简单计算器的说明文档

--姓名：强波
--学号：14331229

为了实现简单的计算器功能，我使用的方法是将表达式转变成逆波兰式，然后再对逆波兰式求值。

原理说明

转变的步骤如下：

(1)首先，需要分配 2 个栈，栈 s1 用于临时存储运算符（含一个结束符号），此运算符在栈内遵循越往栈顶优先级越高的原则；栈 s2 用于输入逆波兰式，为方便起见，栈 s1 需先放入一个优先级最低的运算符，在这里假定为'#'；

(2)从中缀式的左端开始逐个读取字符 x，逐序进行如下步骤：

1.若 x 是操作数，则分析出完整的运算数（在这里为方便，用字母代替数字），将 x 直接压入栈 s2；

2.若 x 是运算符，则分情况讨论：

若 x 是'('，则直接压入栈 s1；

若 x 是')'，则将距离栈 s1 栈顶的最近的'('之间的运算符，逐个出栈，依次压入栈 s2，此时抛弃'('；

若 x 是除'('和')'外的运算符，则再分如下情况讨论：

若当前栈 s1 的栈顶元素为'('，则将 x 直接压入栈 s1；

若当前栈 s1 的栈顶元素不为'('，则将 x 与栈 s1 的栈顶元素比较，若 x 的优先级大于栈 s1 栈顶运算符优先级，则将 x 直接压入栈 s1。否者，将栈 s1 的栈顶运算符弹出，压入栈 s2 中，直到栈 s1 的栈顶运算符优先级别低于（不包括等于）x 的优先级，或栈 s2 的栈顶运算符为'('，此时再则将 x 压入栈 s1;

(3)在进行完(2)后，检查栈 s1 是否为空，若不为空，则将栈中元素依次弹出并压入栈 s2 中（不包括'#'）；

(4)完成上述步骤后，栈 s2 便为逆波兰式输出结果。但是栈 s2 应做一下逆序处理，因为此时表达式的首字符位于栈底；

然后是逆波兰式求值，首先从操作符栈顶取出一个操作符，然后从操作数栈顶连续取出 2 个操作数，接着对这两个数和操作符进行计算，运算结果压回操作数栈中。如果逆波兰式是正确的，到了最后，操作符栈为空，操作数栈中恰好只有一个数，这就是运算结果。

测试如下：