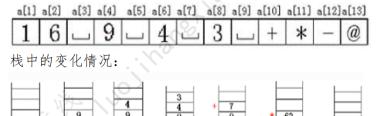
1331: 【例1-2】后缀表达式的值

题目描述

从键盘读入一个后缀表达式(字符串),只含有0-9组成的运算数及加(+)、减(—)、乘(*)、除(/)四种运算符。每个运算数之间用一个空格隔开,不需要判断给你的表达式是否合法。以@作为结束标志。

比如, 16-9*(4+3) 转换成后缀表达式为: $16\Box 9\Box 4\Box 3\Box +*-$, 在字符数组A中的形式为:



运行结果: -47

提示:输入字符串长度小于250,参与运算的整数及结果之绝对值均在264范围内,如有除法保证能整除。

输入格式

一个后缀表达式。

输出格式

一个后缀表达式的值。

输入样例

16 9 4 3 +*-@

输出样例

-47

解析

后缀表达式是典型的利用栈来进行解析的题目,我们不断的将数字读入栈中,遇到符号时,则取出最近的两个数字进行计算,然后将结果继续存入即可。

```
#include <bits/stdc++.h>
using namespace std;
//待计算参数栈
stack<long long> n;
//经过计算后,读入的真实数值
long long s = 0;
int main() {
   char ch;
   //使用do while进行循环读入
  do {
       ch = getchar();
       //读入一个数字字符,进行计算
     if (ch >= '0' && ch <= '9') {
           s = s * 10 + ch - '0';
          //遇到空格说明遇到数字的结尾了,将计算完毕的数字存入
      else if (ch == ' ') {
          n.push(s);
           //注意一定要清空数据
         s = 0;
       }
          //说明遇到了计算符号
      else if (ch != '@') {
          //一定注意这里读的顺序,第二个参数在后
         //第二个参数
         long long x = n.top();
          n.pop();
          //第一个参数
         long long y = n.top();
          n.pop();
          //根据当前的符号进行计算
         switch (ch) {
              case '+':
                  n.push(x + y);
                 break;
              case '-':
                  n.push(y - x);
                 break;
              case '*':
                  n.push(x * y);
                 break;
              case '/':
                  n.push(y / x);
                  break;
          }
```

```
} while (ch != '@');
//取出最后的计算结果
printf("%lld\n", n.top());
return 0;
}
```

逻辑航线培优教育、信息学奥赛培训专家。

扫码添加作者获取更多内容。



Etial Junihanetian Alle Junihanetian Alle Junihanetian