实验二 栈

班级**: 计算机17-1班** 姓名**: 冯子旋** 学号：07号

# 一、实验目的

1．掌握顺序栈的存储方式及基本操作的实现；

2．掌握利用栈解决问题的方法；

# 二、实验题目与要求

题目：顺序栈及其应用要求：

（1）实现顺序栈的类型定义及初始化、入栈、 出栈作、取栈顶等操作；

（2）利用栈，解决括号配匹检测问题；

# 三、实验内容

1．顺序栈的实现

完成顺序栈的类型定义及其基本操作函数代码，并将其保存为 “SqStack.h”。设计完成空缺代码并上机调试。

//顺序栈类型及其基本操作的实现

#define StackSpaceIncr 20 typedef struct

{ SElemType \*Elem; int top;

int stackSize;

}SqStack; //顺序栈类型

Status InitSqStack(SqStack &S, int InitSize) {

//初始化操作

S.Elem=(SElemType \*)malloc(InitSize\*sizeof(SElemType));

if(!S.Elem) return OVERFLOW;//栈存储空间分配失败

S.stackSize=InitSize;

S.top=0; //置为空栈

return OK;

}

Status stackIsEmpty(SqStack S)

{ //判断栈 S 是否为空

if(!S.top) return TRUE;

else return FALSE;

}

void clearStack(SqStack &S)

{ //将顺序栈 S 清空

S.top=0;

}

int stackLength(SqStack S)

{ //求顺序栈 S 的长度

return S.top;

}

Status Push(SqStack &S, SElemType e)

{ //对于顺序栈 S，将元素 e 入栈

SElemType \*newBase;

if(S.top==S.stackSize){//若当前栈的空间已满，则重新获取更大空间

newBase=(SElemType \*)realloc (S.Elem,(S.stackSize+StackSpaceIncr)\*sizeof(SElemType));

if(!newBase) return OVERFLOW;//扩充失败

S.Elem=newBase;

S.stackSize+=StackSpaceIncr; //扩充成功

}

S.Elem[S.top]=e; //插入元素e

S.top++; //栈顶指针top上移一个位置

return OK;

}

Status Pop(SqStack &S, SElemType &e)

{ //将顺序栈 S 的栈顶元素出栈

if(!S.top) return ERROR;//栈空 操作失败

S.top--; //栈顶指针top下移一个位置

e=S.Elem[S.top];

return OK;

}

Status getTop(SqStack S,SElemType &e)

{ //取顺序栈 S 的栈顶元素

if(!S.top) return ERROR; //栈空 操作失败

e=S.Elem[S.top-1]; //由参数e返回栈顶元素的值

return OK;

}

#include <stdio.h> #include <stdlib.h> #include "Status.h"

typedef SElemType; //定义栈中元素类型

#include "SqStack.h"

int bracketsCheck( ) { //输入表达式串，检查括号匹配情况

char ch;

SqStack S;

InitSqStack(S,30);

ch=getchar();

while(ch!='\n'){

if(ch=='(') Push(S,ch); //为左括号为入栈

else if(ch==')'){//为右括号

if(!stackIsEmpty(S)) Pop(S,ch);//栈非空则出栈匹配

else return -1;//栈空说明缺与ch匹配的左括号 }

ch=getchar(); }

return stackLength(S);

}

int main() {

//用于测试括号匹配检验算法的主函数

int result;

printf("输入表达式串\n");

result= bracketsCheck();

if(result==0) printf("括号匹配正确！\n");

else if(result>0) printf("错误，缺%d个\')\'!\n",result);

else printf("错误，缺一个或多个\'(\'!\n");

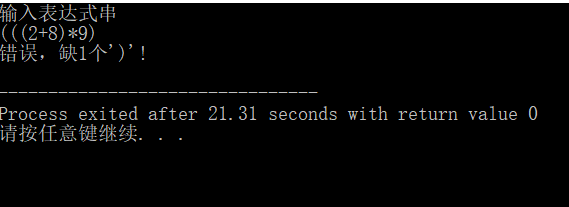
system("pause"); return 0;

3．程序运行结果

至少运行 3 次程序，每次输入一个表达式串

（如“a\*((a+b)/(x-y)-6)”）给出测试结果。

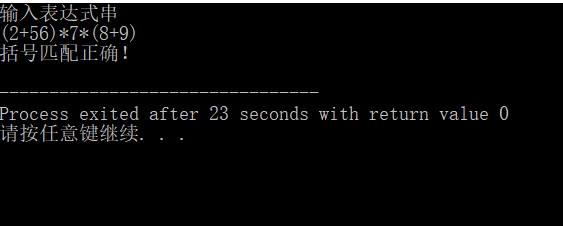
测试结果 1： 输入：(((2+8)\*9)

输出：

测试结果 2： 输入：

(2+56)\*7\*(8+9)

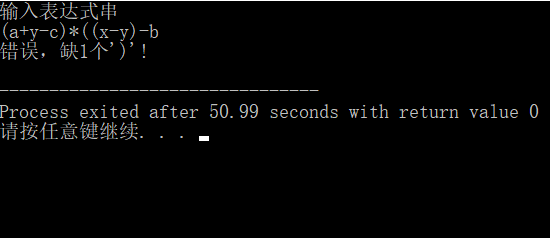
输出：



测试结果 3： 输入：

(a+y-c)\*((x-y)-b

输出：



# 成绩：