浙江大学实验报告

专业：计算机科学与技术
姓名：蔡雨谦
学号：3210102466
日期：2022.05.04

课程名称：C 程序设计专题　　指导老师：翁恺　　成绩：
实验名称：作业 3：表达式计算

一、实验题目要求

Write a program that reads an expression in a line as input and prints out the result. Only integers and operators below are allowed in the expression:

```
+ - * / % ( )
```

二、实验思路和过程描述

一个表达式可由符号划分成若干表达式，因此考虑可以由递归实现。首先在全局范围定义一个字符串，整个字符串的内容自始至终不发生变化，以子字符串的长度和每个字符的位置（下标）为突破口。

1、base case：表达式最终会划分为若干无符号的数字串，base case 就是将这些数字串转换成对应的数并返回。

实现方式：定义一个函数 num(int left , int right)，参数为数字串的两端的下标。从最左端开始，每次取一位数，乘以 10 后再加上后一位数，依此遍历整个数字串，返回总和。

2、定义递归函数: double part(int l ,int r)，参数为字符串两端的下标。

实现方式：

（1）记录符号位置

初始化符号的位置：add=0（和‘+’同级的符号），mul=0（和‘*’同级的符号），mns =-1（负号）。括号的状态：par =0（数值表示左括号与右括号数量的差值）。

i 从 l 开始遍历字符串，遇见‘（’par++ ，遇见‘）’par--。遇见‘+’add=i;遇见‘*’‘/’‘%’mul = i; 遇见‘-’时判断前一位是否为（，*，/，%，+或-，若是则 add=i,反之 mns=i。

这一步记录了各优先级最后一个符号的位置。

然后判断是否为 base case：if(!par&&!add&&!mul)

(2)若是 base case，判断是否有负号或括号，有负号就返回：-part(l+1,r)，无负号且有括号就返回：part(l+1,r-1)，无负号无括号就返回 num(l,r)。

（3）若非 basecase，先判断 add 是否存在，存在则返回 part(l,add-1) +或- part(add+1,r)。再判断 mul 是否存在，存在则返回 part(l,mul-1) *或/ part(mul+1,r) ， 或 （int） part(l,mul-1) % (int)part(mul+1,r)。

至此递归函数得以实现

3、int main()

{

gets(n);

printf("%d",(int)part(0,strlen(n)-1));

return 0;

}

三、实验代码解释

1、

```
double part(int l, int r)
{
    int par = 0, add = 0, mul = 0,mns = -1;
    for(int i = l; i ≤ r; i++)
    {
        if(n[i] == '(' )par++;
        else if(n[i] == ')')par--;
        else if(!par)
        {
            if(n[i]=='+')add=i;
            else if(n[i]=='*'||n[i]=='/'||n[i]=='%')mul=i;
            else if(n[i]=='-')
            {
                if(i==0)mns=i;
                else if(n[i-1]=='('||n[i-1]=='*'||n[i-1]=='/'||n[i-1]=='%'||n[i-1]=='+')mns=i;
                else add=i;
```

如图是用于记录符号位置的代码。

（1）当 par 不为 0 时，表示当前处理位置在括号内部，此时不记录运算符号，将括号内容当作整体处理。进括号时 par 为正，出括号时 par 变为 0，表示一对括号处理结束，开始记录运算符号。

（2）n[i]=‘-’时，若 i==0，或前一位是非‘）’的符号，说明此时‘-’为负号，mns=i。否则‘-’为减号，add=i。

2、

```
if(add)
{
    if(n[add] == '+')return part(l, add-1) + part(add+1, r);
    else if(n[add]=='-')return part(l, add-1) - part(add+1, r);
}
if(mul)
{
    if(n[mul] == '*')return part(l, mul-1) * part(mul+1, r);
    else if(n[mul]=='/') return part(l, mul-1) / part(mul+1, r);
    else if(n[mul]== '%')return (int)part(l, mul-1) % (int)part(mul+1, r);
}
```

（1）add 放在 mul 前判断的原因：划分后的表达式首先分别计算，表达式之间的连接符号后计算，所以应从优先级低的符号处划分。

（2）%只能用于整型数的计算，所以计算前先改变类型。

四、实验心得与体会

1、num 函数和 part 函数若定为 int 类型，则计算过程中误差过大，所以采用 double 类型，最后输出的时候转为 int 类型。

2、为防止括号内的内容被拆散，在记录符号位置时，对括号内的符号不予记录。

3、负号可以位于字符串起始端，所以 mns 初始化时不能为 0。

4、负号和减号的差别可通过前一位是不是某些特定符号来判断。