/2;    ii=0;    int num2[num\_n];  char symbol2[sym\_n];    for(int i = 0 ; i < num\_n ; i++){    num2[i]=NULL;    }    for(int i = 0 ; i < sym\_n ; i++){    symbol2[i]=NULL;    }      for(int i = 0 ; i < num\_n ; i++){    num2[i]=num[19-num\_n+1+i];  }    for(int i = 0 ; i < sym\_n ; i++){    symbol2[i] = symbol[19-sym\_n+1+i];    }    polynomial.push(calculate(num2, num\_n, symbol2,sym\_n));    }  cout<<"Output"<<endl;  cout<<polynomial.work()<<endl;  cout<<"End"<<endl;  } | | | |
| 结论分析与体会：代码的主要难度就是堆栈的应用以及多项式情况的分析，就是运算顺序以及遇到右括号出栈的判断。让我清晰的感受就到了堆栈的实用性。 | | | |