**实 验 报 告**

**（与程序设计有关）**

**课程名称： 数据结构**

**实验题目： 栈和队列**

**班级学号：** 2203050320

**姓 名:** 闻家尉

**成 绩：**

**沈 阳 理 工 大 学**

**2023年 11 月 8 日**

|  |
| --- |
| **实验目的及要求：**  1.掌握栈的顺序表示和实现。 |
| **软硬件环境：**计算机一台，Turbo C 软件或VC++ 软件 |
| **算法或原理分析（实验内容）：** 1.问题描述在计算机中，算术表达式由常量、变量、运算符和括号组成。由于不同的运算符具有不同的优先级，又要考虑括号，因此，算术表达式的求值不可能严格地从左到右进行。因而在程序设计语言编译过程中，借助栈实现算术表达式的求值。2.基本要求输入一个算术表达式，由常量、变量、运算符和括号组成（规定：操作数只能为正整数，而操作符只能是＋、－、×、/这几个二元运算符，用“#”表示表达式结束），输出对算术表达式计算的结果。3.算法提示（1）算术表达式变换成无括号的表达式，假设运算符的优先级为：（）、×/、＋－； ①首先将一左括号“（”入栈，作为栈底元素；  ②从左至右对算法表达式进行扫描，每次读一个字符；  ③若遇到左括号“（”，则压栈； ④若遇到操作数，则立即输出；⑤若遇到运算符，如果它的优先级比栈顶元素的优先级高，则直接进栈，否则输出栈顶元素，直到新栈顶元素的优先级比它的低，然后将它压栈；⑥若遇到右括号“）”，则将栈顶的运算符输出，直到栈顶元素为左括号“（”，然后，左右括号互相抵消；⑦当输入符为“#”时，表明表达式串已全部输入，将栈中的运算符全部输出，并删除栈底的左括号。（2）对无括号的算术表达式求值输入无括号的表达式；若遇数值，操作数进栈；若遇运算符，让操作数栈的栈顶和次栈顶依次出栈并与此运算符进行相应的运算，运算结果入操作数栈；重复步骤②、③，直到输入为“#”，则此时栈中的结果便是所求的无括号表达式的值。 |

|  |  |  |  |
| --- | --- | --- | --- |
| **程序代码或实现过程：**  #include<iostream>  #include<cstdlib>  #include<cmath>    #define MAXSIZE 20  #define OK 1  #define ERROR 0  #define TRUE 1  #define FALSE 0    using namespace std;    typedef int DataType;    typedef struct  {  DataType data[MAXSIZE];  int top;  }SqStack;    int InitStack(SqStack &S)  {  S.top=-1;  return OK;  }    int StackEmpty(SqStack S)  {  return(S.top==-1?TRUE:FALSE);  }    int StackFull(SqStack S)  {  return(S.top==MAXSIZE-1?TRUE:FALSE);  }    int Push(SqStack &S,DataType e)  {  if(StackFull(S))  return ERROR;  S.top++;  S.data[S.top]=e;  return OK;  }    int Pop(SqStack &S,DataType &e)  {  if(StackEmpty(S))  return ERROR;  e=S.data[S.top];  S.top--;  return OK;  }    DataType GetTop(SqStack S)  {  DataType e;  if(StackEmpty(S))  return ERROR;  e=S.data[S.top];  return e;  }    char Precede(char a,char b)  {  char z;  if((b=='+')||(b=='-')||(b=='\*')||(b=='/')||(b=='(')||(b==')')||(b=='='))  switch(a)  {  case '+':  case '-':  if((b=='\*')||(b=='/')||(b=='('))  z='<';  else  z='>';break;  case '\*':  case '/':  if(b=='(')  z='<';  else  z='>';break;  case '(':  if (b=='=')  z='E';  else if(b==')')  z='=';  else z='<';  break;  case ')':  if(b=='(')  z='E';  else  z='>';  break;  case '=':  if(b=='=')  z='=';  else if(b==')')  z='E';  else z='<';break;  }  else z='E';  return(z);  }    int In(char ch)  {  int i,flag=0;  char op[7]={'+','-','\*','/','(',')','='};  for(i=0;i<7;i++)  {  if(ch==op[i])  {  flag=1;break;  }  }  return flag;  }    DataType Operate(DataType a,char theta,DataType b)  {  DataType z;  switch(theta)  {  case '+':z=a+b;break;  case '-':z=a-b;break;  case '\*':z=a\*b;break;  case '/':z=a/b;break;  }  return(z);  }    int CaculateExpression()  {  SqStack optr,opnd;  DataType x,theta,a,b,s;  char c;  c=getchar();  InitStack(optr);  Push(optr,(DataType)'=');  InitStack(opnd);    while(c!='=' || (char)GetTop(optr)!='=')  {  if(!In(c))  {  s=c-48; // s=c-'0';  Push(opnd,(DataType)s);  c=getchar();  }  else  switch(Precede((char)GetTop(optr),c))  {  case '<':  Push(optr,(DataType)c);  c=getchar();  break;  case '=':  Pop(optr,x);  c=getchar();  break;  case '>':  Pop(optr,theta);  Pop(opnd,b);  Pop(opnd,a);  Push(opnd,Operate(a,(char)theta,b));  break;  case 'E':printf("表达式中括号不匹配！");exit(1);  }  }  return GetTop(opnd);  }    int main()  {  printf("%d\n",CaculateExpression());  return 0;  } | | | |
| **结果分析：** | | | | |
| **教师签字** |  | **日 期** |  | |

*双面打印*