**哈尔滨工程大学**

**《计算思维（二）》实验报告**

**基础实践一**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 班级： |  |  |
|  | 学号： |  |  |
|  | 姓名： |  |  |

实验时间： 2024 年 第02~12周

|  |  |
| --- | --- |
| 成绩 |  |

**哈尔滨工程大学计算机教育与实验创新中心**

实验一：运行环境及顺序结构

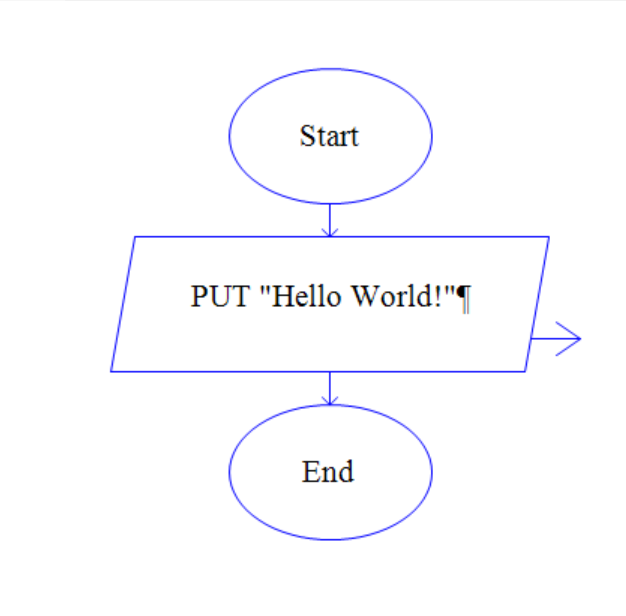
实验题目1：C编程环境

编写C程序，输出“Hello World”。编辑完成后存盘、编译、连接、运行并查看结果。

设计思想：

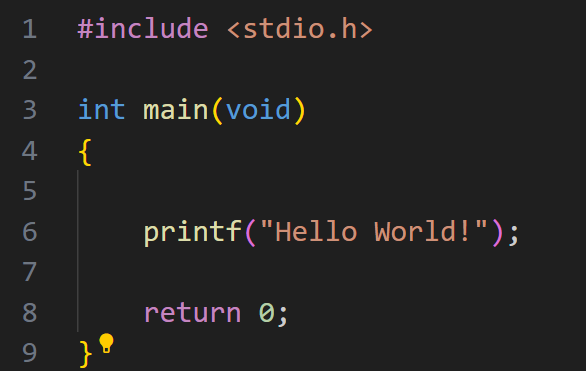
只需要导入头文件，用printf 函数打出“Hello World” 就可以啦！

Raptor 算法流程图如下：



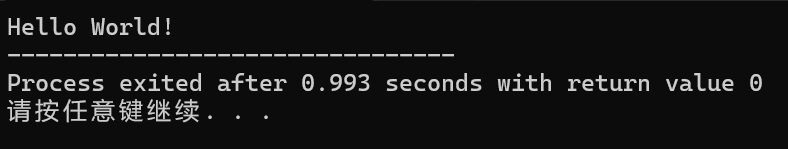
实验代码、验证与结论：

实验代码如下：



经验证，编译及运行结果如下：





与样例输出相同，且可以通过评测。

总结与心得体会：

第一题还算比较顺利地完成了。

发现程序先编译而成了一个后缀名为 .exe 的可执行文件，然后再运行。如果代码不符合编程格式规范或有语法错误，编译器会给出警告或报错。

将return 0改为return 1，发现运行结束后的语句也相应改变了。



并且，在点击 .exe 文件直接运行时，为了不让黑框框一闪而过，除了老师所讲的引用新的头文件的做法，我发现也可以在后面添加个 scanf 函数。但这样，我们虽可以让程序“暂停”，但其实程序还没有结束，我们无法看到结束后的返回值。在某些情况下，可能就无法依此判断我们的程序是否有误。

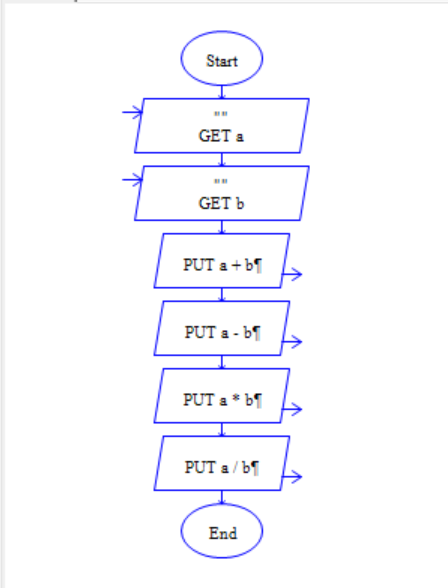
实验题目2： 简单的四则运算实现

由键盘输入两个整数，分别计算这两个数的和、差、积、商，并输出结果。

设计思想：

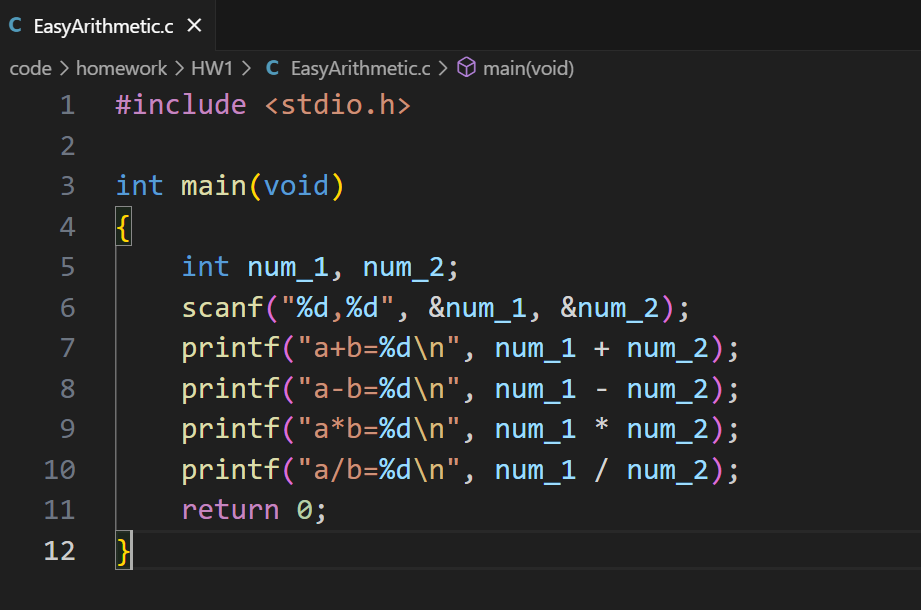
将输入的两个值存进表达式，然后对其进行和差积商运算即可。

Raptor 算法流程图如下：

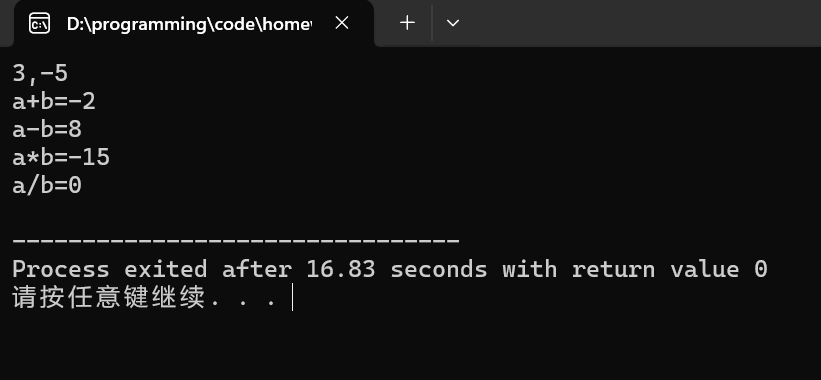


实验代码、验证与结论：

实验代码如下：



经验证，编译及运行结果如下：



­

与样例输出相同，且可以通过评测。

总结与心得体会：

这里，由于两个变量都是整数，所以进行除法运算的结果一定也为整数。注意看题目的样例输出，结果也是整数0，因此我们不用更改变量类型或做类型转换。

不过如果想转化为浮点数的输出，既可以将变量的类型更改为float，亦可以在表达式前先乘以1.0。不过无论哪种改法，都不要忘记将占位符更改为 %f 。

实验题目3： 字母密码

从键盘任意输入5个小写英文字母，将其加密后输出。加密规则是：用原来英文字母后面的第4个字母代替原来的字母。例如，小写字母a后面第4个字母是e，e代替a。再如，大写字母w后面第4个字母是a，a代替w。请编程实现上述加密方案。

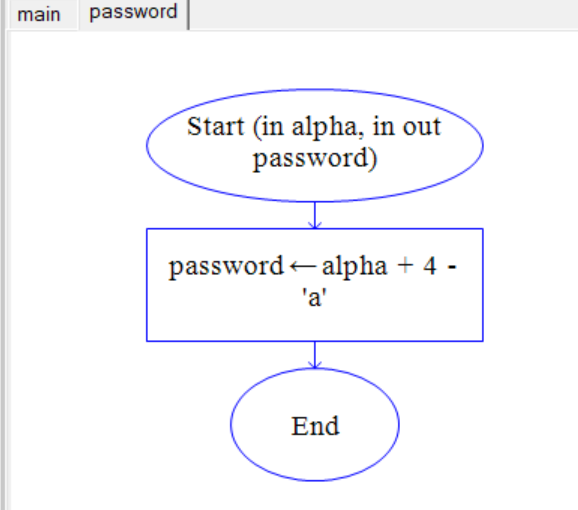
设计思想：

将字母向后移动四位后与字符 ’a’ 作差，之后除以2取余数即可。

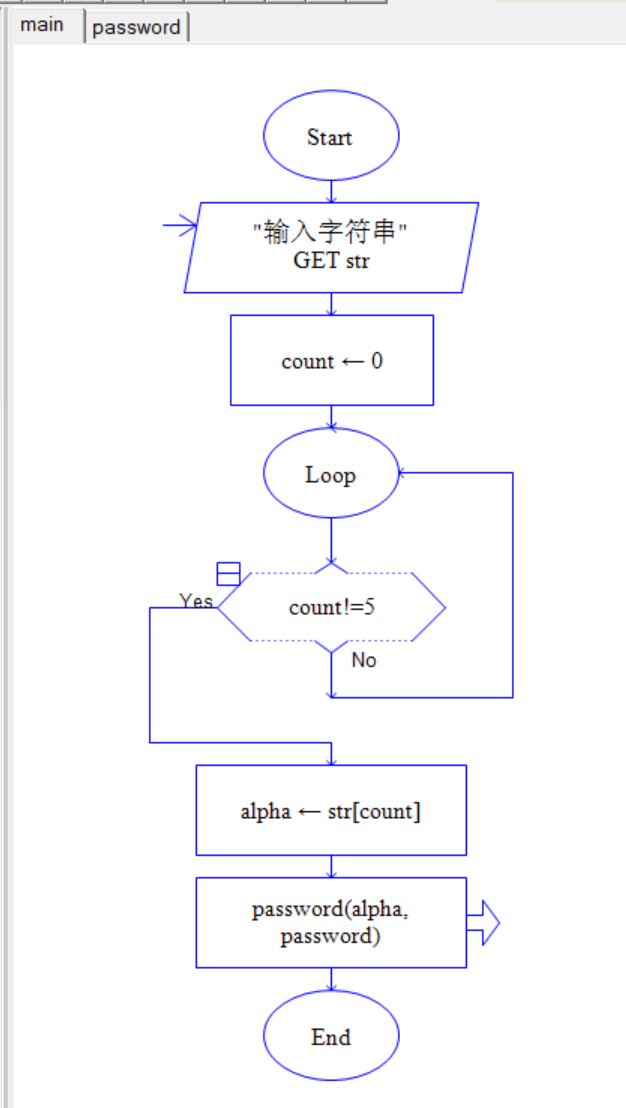
我把输入的5个字母视为字符串，将加密的功能封装在了password函数了里，循环遍历每个字符，将其经过password函数改造，最后一起输出。

Raptor 算法流程图如下，

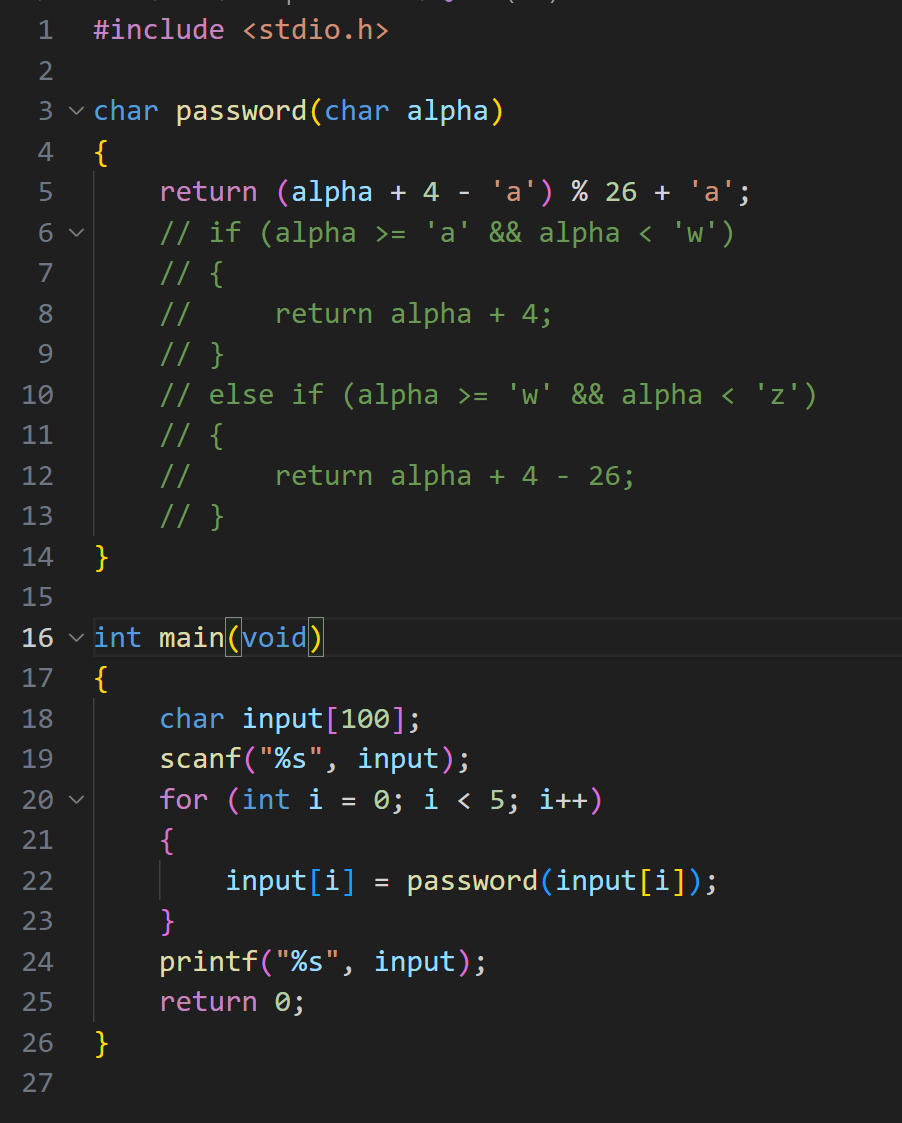
功能函数：



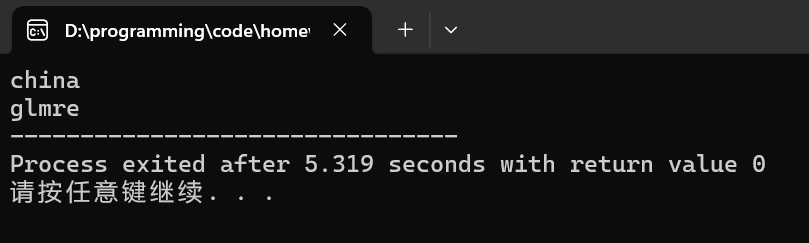
主函数：



实验代码、验证与结论：



经验证，编译及运行结果如下：





与样例输出相同，且可以通过评测。

总结与心得体会：

一开始我是采用if - else语句来进行字母的回环的，可是后来老师在实验课上点拨了我们：数本是无穷的，但有了余数，数就有了归类。所以我又尝试了取余，果然化繁为简！

实验题目4： 物品折旧计算器

某种物品的每年折旧费的线性计算方法如下：每年折旧费=（购买价格-废品价值）/产品设计寿命（年）。而折旧价值的计算方法如下：购买价格-每年折旧费\*使用年限。

请编写一个程序，当输入物品的购买价格、废品价值、产品设计寿命和使用年限后，程序能计算出该物品的每年折旧费（结果保留两位小数），以及在到达某使用年限时的折旧价值（即残余价值，结果保留两位小数）。

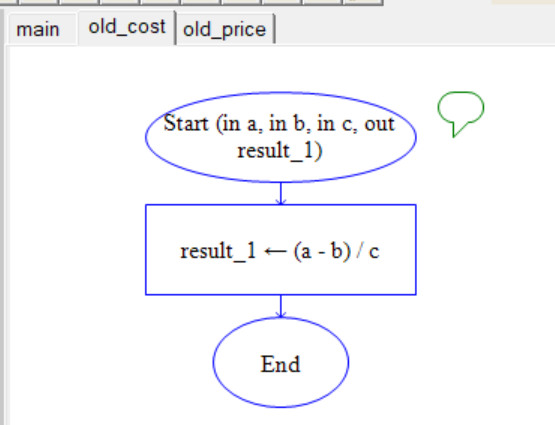
设计思想：

这道题主要就是套公式嘛，只要认真读题就可以。

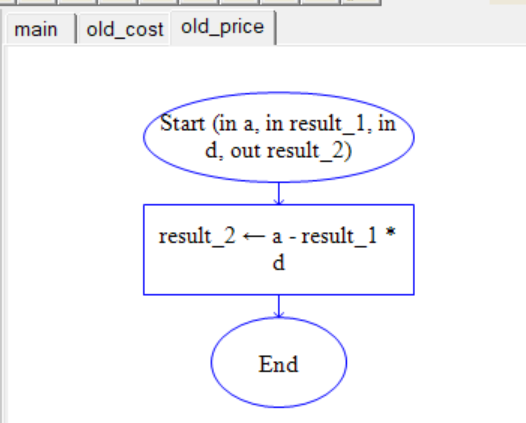
需要注意的是，第二个函数的参数是第一个函数的结果。

Raptor 算法流程图如下，

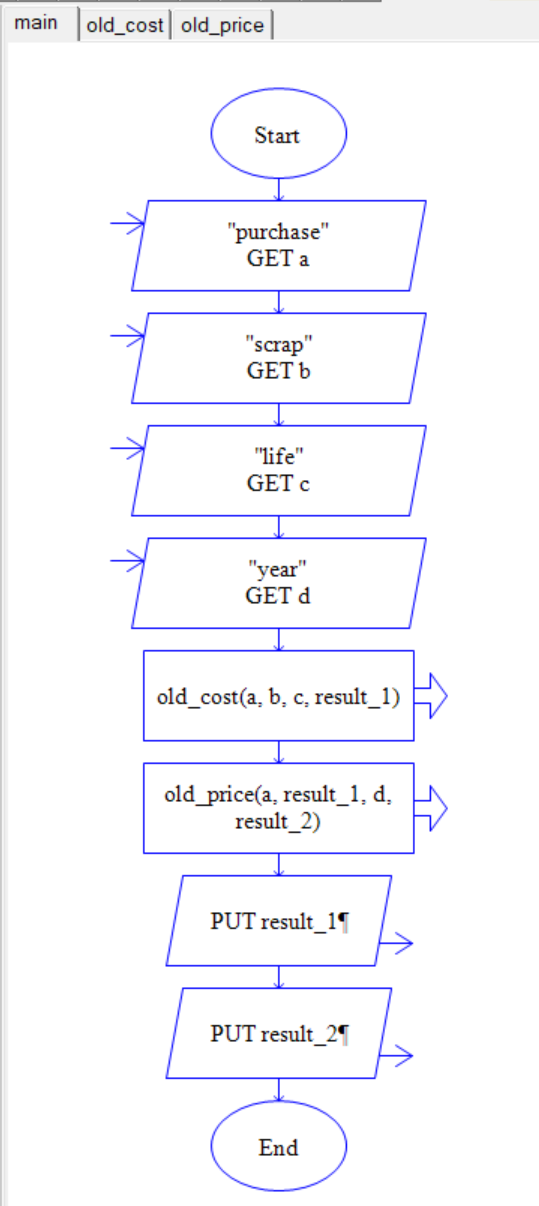
old\_cost函数：



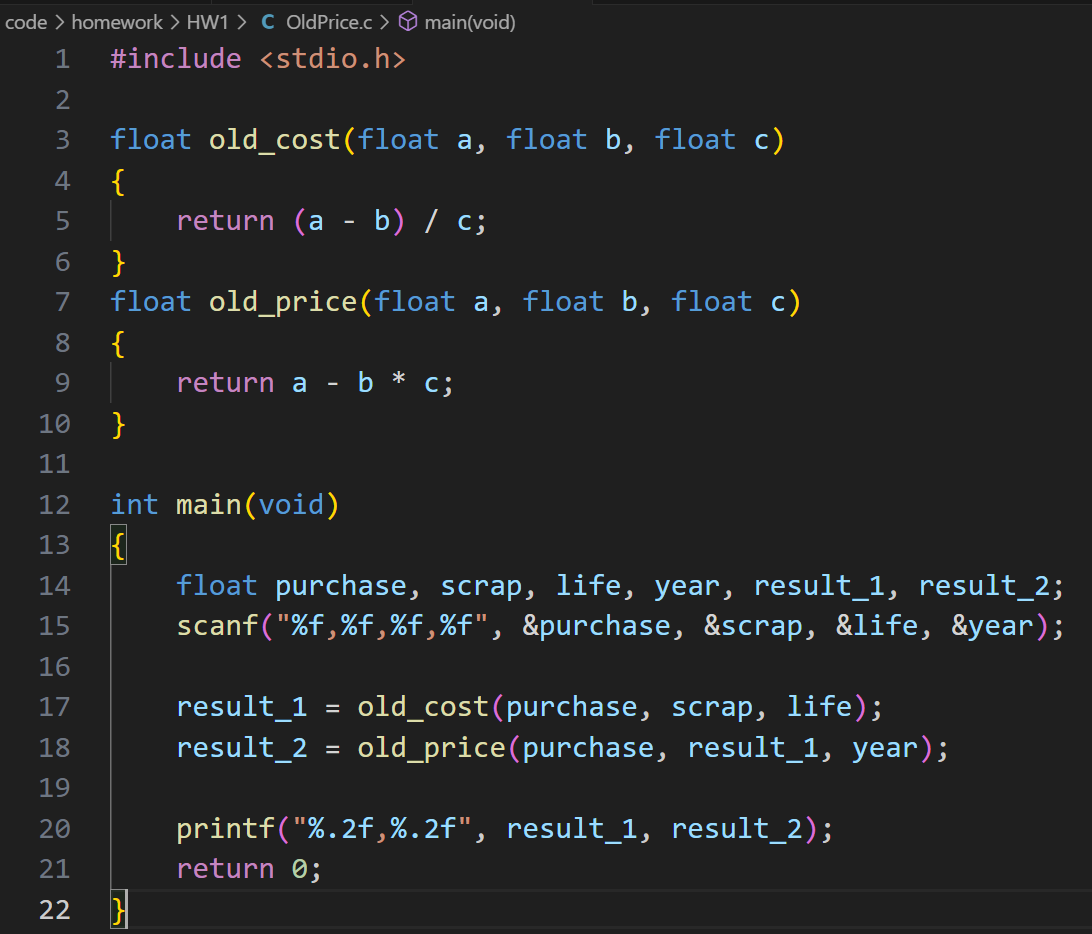
old\_price 函数：



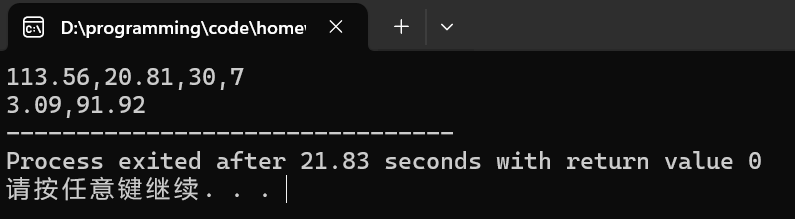
主函数：



实验代码、验证与结论：



经验证，编译及运行结果如下：





与样例输出相同，且可以通过评测。

总结与心得体会：

这道题中比较复杂而易错的是传入函数的参数太多太混乱，如果变量的命名不简短易懂，就很容易做错。

我的室友在做这道题时，就由于他的变量英文命名过于冗长且缺乏区分度，导致传入函数的一个参数填错了，我们一起找了半个小时才找出来这个问题。当天晚上，我甚至还因此做了一个噩梦，梦里，我一直在改代码，直到梦醒天亮，都没有找到程序的bug。

变量的命名应该简洁而规范，这样既易于更改和纠错，又方便后续阅读和他人协作。

实验二：选择结构

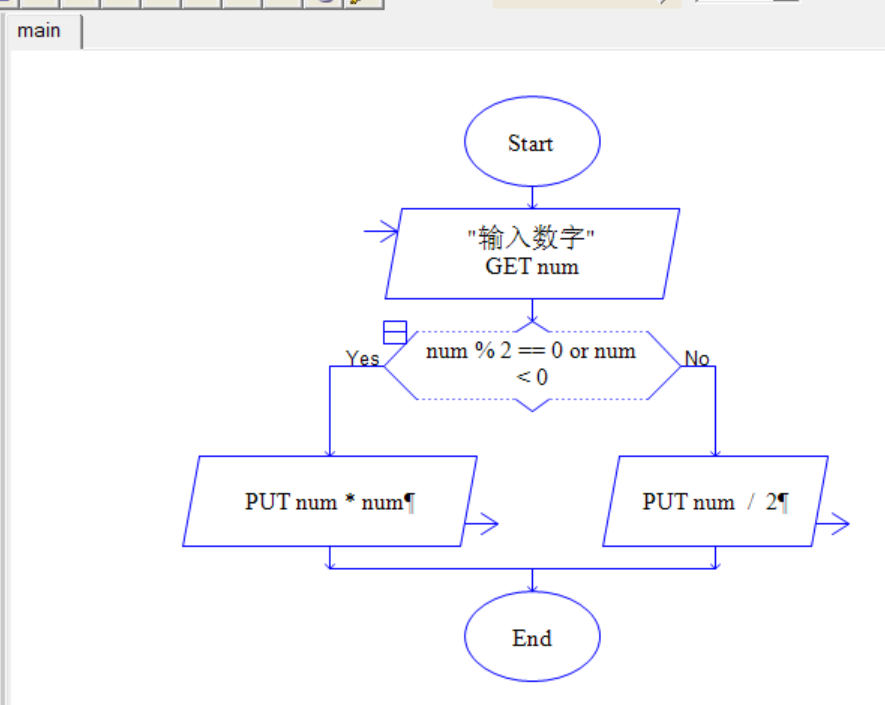
实验题目1：if双分支结构简单应用

输入一个整数，如果输入的数是偶数或者负数，则输出其平方，否则输出其一半的值。

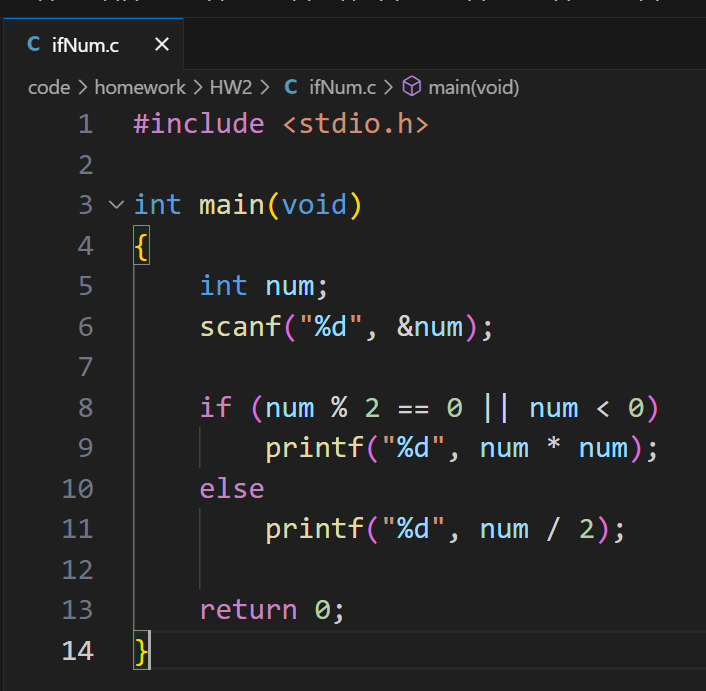
设计思想：

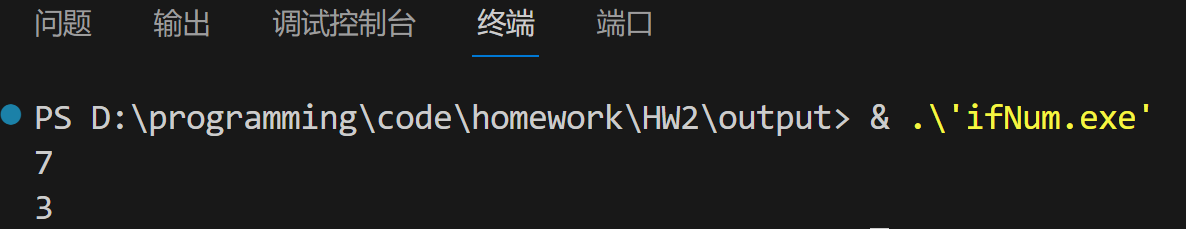
比较基础的一道题，使用逻辑运算符 || 即可。

Raptor算法流程图如下：



实验代码、验证与结论：





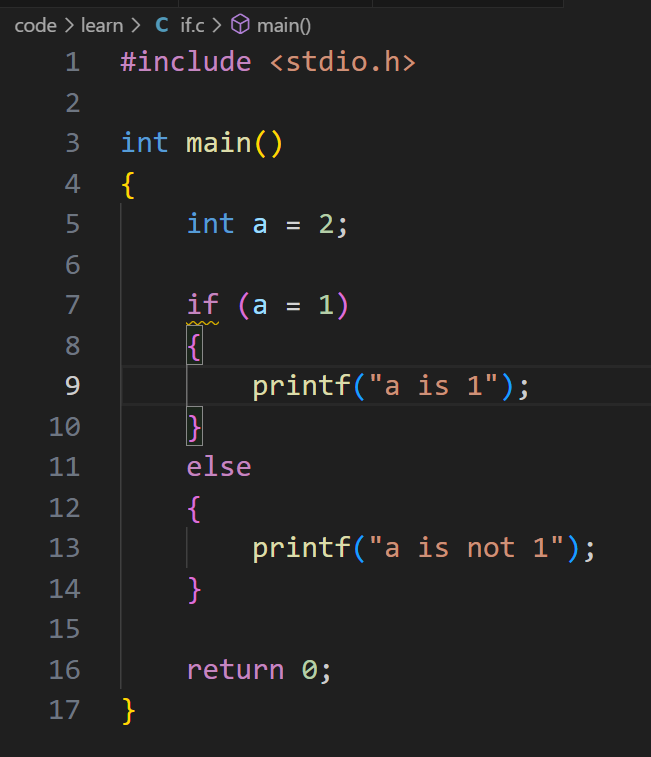
输出结果与样例相同，评测通过。



总结与心得体会：

在涉及到逻辑运算时，发现大家都非常容易将逻辑运算符“==”写错成赋值符“=”。尤其是在判断左侧变量里的值是否等于右侧表达式时，很容易出现这种逻辑错误却无法轻易发现。例如下面的一个代码：

由于第7行if 括号内表达式返回的值为1 (True)，else语句永远不会被执行！





发现用word文档写报告有一些不太方便的地方，如果实验报告可以以markdown的格式提交，我想无论是老师制作这篇报告以及排版，还是学生提交代码和图片，这样的效率或许会更高一些叭？

实验题目2： 商品优惠计算器

使用if语句编程实现输入购货金额，输出实际付款金额。购货折扣率如下：

购货金额≤500元 不打折；

500元<购货金额≤1000元 9折；

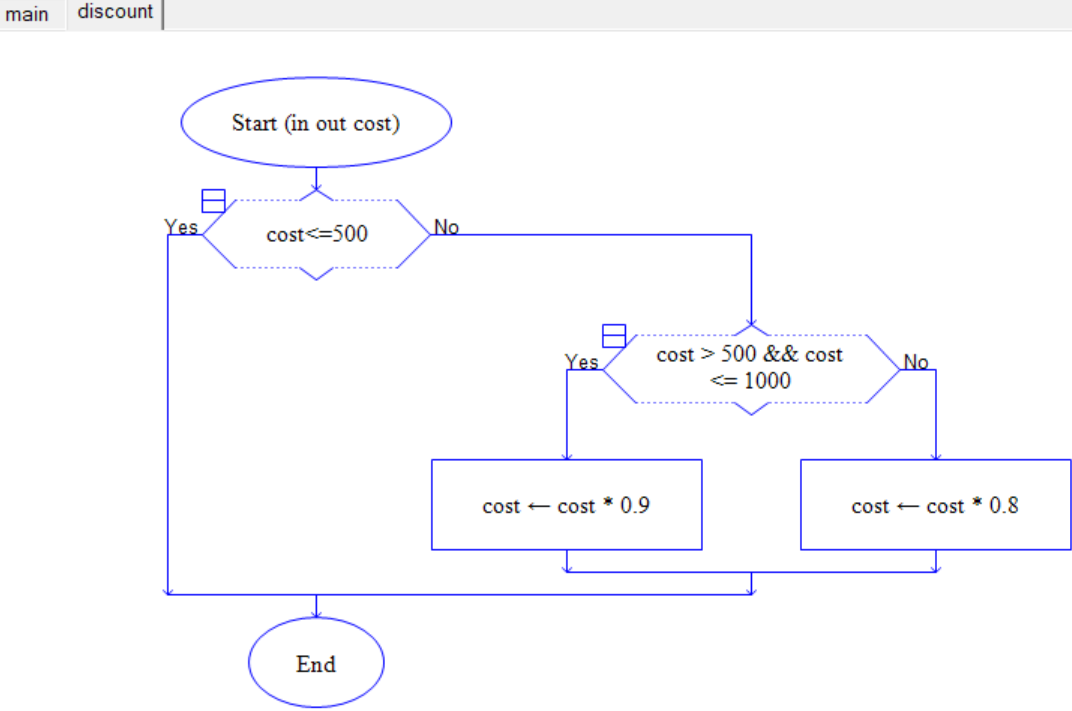
1000元<购货金额 8折；

设计思想：

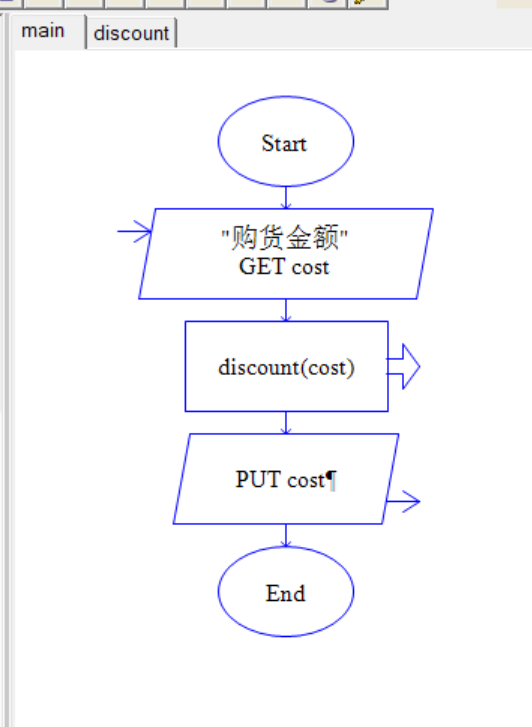
可以用if - else if - else来表示这种分段函数类问题。

Raptor算法流程图如下，

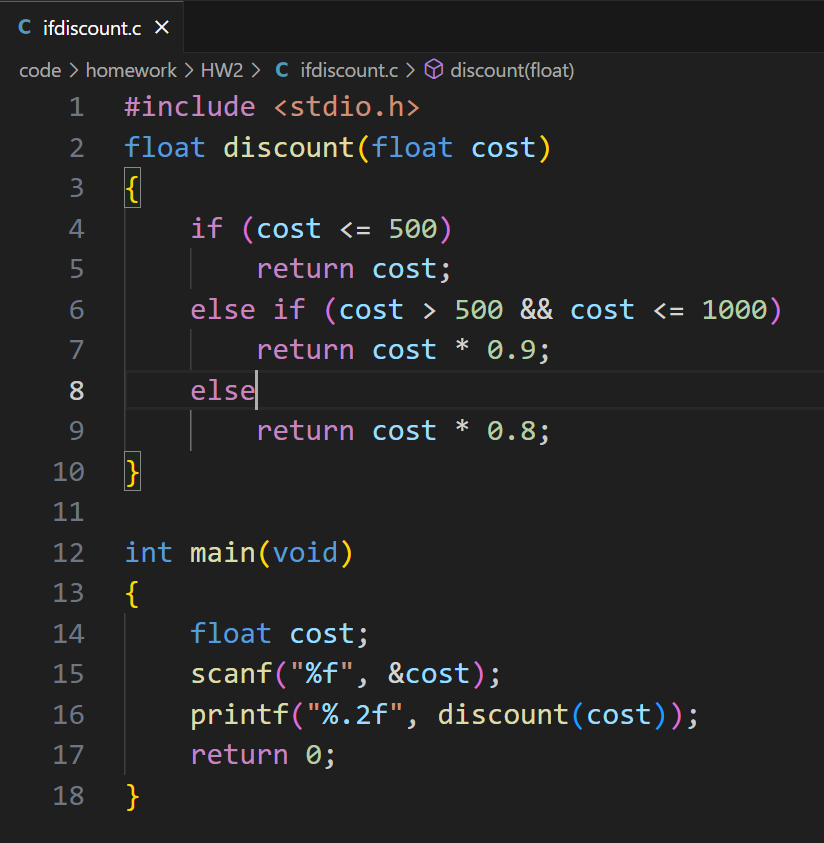
功能函数：

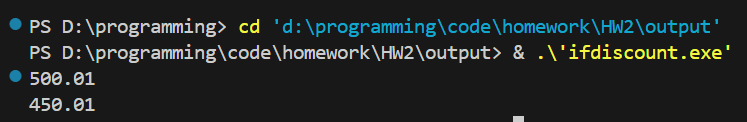


主函数：



实验代码、验证与结论：







总结与心得体会：

合理的缩进非常重要，尤其是涉及到复杂嵌套时，拥有正确的代码缩进可以快速看出其内在的层级结构等逻辑关系，便于阅读和修改。

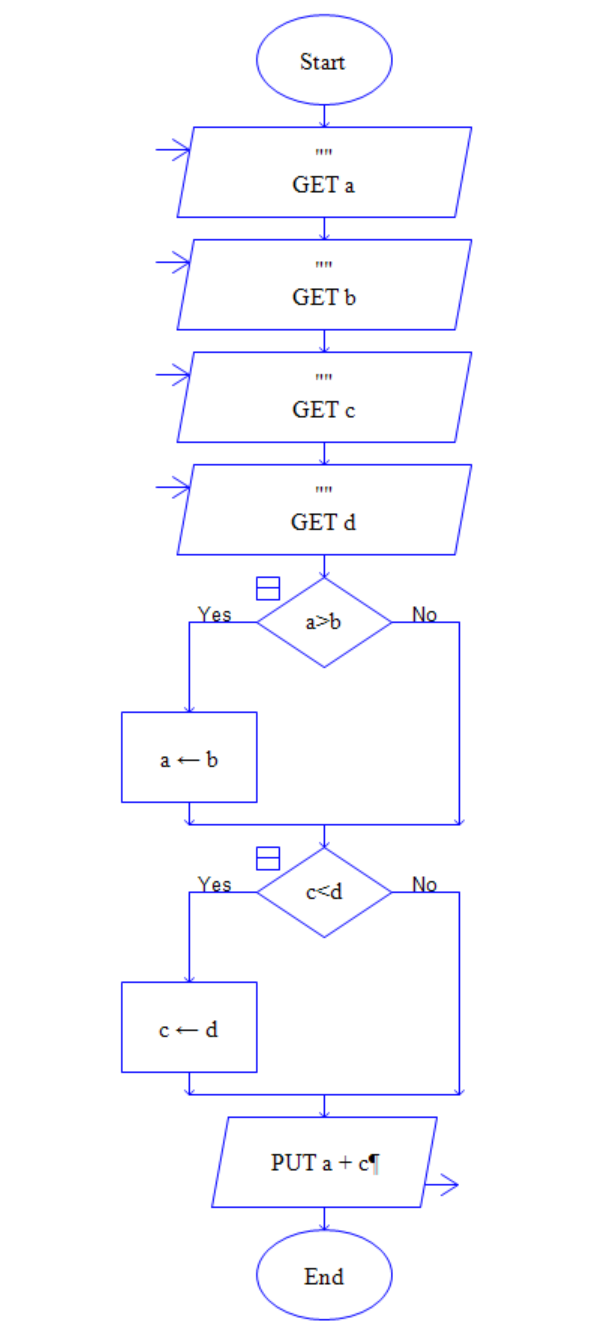
实验题目3： 比较大小并求和

输入4个整数，求其前两个数中较小的数与后两个数中较大的数之和。

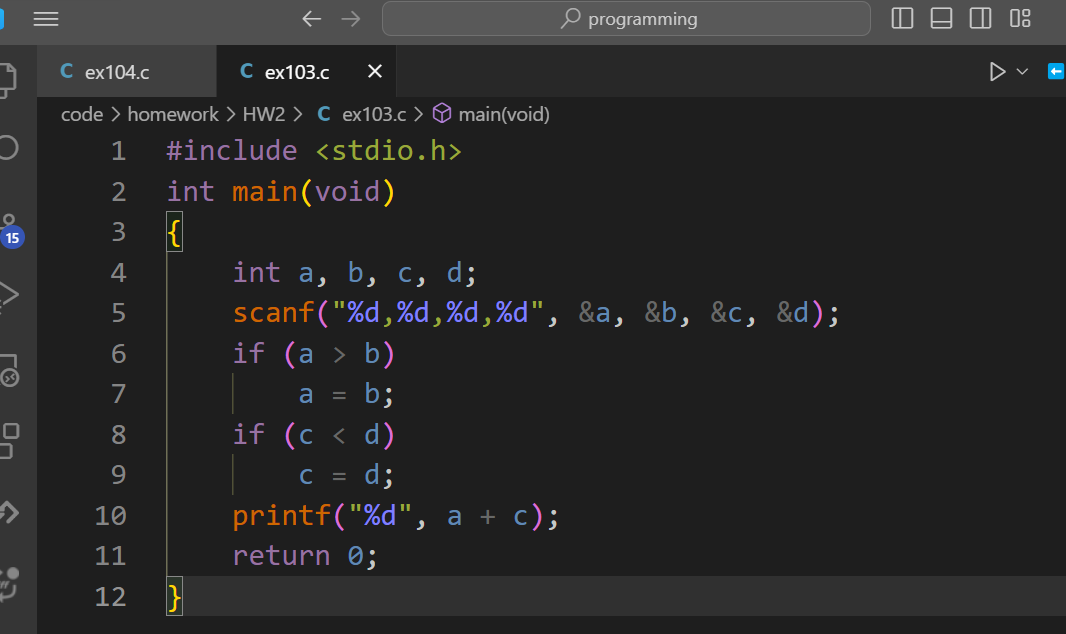
设计思想：

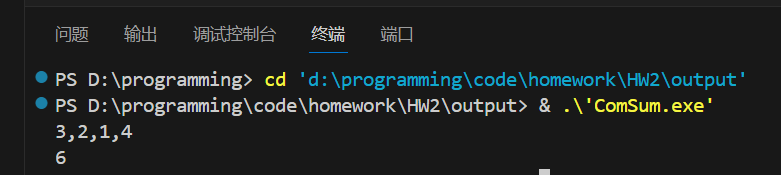
这道题可以先判断前两个变量谁大谁小，将小的数覆盖到大的数中，后面两个同理，将大的数覆盖到小的数中，最后只需要加第一个和第三个变量就可以了。

Raptor算法流程图如下：



实验代码、验证与结论：







总结与心得体会：

之前我一开始想到的思路是将使用if嵌套，但倘若真要这样写下去，估计代码会非常繁琐。而现在的这种思路灵感来源于我的室友，只需两个if语句，且不需要嵌套的形式，简洁易懂。

当然，这种方式覆盖掉了原本变量的值，倘若需要后续计算，还是采用交换变量更好。

实验题目4： 电费计算器

一个电表按照如下的比率计费：

前200度电：每度0.8元；

后100度电：每度0.9元；

超过300度电：每度1元。

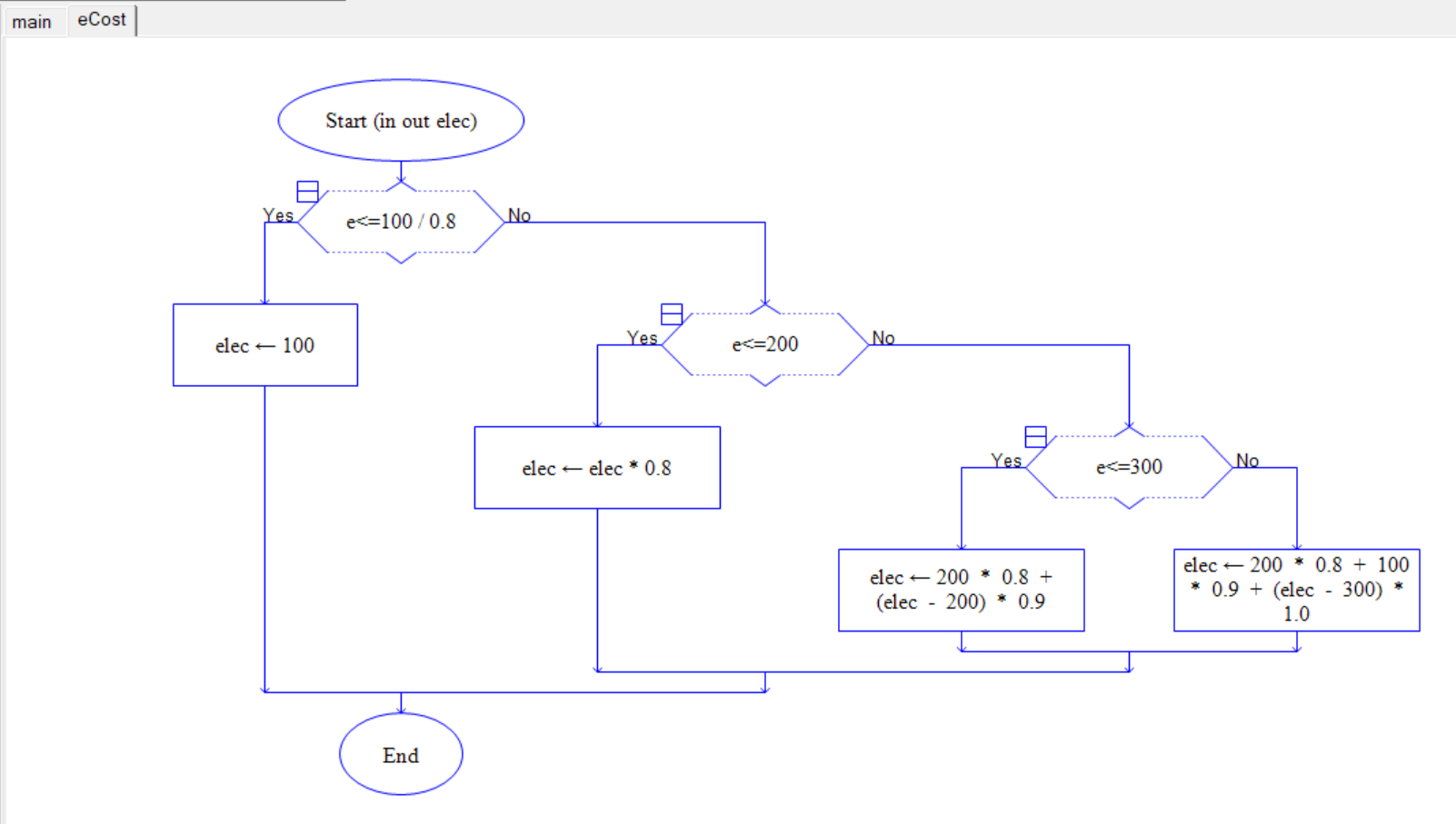
所有用户都是按最少100元进行收费。如果总费用大于400元，还要加收总数的15%的费用。请编写程序，从键盘输入用电量，经过计算后输出显示应收电费。

设计思想：

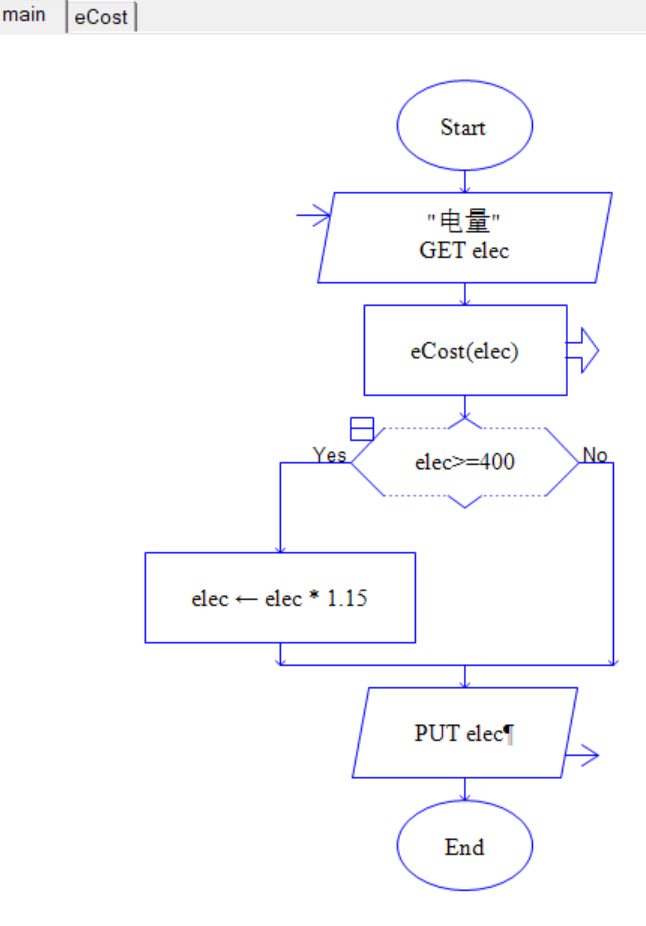
同样是分段函数，需要注意的是不要落下已知条件即可。

Raptor算法流程图如下，

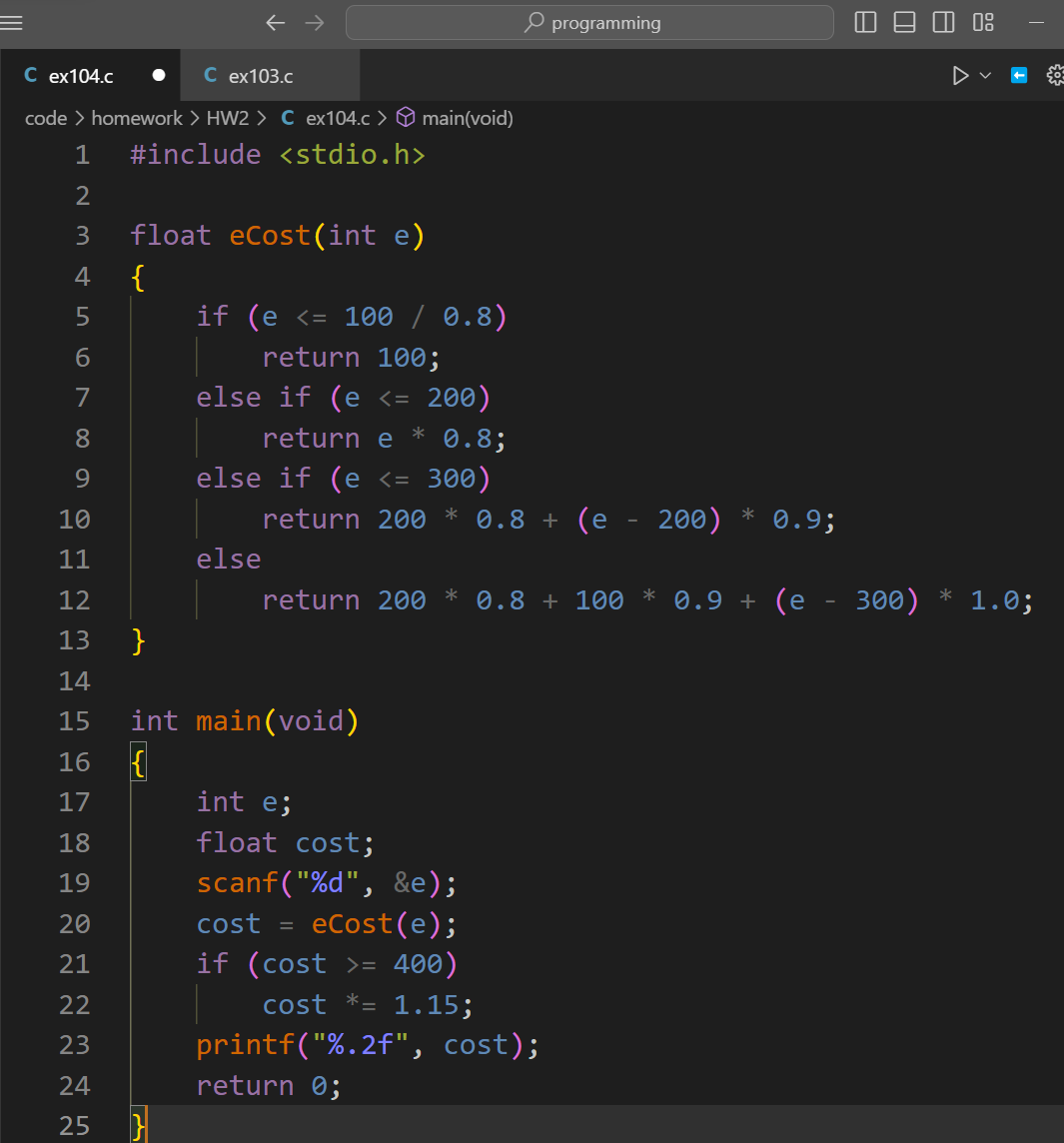
功能函数：

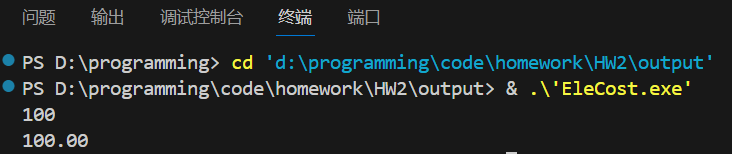


主函数：



实验代码、验证与结论：







总结与心得体会：

使用if – else -if 语句相当于树状的结构，只会进入一条分支处，所以在写条件时，我没有包括前面判断过的范围，因为符合前面的条件，就根本不会进入到后面的判断中，这一点可以从流程图中清晰的看出。

这次的程序不足之处是，没有把总费用大于400元，还要加收总数的15%的费用封装好，而是在主函数里进行的判断。

实验题目5： 数字转换单词

编写一个程序，要求用户输入一个两位数，然后输出显示该数的英文单词。提示：把数分解为两个数字。用一个switch语句显示第一位数字对应的单词（“twenty”、“thirty”等），用第二个switch语句显示第二位数字对应的单词。不要忘记11~19需要特殊处理。

设计思想：

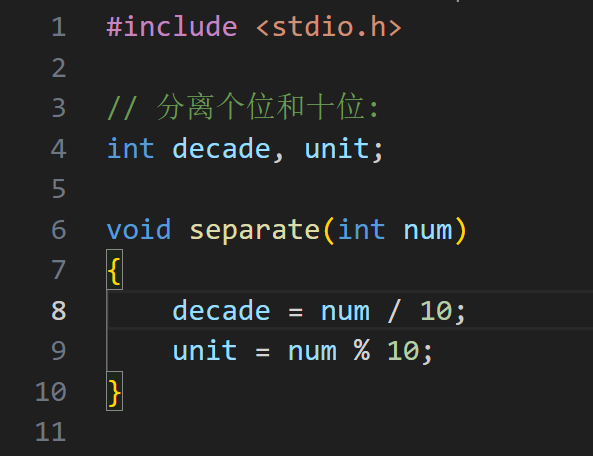
这道题既然要求使用switch语句，并且思路不算复杂，只是有点繁琐，这里我就用文字来代替流程图了。

题目中需要把11~19单独处理，所以先用if语句判断是否处于11~19，然后特殊处理即可。至于20~99，我们直接将个位与十位分离，然后单独转换即可。

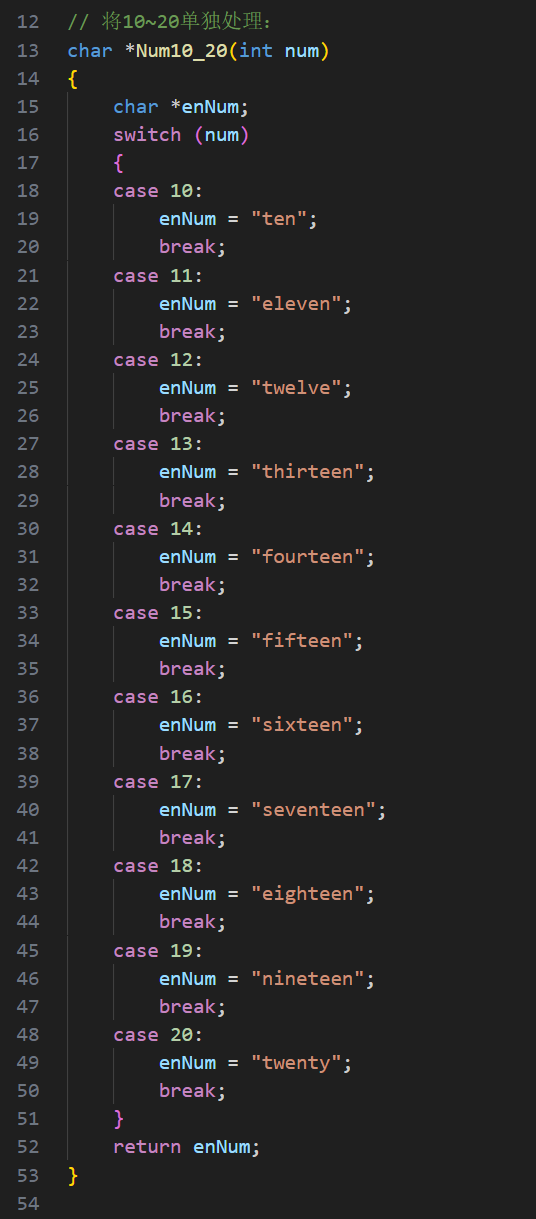
不要忘记还有题目中没有强调的10，也是两位数。

实验代码、验证与结论：

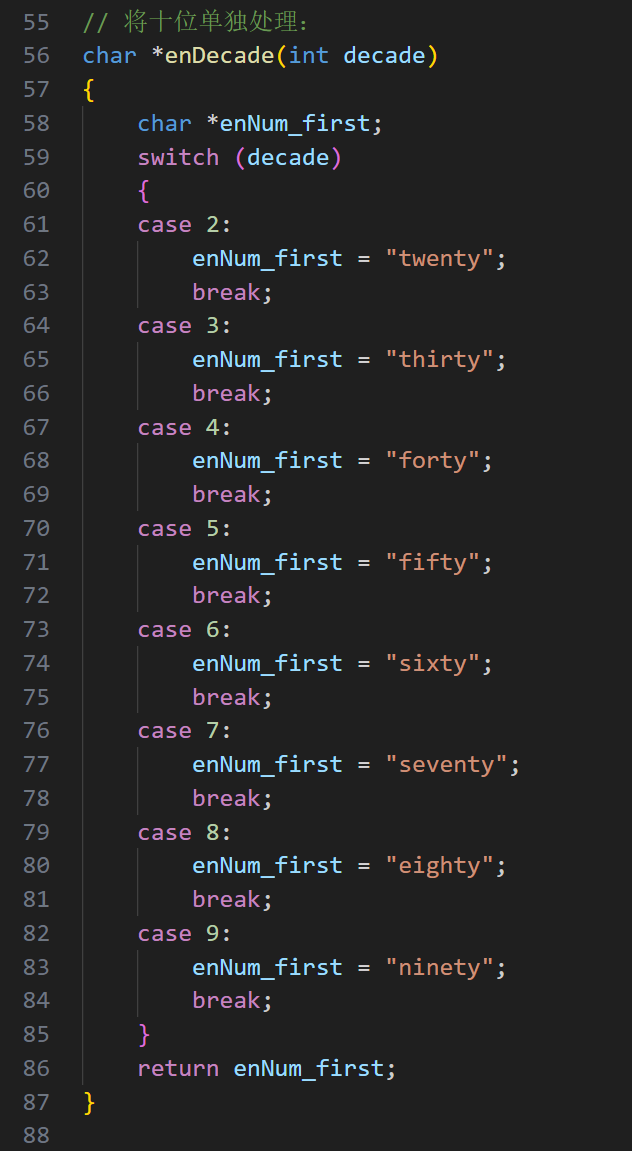
引入头文件，创建全局变量，创建分离个位和十位的函数。



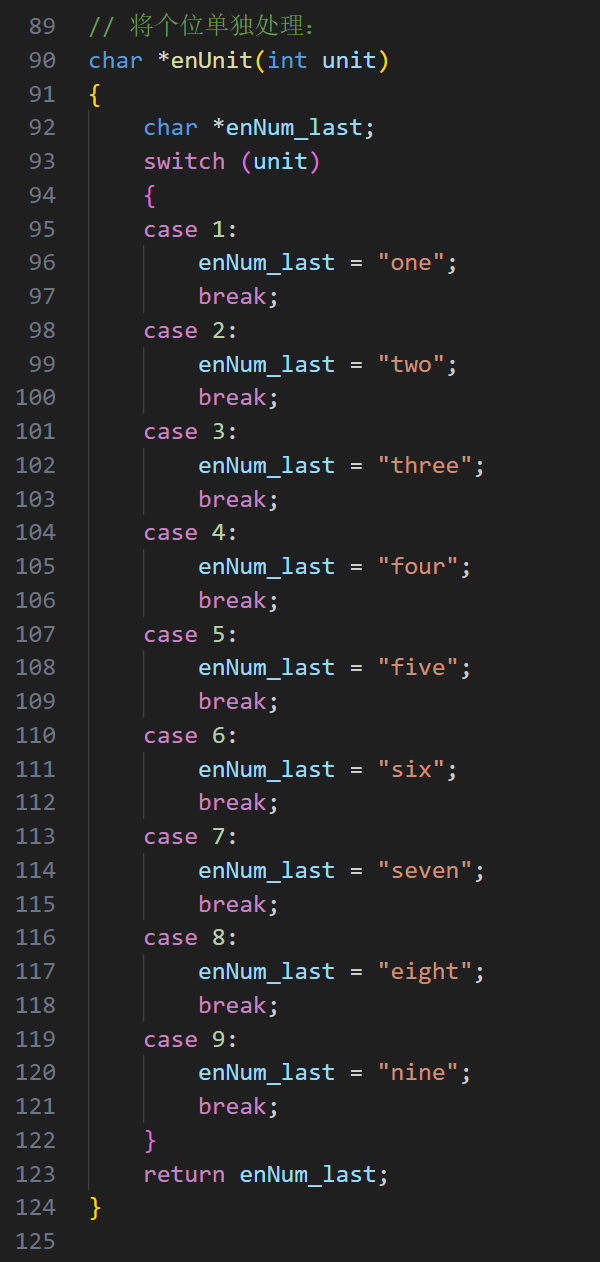
将10~20单独处理：



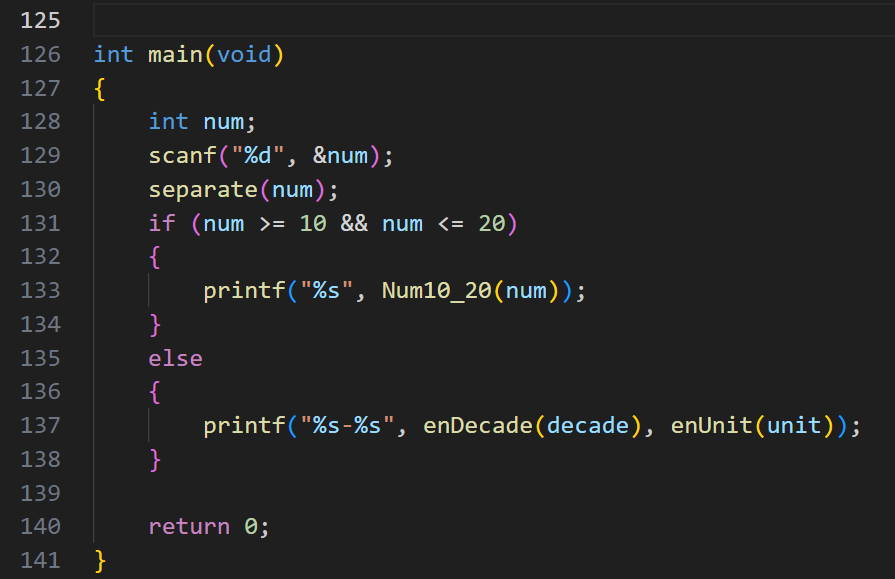
将十位单独处理：



将个位单独处理：



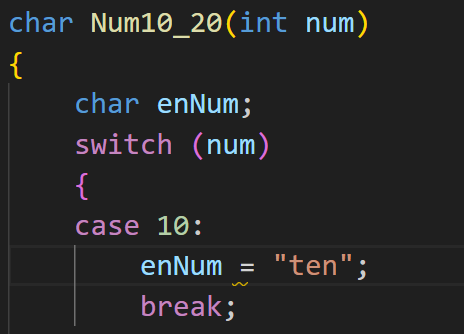
主函数：



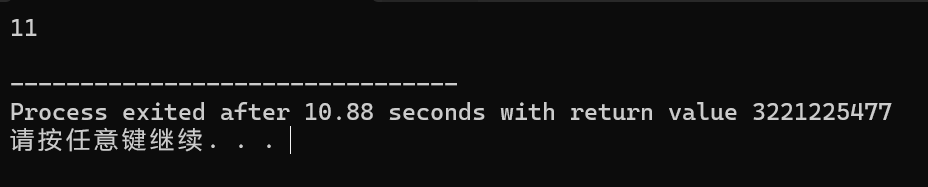


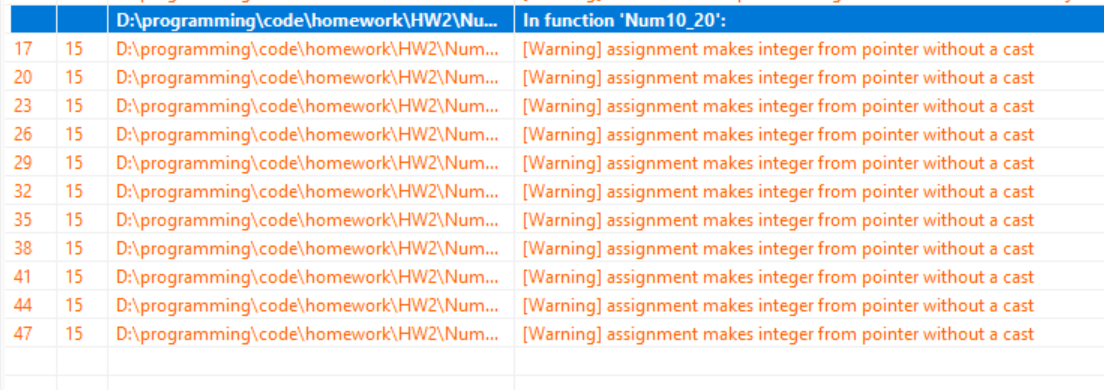
总结与心得体会：

在创建字符型返回值的函数时，我一开始是这样写的：



发现：

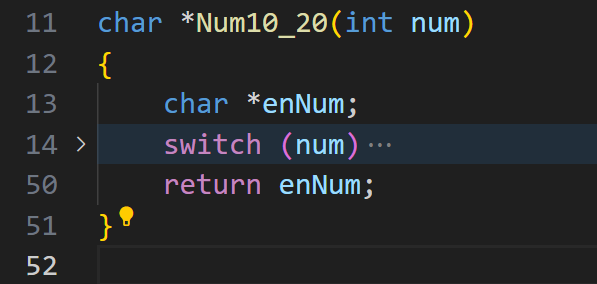




首先返回值为3221225477，上网上搜索资料发现，是因为空指针异常，推测可能是访问了局部变量。

其次是警告，发现我定义的enNum变量是一个字符类型的变量，而我存进去的是字符串。搜索发现，在C语言中没有专门的字符串类型，只能使用数组或者指针来间接地存储字符串。

尝试修改，将enNum变量修改为char \* 类型，由于返回的是指针类型，所以函数的类型也应修改为char \*。



这样就可以返回字符串类型了。

实验三：循环结构

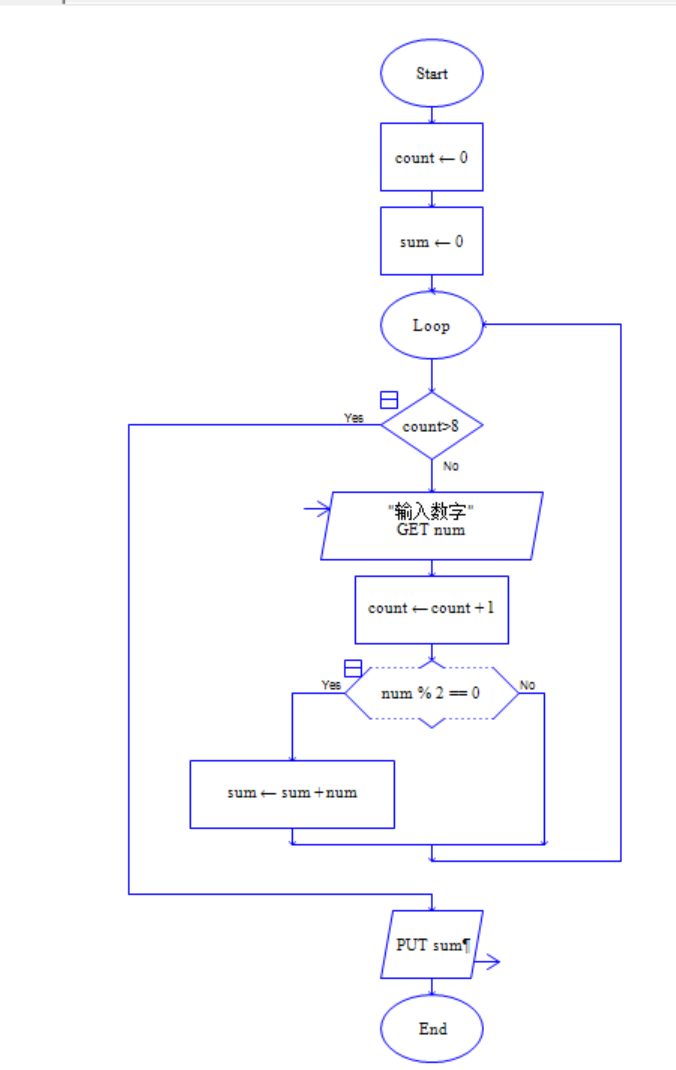
实验题目1：偶数之和

输入8个整数，求其中所有偶数的和。

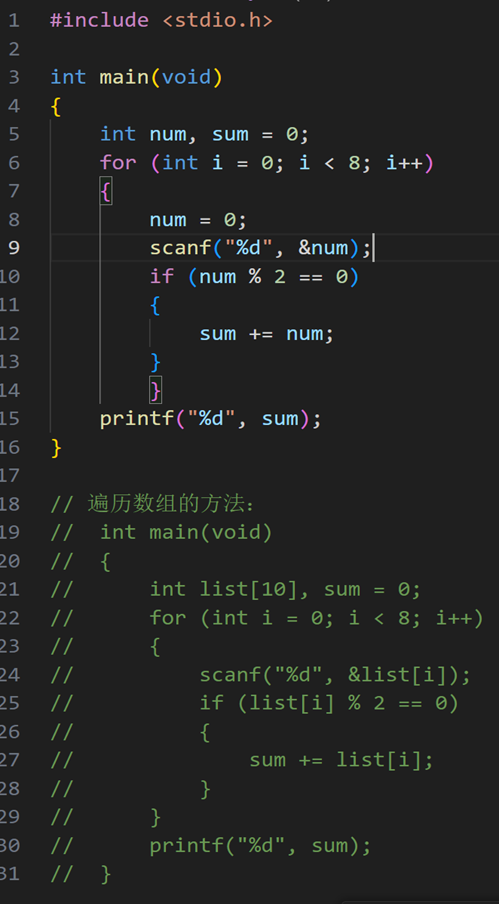
设计思想：

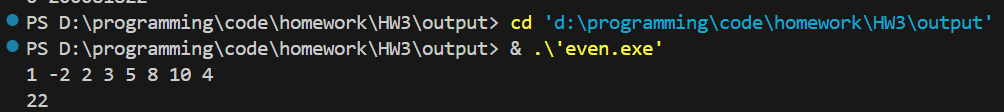
偶数，就是除以2余数为0的数。

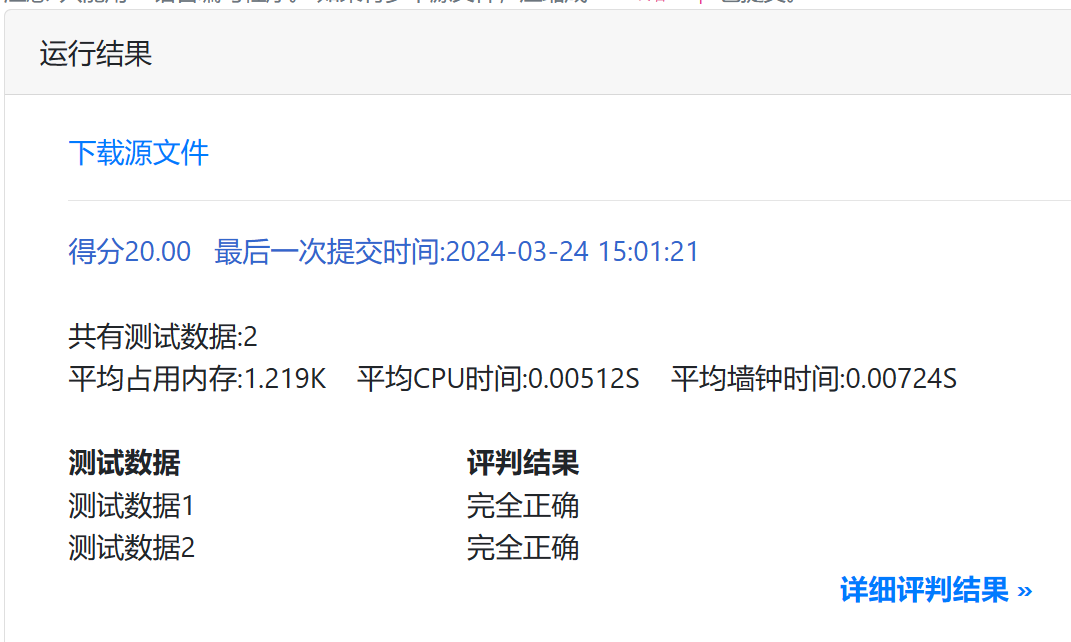
Raptor算法流程图如下：



实验代码、验证与结论：







总结与心得体会：

一开始看到题目输入多组数据总是往数组那方面去想，后来发现不用数组也可以的，反而比数组还方便。

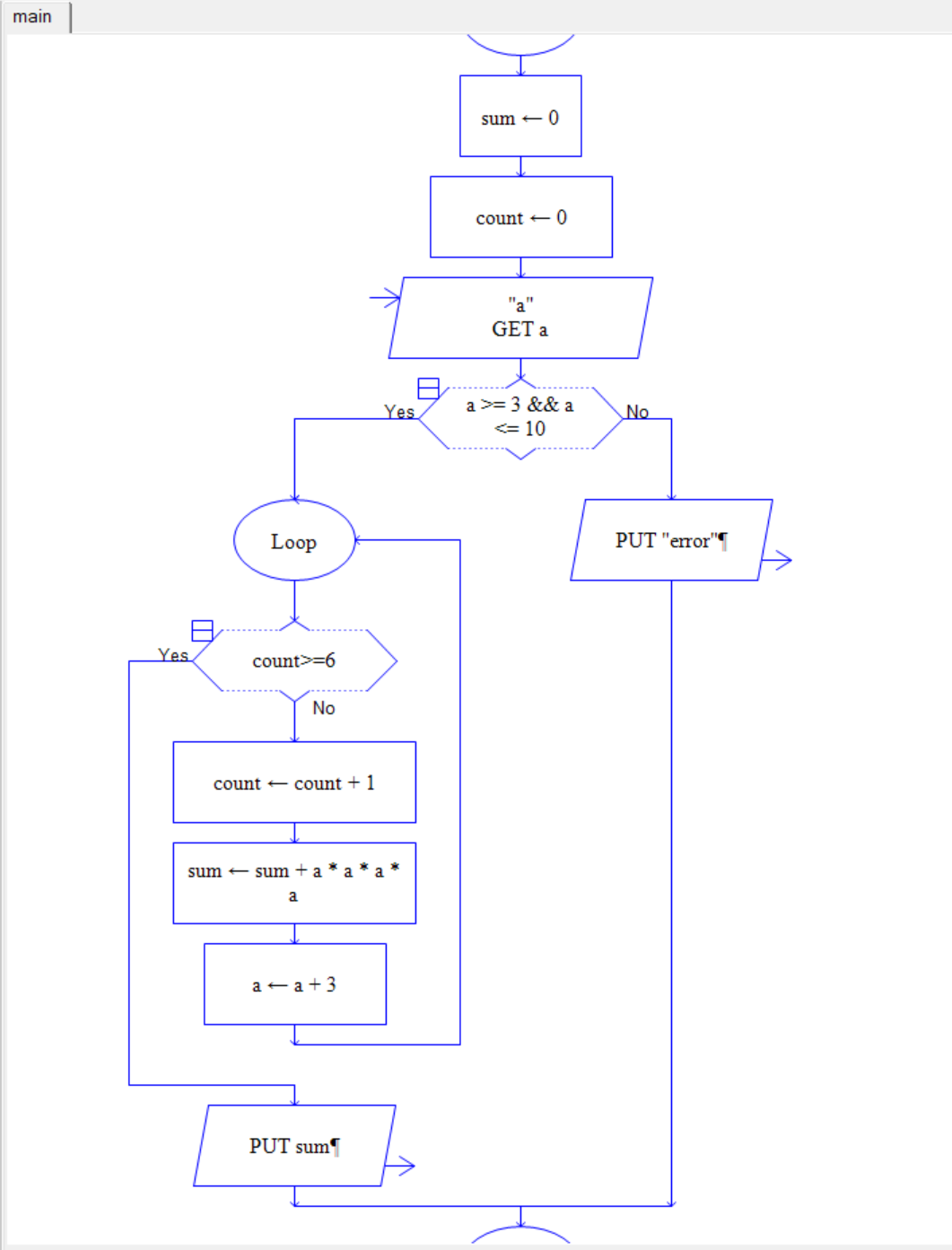
实验题目2：计算乘方和

从键盘为整型变量a赋值，如果a的值为闭区间[3,10]之间的整数，则计算并输出a4+(a+3)4+(a+6)4+(a+9)4+(a+12)4+(a+15)4的值；否则输出“error”。

设计思想：

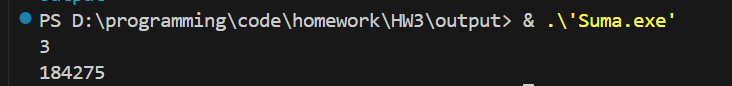
发现公式太长，且都是括号内值的4次方相加，可以使用循环来简化处理，只需在每次循环时将括号内的值加4即可。

Raptor算法流程图如下：



实验代码、验证与结论：







总结与心得体会：

编程中结构相似的语句，可以通过循环、函数等来进行优化；生活中的一些单一繁琐的工作与计算，同样也可以交给计算机轻松解决。

实验题目3： 猴子吃桃问题

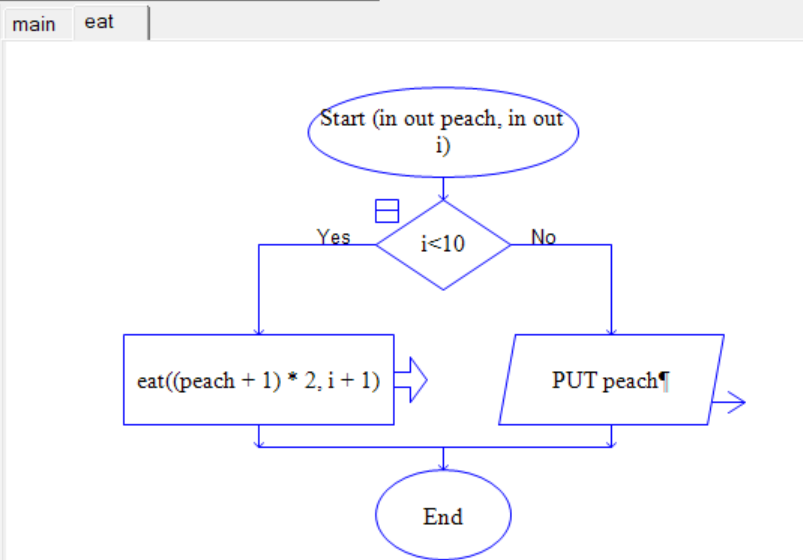
猴子第一天摘下若干个桃子，当即就吃了一半，还不过瘾，又多吃了一个。第二天早上又将剩下的桃子吃掉一半，又多吃一个。以后每天早上都吃了前一天剩下的一半多一个。到第10天早上想再吃时，发现只剩下一个桃子了。求第一天共摘多少个桃子？

设计思想：

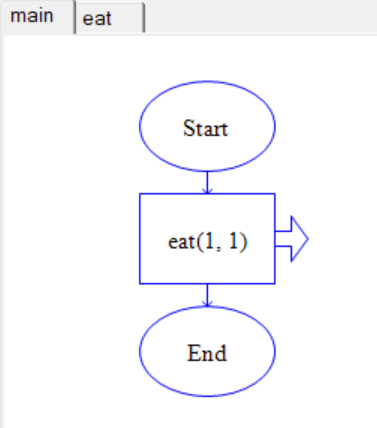
听说这道题好像还是小学奥数题，反着推就可以了。10天，如果写数学运算式会非常长，但运算式的结构相似，可以用循环或是递归来解决。

Raptor算法流程图如下，

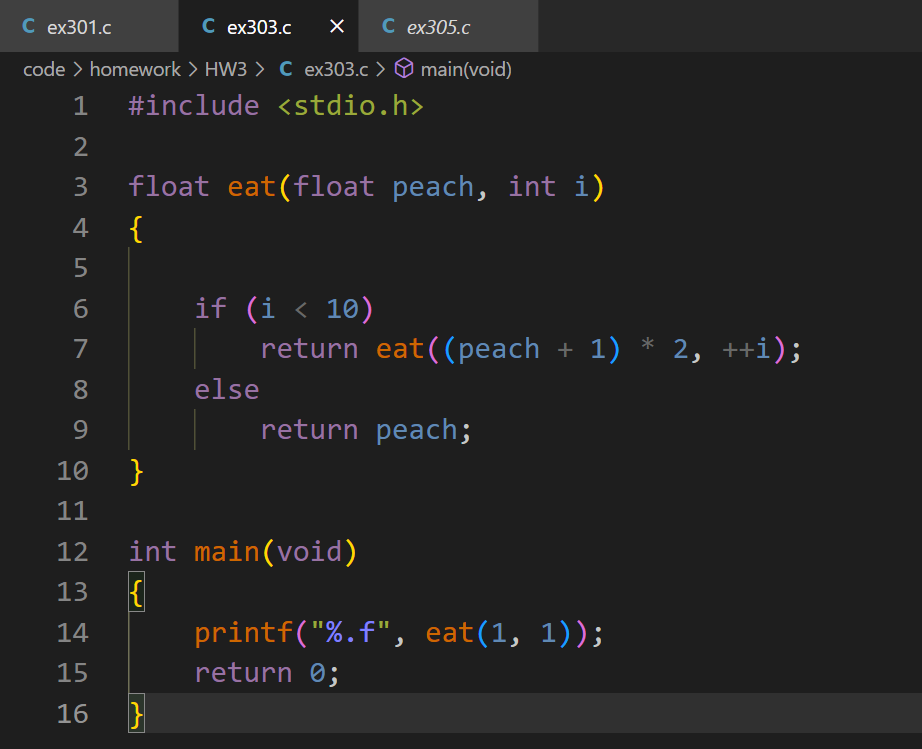
功能函数：

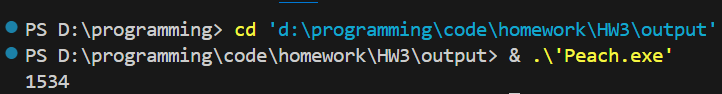


主函数：



实验代码、验证与结论：







总结与心得体会：

与上一道题类似，可以采用循环、递归等思路简化数学公式。

实验题目4：寻找完数

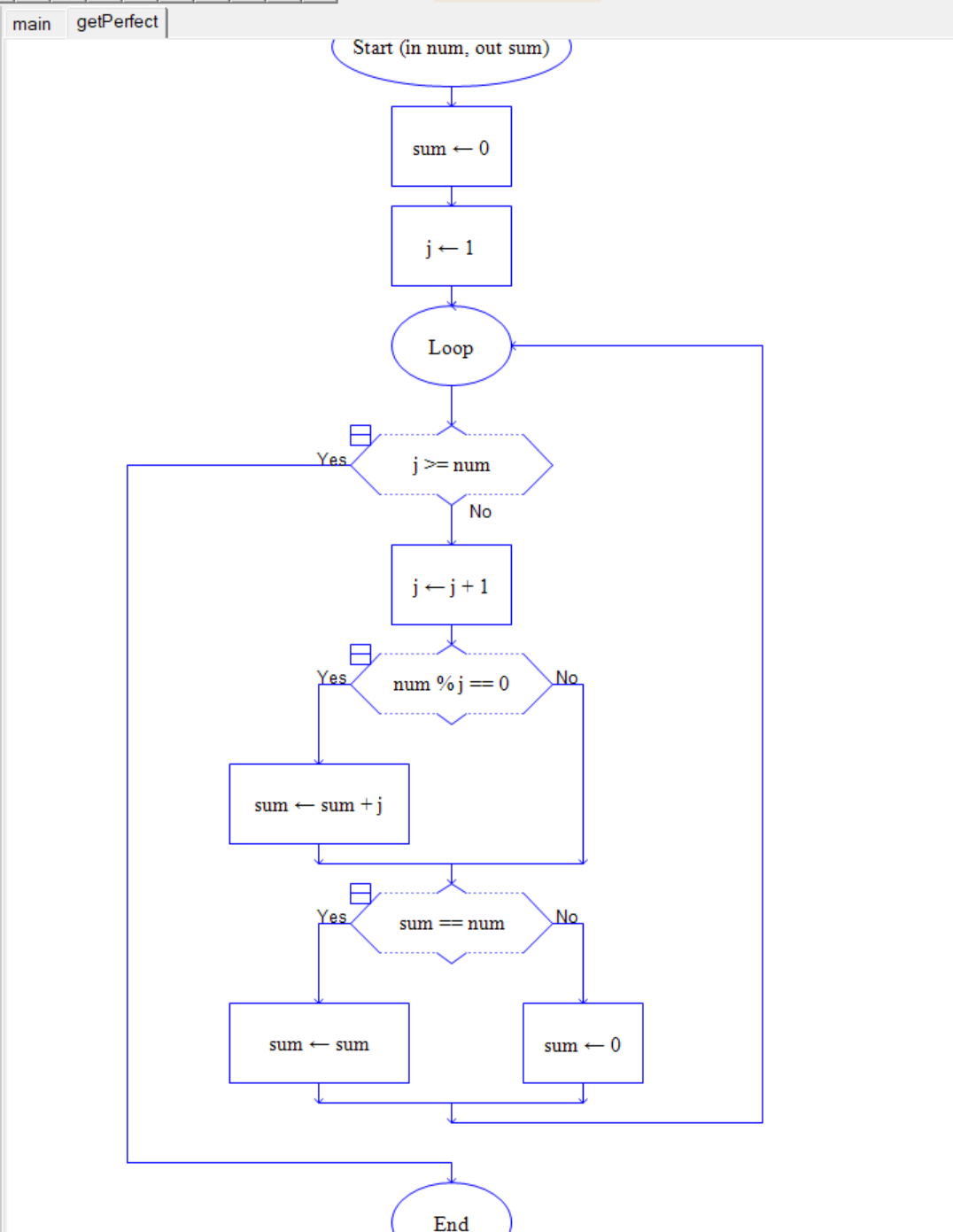
完数是指一个整数恰好等于它的因子之和（除自身外），则称这个数为完数。从键盘先后输入两个不大于9999的正整数m和n，若m>n，则交换两数。然后求m~n（m和n均为正整数且m≤n）之间的所有完数。

设计思想：

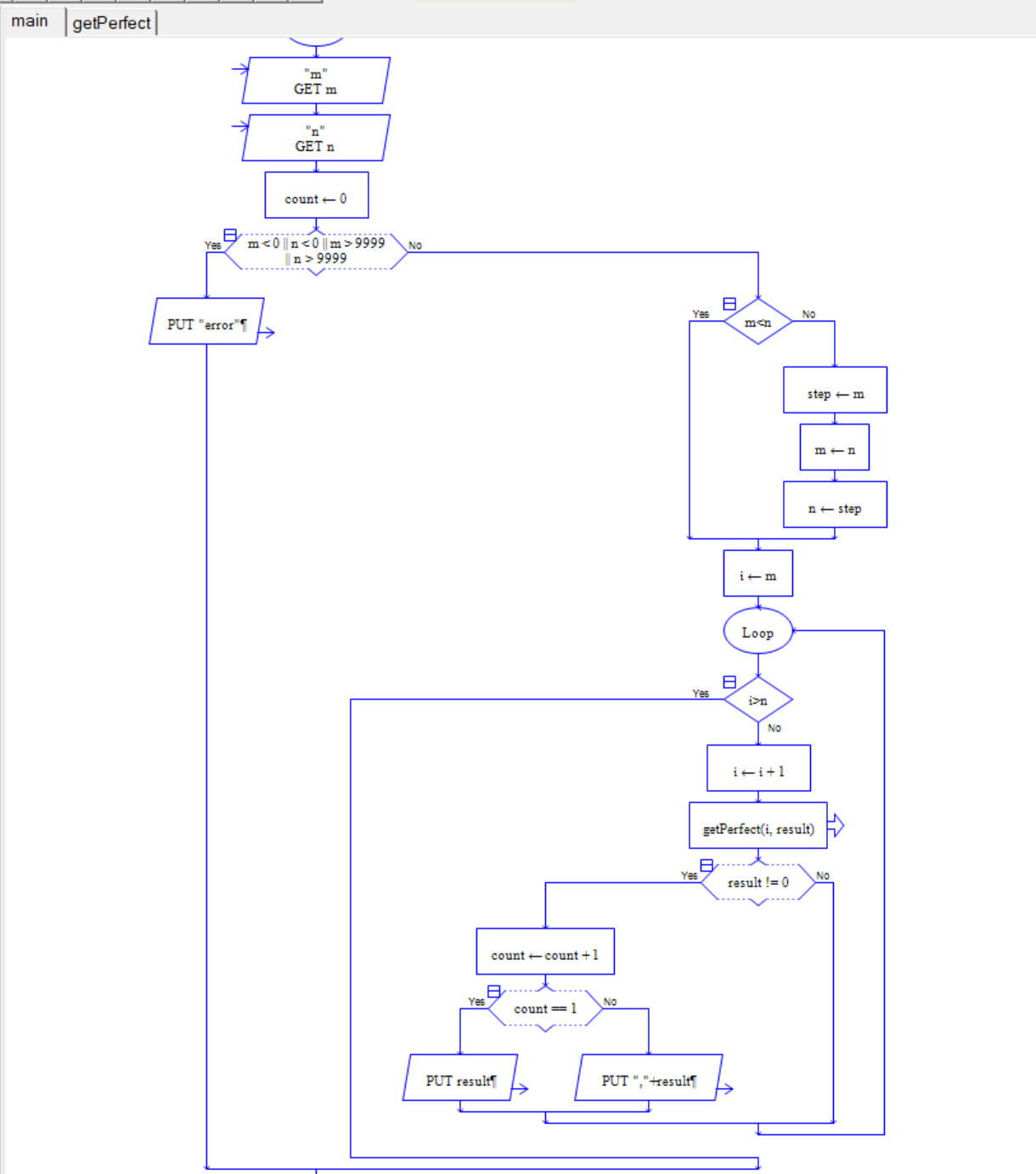
因子还是用到相除余数为0的判断，可以使用for循环将所求区间里的数逐一提取出来。

Raptor代码如下，

功能函数：

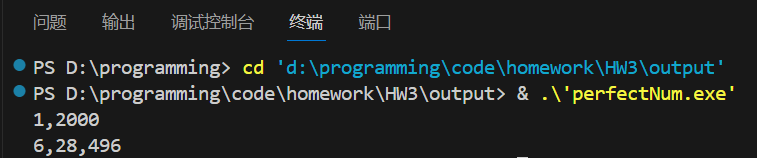


主函数：



实验代码、验证与结论：







总结与心得体会：

做这道题时，面对输出结果要用逗号分隔的要求时，我一开始有点不知道从何入手，后来看到群里面何厚霖同学的题解时才恍然大悟。

这道题我在判断非法输入时还想多了，推测评测机可不可能只输入一个数字？或是输入了3个甚至更多？搜索了解到scanf函数的返回值可以判断输入值的个数，多输入的东西我打算转为字符串，然后判断它的长度是否为零……不过后来看了群里大家的题解，才发现自己想多了。

这道题如果先交换变量，后判断非法输入，会更好一些，只需要m < 0和n > 9999两个逻辑表达式即可，就不需要写四个了。

实验题目5： 水仙花数

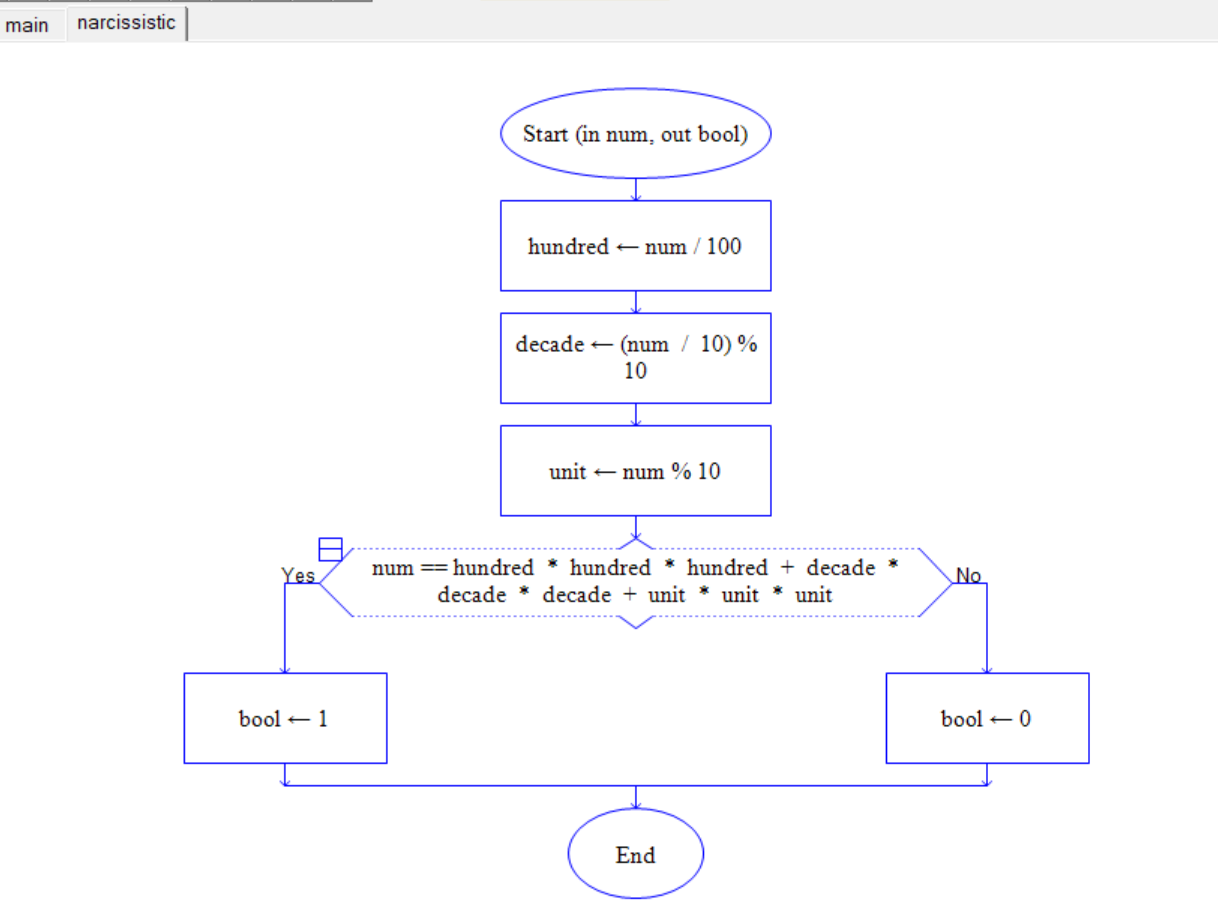
打印出所有“水仙花数”，所谓“水仙花数”是指一个三位数，其各位数字立方和等于该数本身。例如：153是一个水仙花数，因为153=13+53+33。

设计思想：

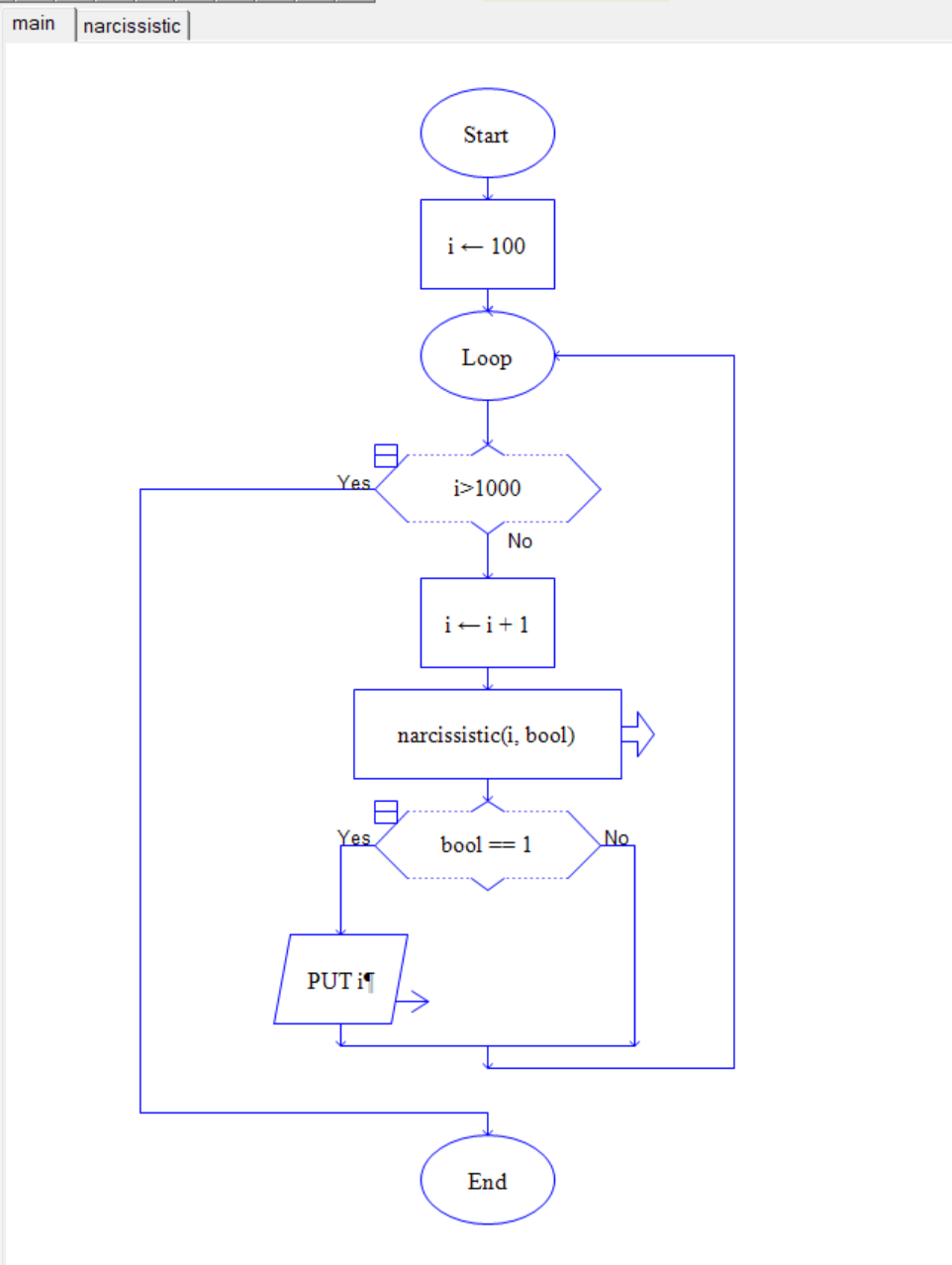
这道题的核心是将个、十、百位分开，需要“/”和“%”两种预算符共同使用。

Raptor算法流程图如下，

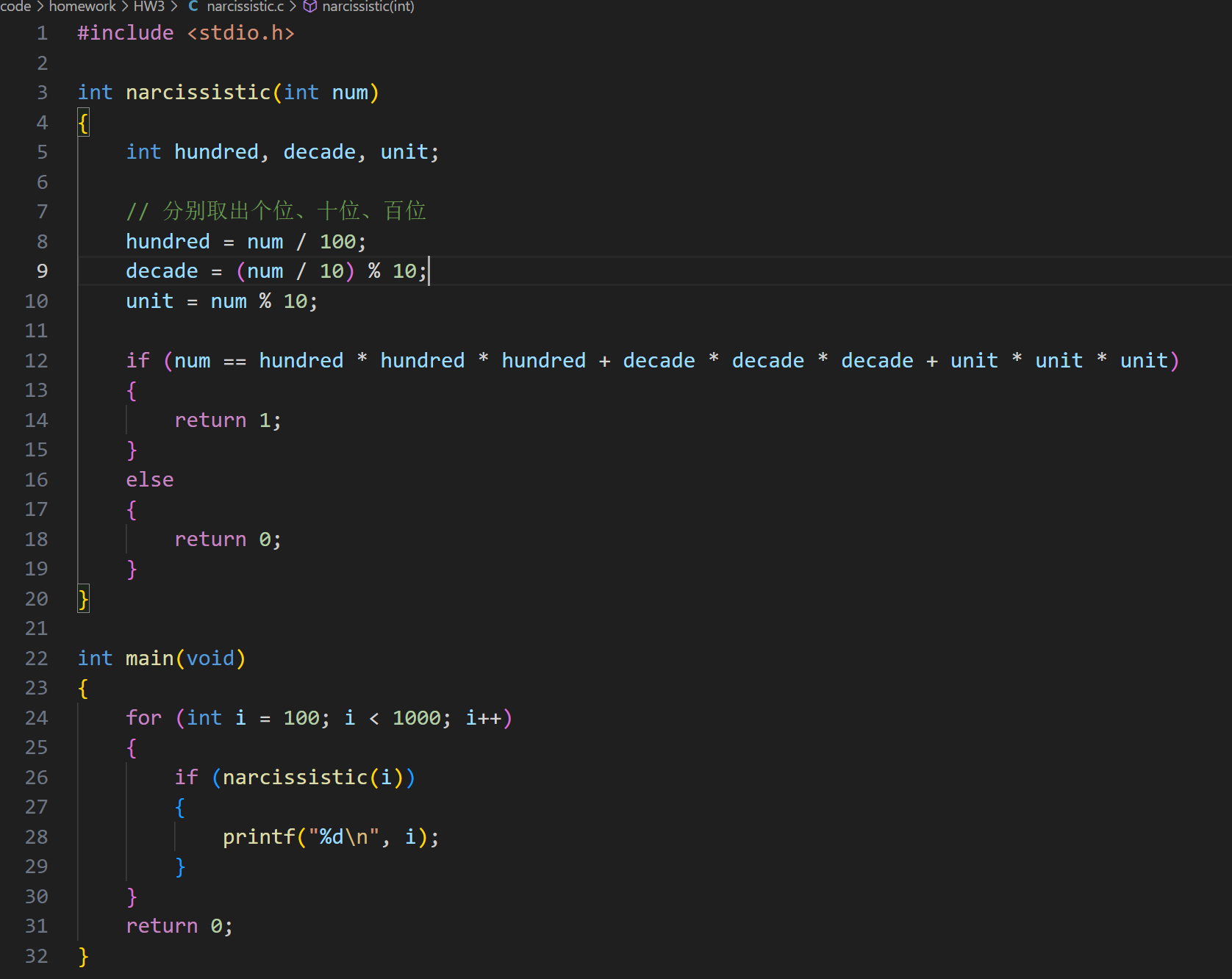
功能函数：

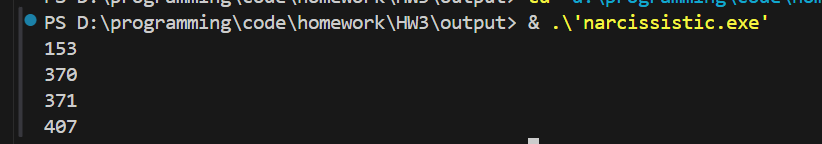


主函数：



实验代码、验证与结论：







总结与心得体会：

这道题的核心就是分离三位数，同时需要注意的是，题目要求是每个输出都要换行，要认真审题。

实验四：函数

实验题目1：数列前n项和

写一个函数，求解以下数列前n项之和并输出，其中n（正整数）应从键盘输入。

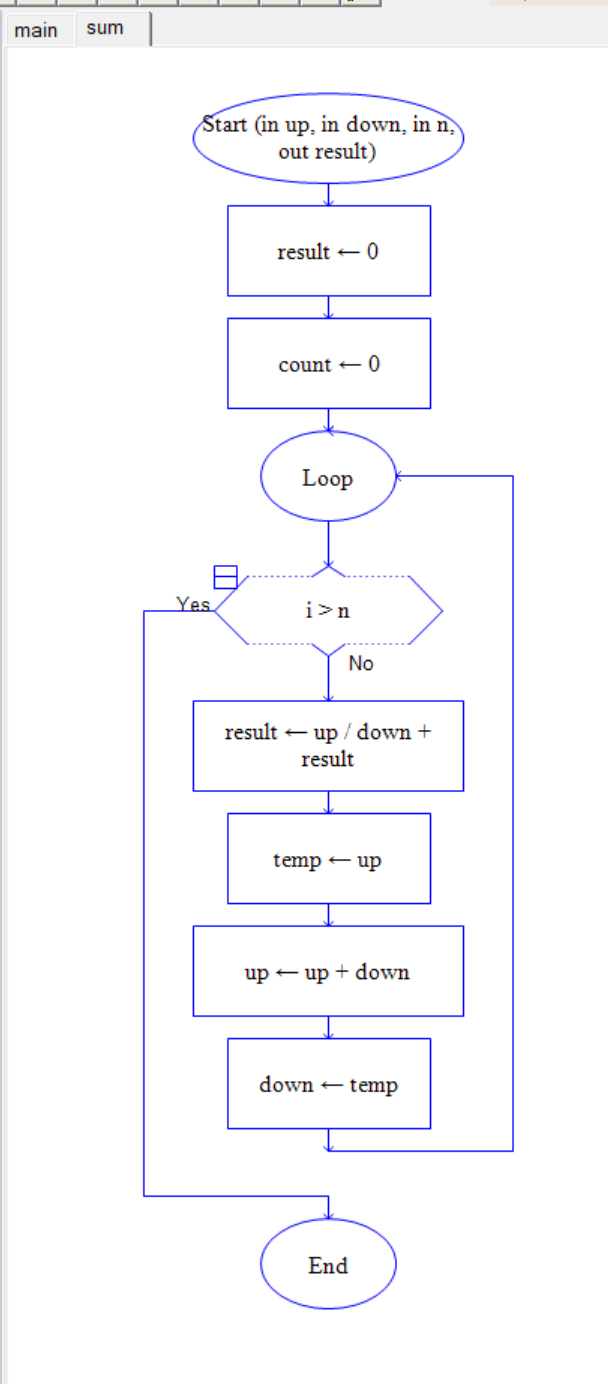
设计思想：

观察数列发现，下一项的分子是上一项的分子与分母之和，上一项的分母为上一项的分子。

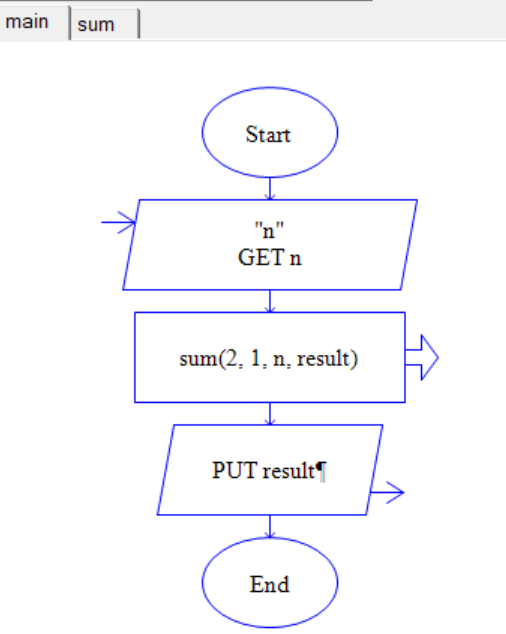
那就可以将分子与分母设为两个变量然后进行上述运算就可以啦。

Raptor算法流程图如下，

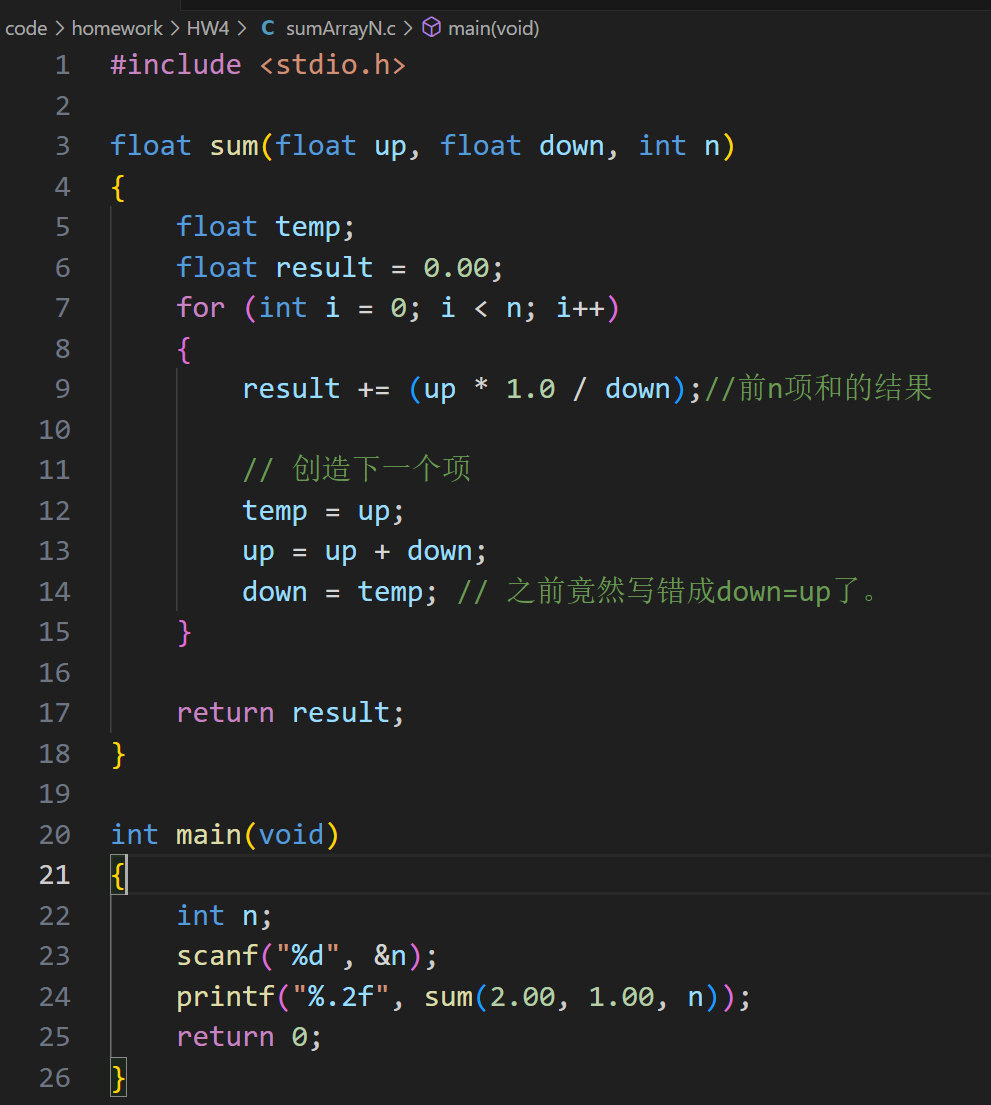
功能函数：

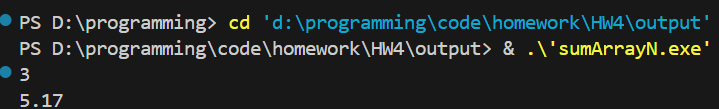


主函数：



实验代码、验证与结论：

**





总结与心得体会：

这道题我之前犯的错误是，我在前一行更改了分子的值，但我没有意识到！而是把我认为没改过的值赋给了分母，导致出错。

在覆盖变量值的时候，一定要想一想这个原先的值后面会不会用到。

实验题目2：寻找双质数

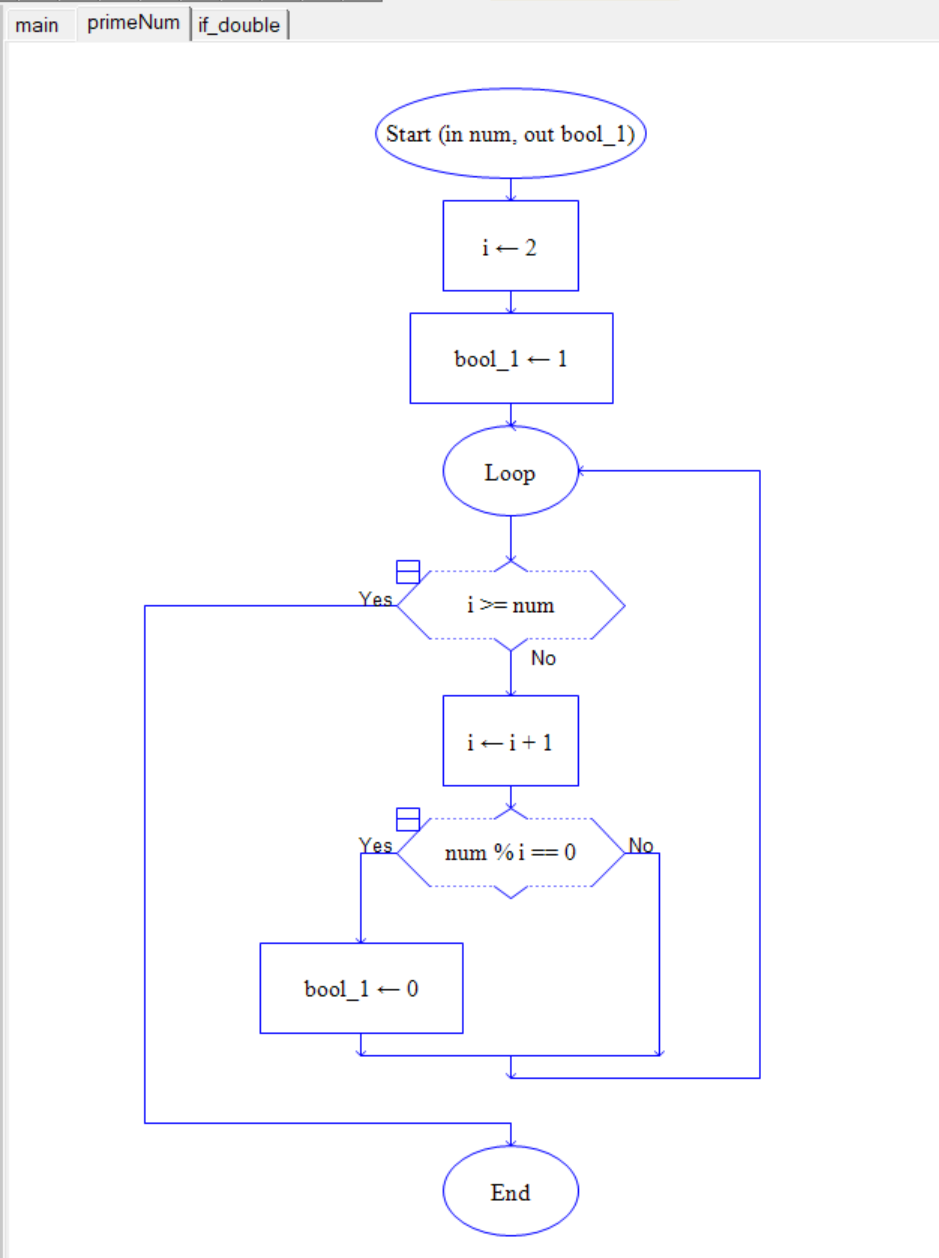
所谓“双质数”是指对于两个质数p和q，如果满足p = q + 2，则p和q为双质数。请编程实现利用函数求解闭区间[m, n]之间的双质数，其中m和n均为正整数且m < n。

设计思想：

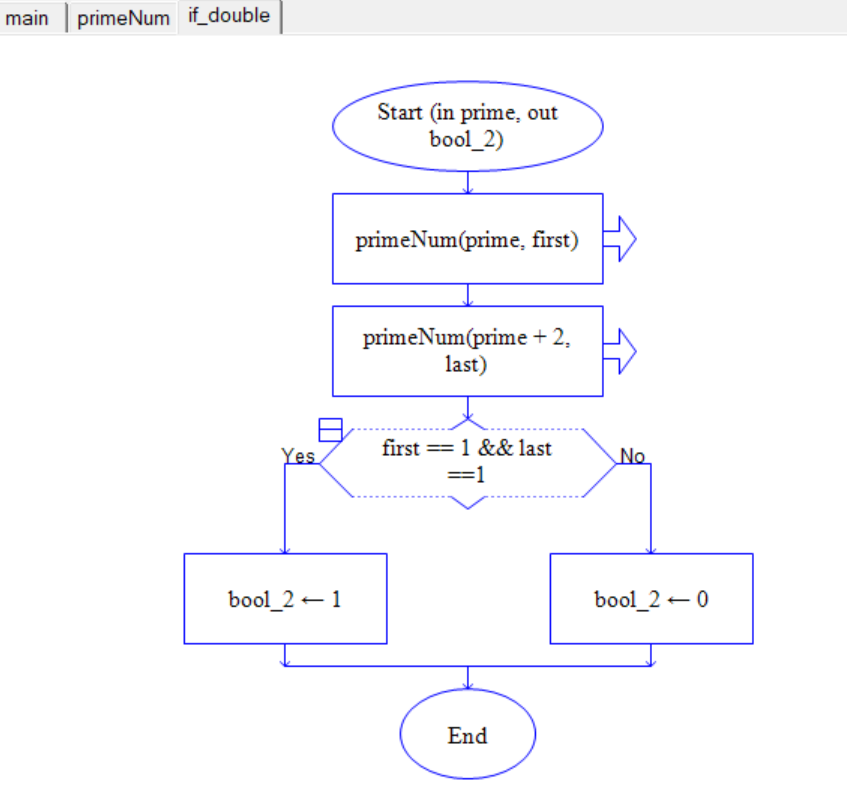
首先要判断它是质数，在判断这个数加2是否是质数，最后还要判断这个数加2是否在范围里。

判断质数还是用余数的方法，Raptor代码如下，

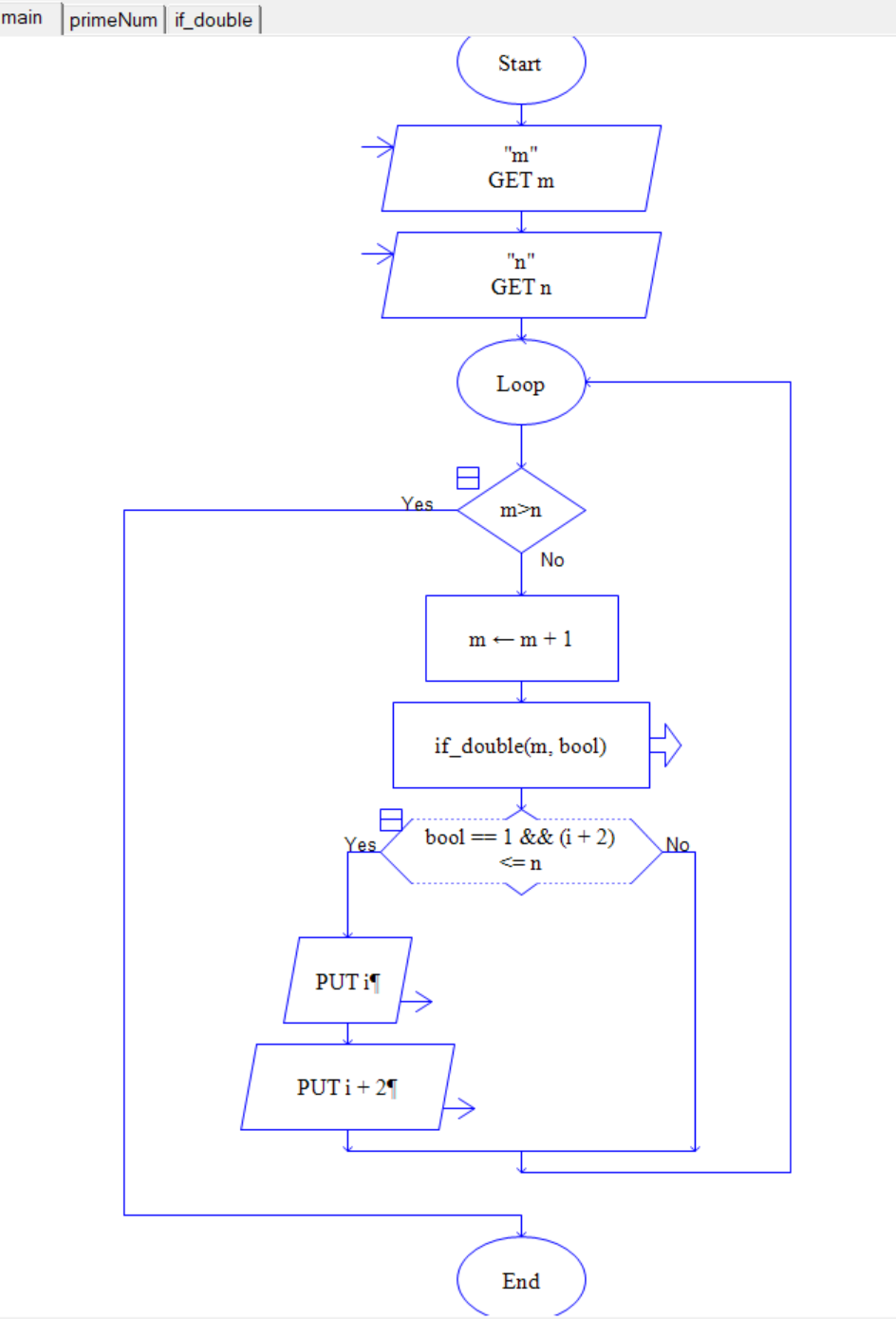
primeNum函数：



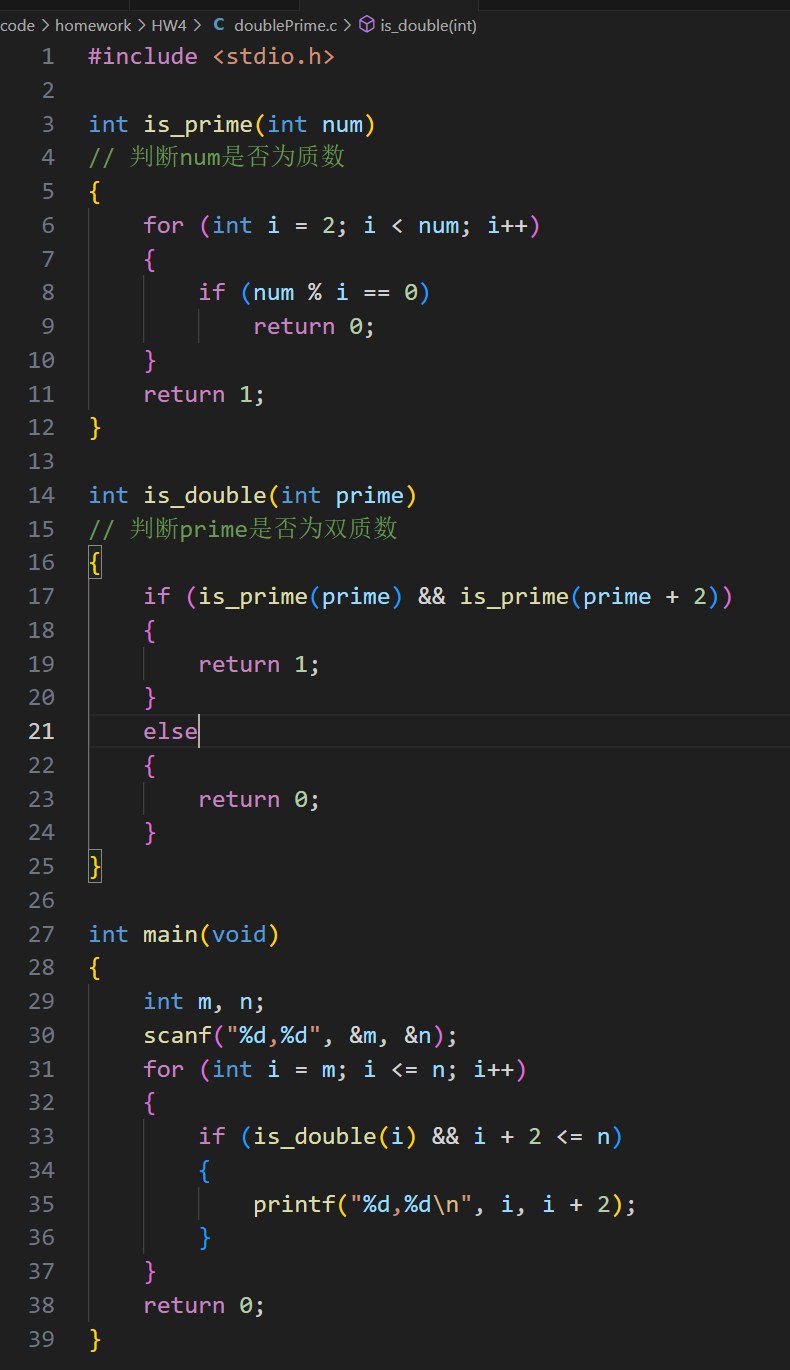
if\_double函数：

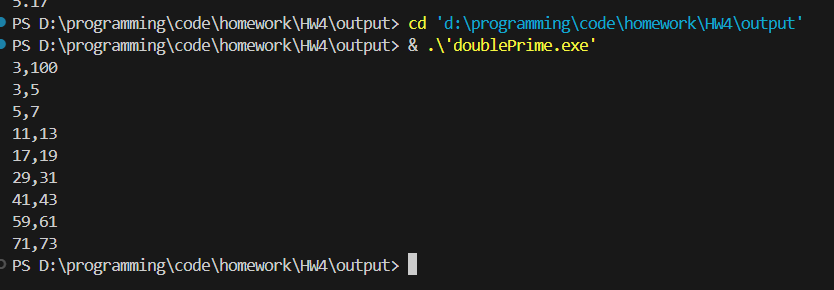


主函数：



实验代码、验证与结论：







总结与心得体会：

在if语句内的逻辑表达式中，0会被转化为False，其余的数会转化为True。

再给函数命名时，我原先的命名是primeNum函数和if\_double函数(从流程图上可以看出)，后来我觉得我的这个命名并不规范，函数的命名应改为动宾结构的短语，改为is\_prime和is\_double会更易懂。因为函数function本身就是功能的意思，应该代表一个功能、一种方法，所以动宾短语会更合适一些。

同时，函数应该加上一些注释以说明其功能。

这道题我的室友有另外一种思路，他是把所有的质数都找出来了，然后判断一个数和比它大二的数是否在这些质数里，而非我这种返回0 或1 的单一判断。

可是这样做的话，应该只能使用数组了吧？

并且他把for循环写在了函数里，每找出一个指数，就return一个值。

可是，函数一旦执行了return语句，应该就会自动退出这个函数了，剩下的质数还是无法找出。

实验题目3：寻找回文数

所谓回文数就是将一个数从左向右读与从右向左读是一样的，例如，121和1331都是回文数。编写一个函数实现求正整数n以内的回文数。

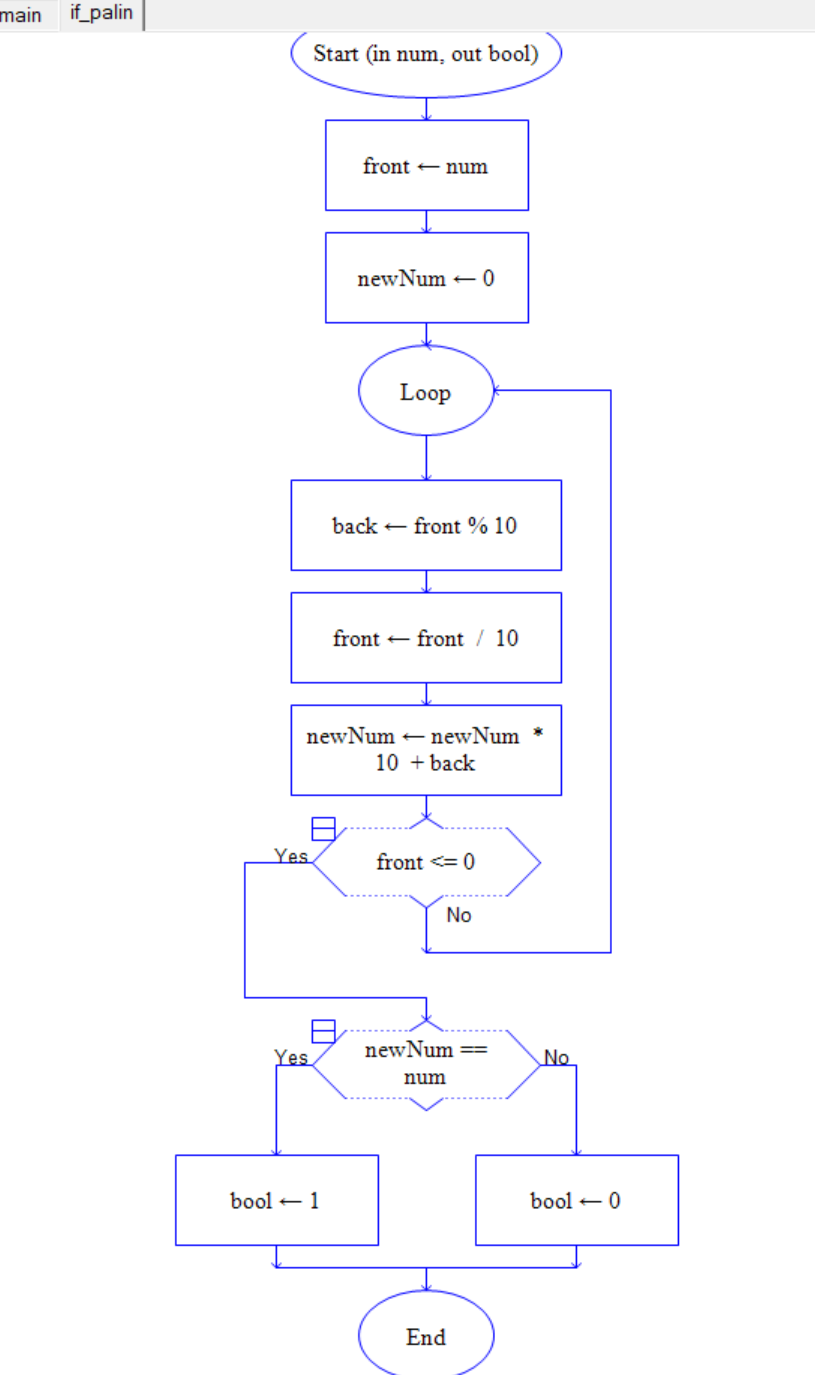
设计思想：

提到回文，我一开始想到的就是两头循环遍历字符串。但是我在课上做到这道题的时候，老师突然强调不可以使用后面没学过的方法，那么我只能令寻他法了。

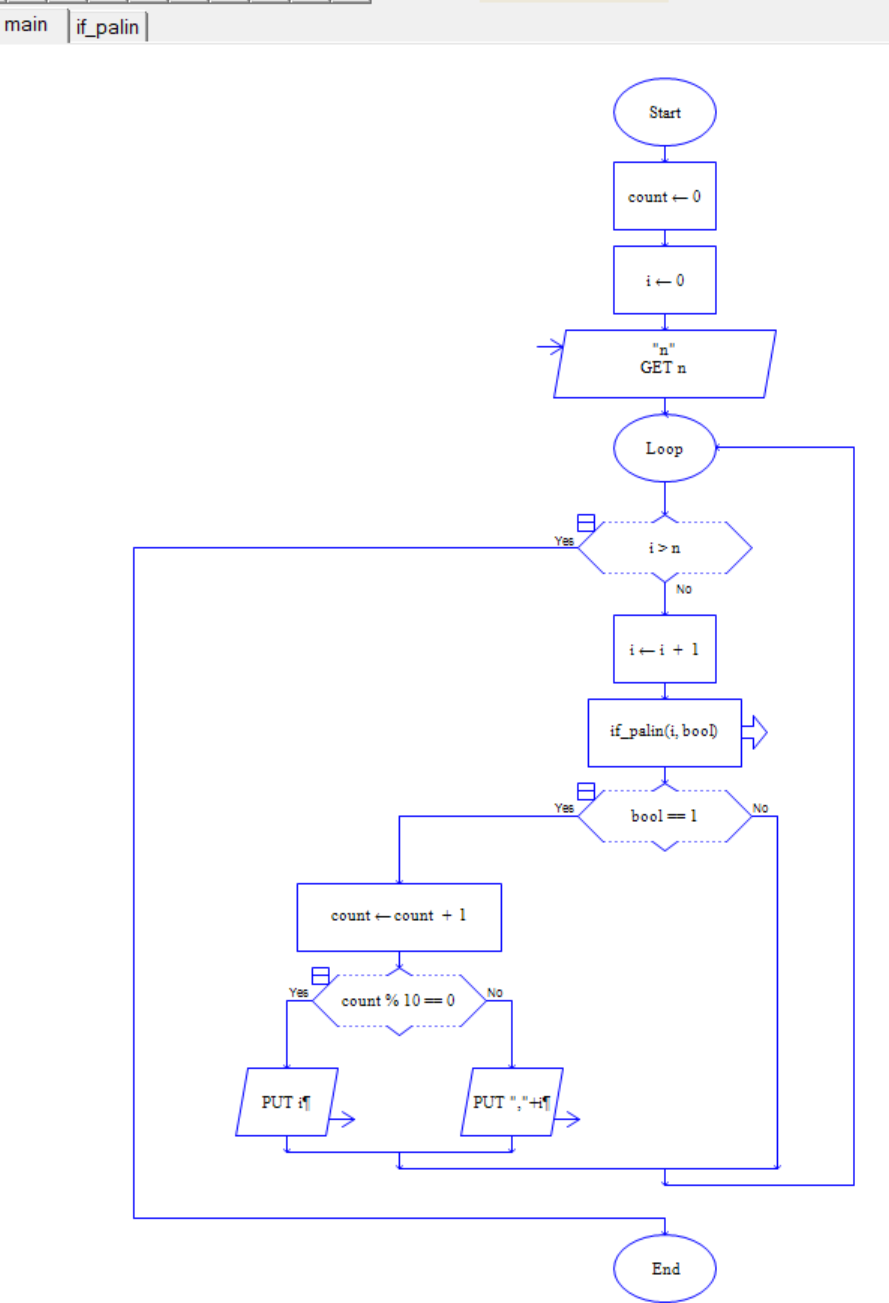
看到题目是数字，那么就可以通过对10取余得到个位数，整除10得到前面的位数，然后循环再取余就可以啦！

Raptor算法流程图如下，

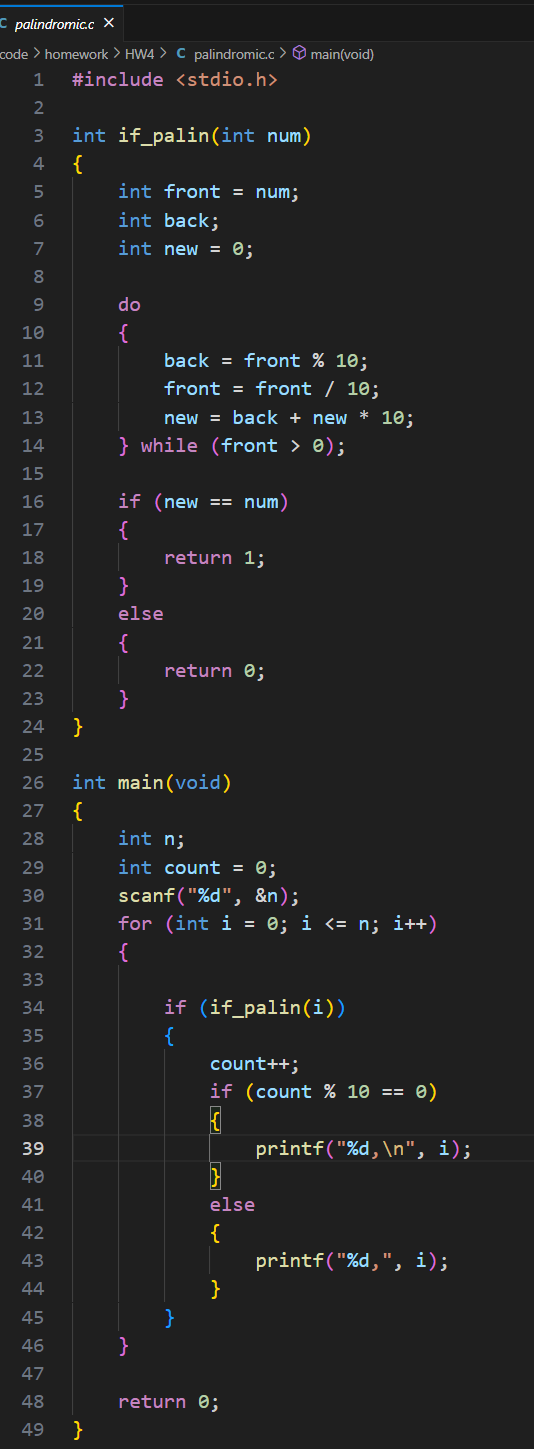
功能函数：

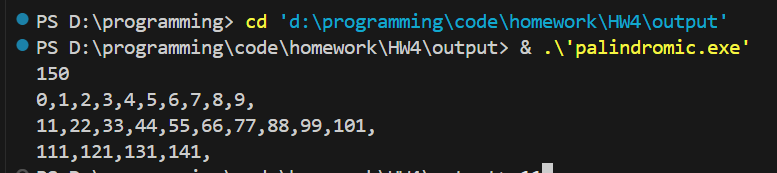


主函数：



实验代码、验证与结论：





输出的结果与样例是一样的，最后一个样例有逗号，是因为样例输入里也有逗号。

评测通过！



总结与心得体会：

老师说的非常有道理，哪个实验就要用哪个方法，这样才有助于我们思考到更优化的算法。

就像这道题，如果我只想到字符串的方法，就不会去琢磨这种取余的算法了。显而易见，后一种方法更简洁。

实验题目4：储蓄账户余额计算器

假设你每月在储蓄账户上存100元，年利率是5%，则每月的利率是0.05/12=0.00417。

第一个月后，账户上的值变成100×(1+0.00417)=100.417；

第二个月后，账户上的值变成(100+100.417)×(1+0.00417)=201.252；

第三个月后，账户上的值变成(100+201.252)×(1+0.00417)=302.507；

以此类推。

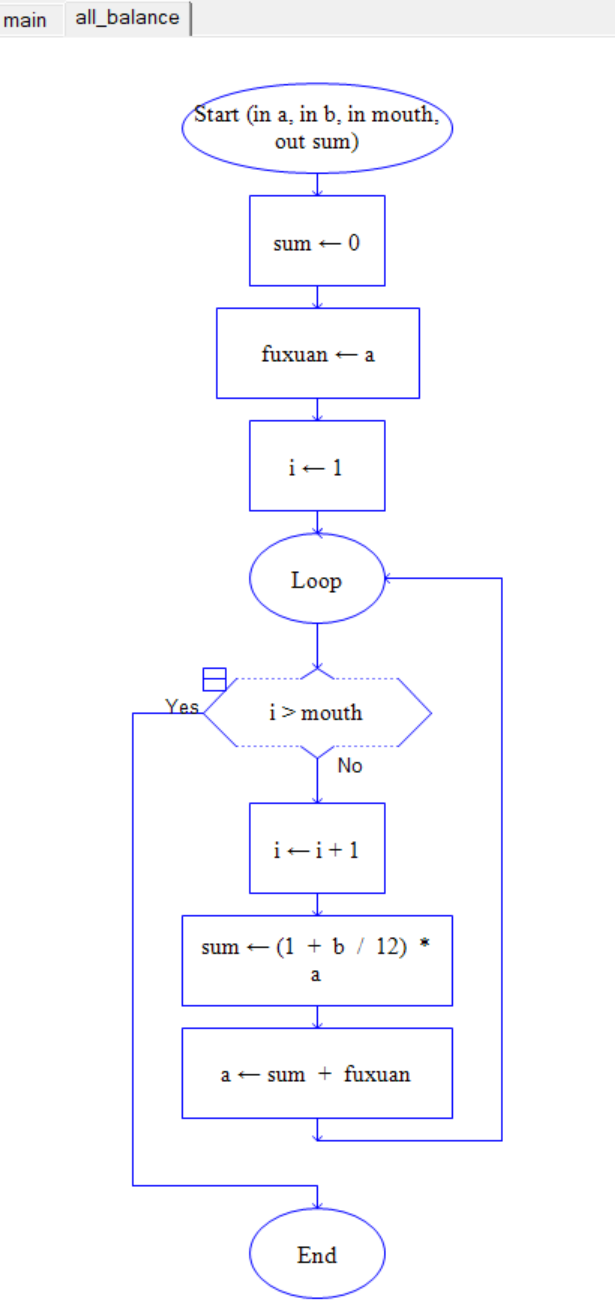
写一个函数，根据用户输入的每月存款数、年利率和月份数，计算给定月份后账户上的钱数并输出。

设计思想：

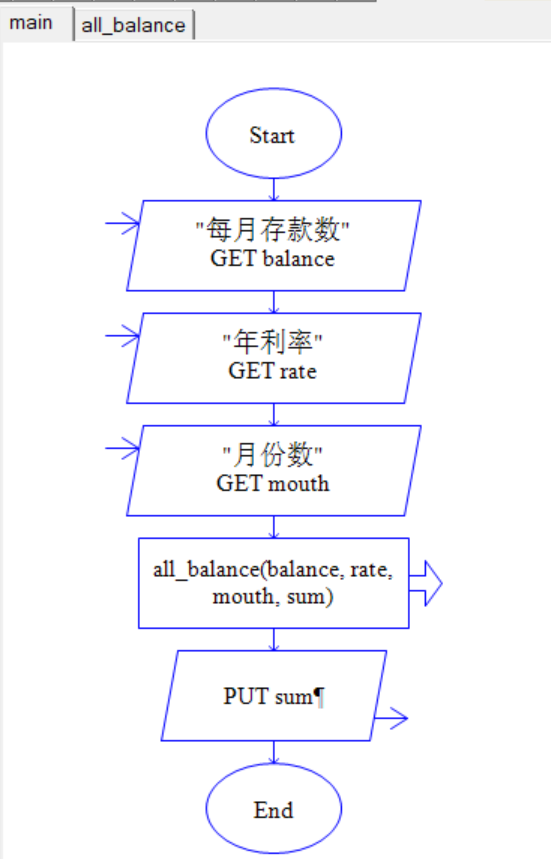
这道题同样是套公式的题，只要注意不要把还没有用到里面的值的变量给覆盖掉了就可以。注意每月存入的都是固定的金额。

Raptor算法流程图如下，

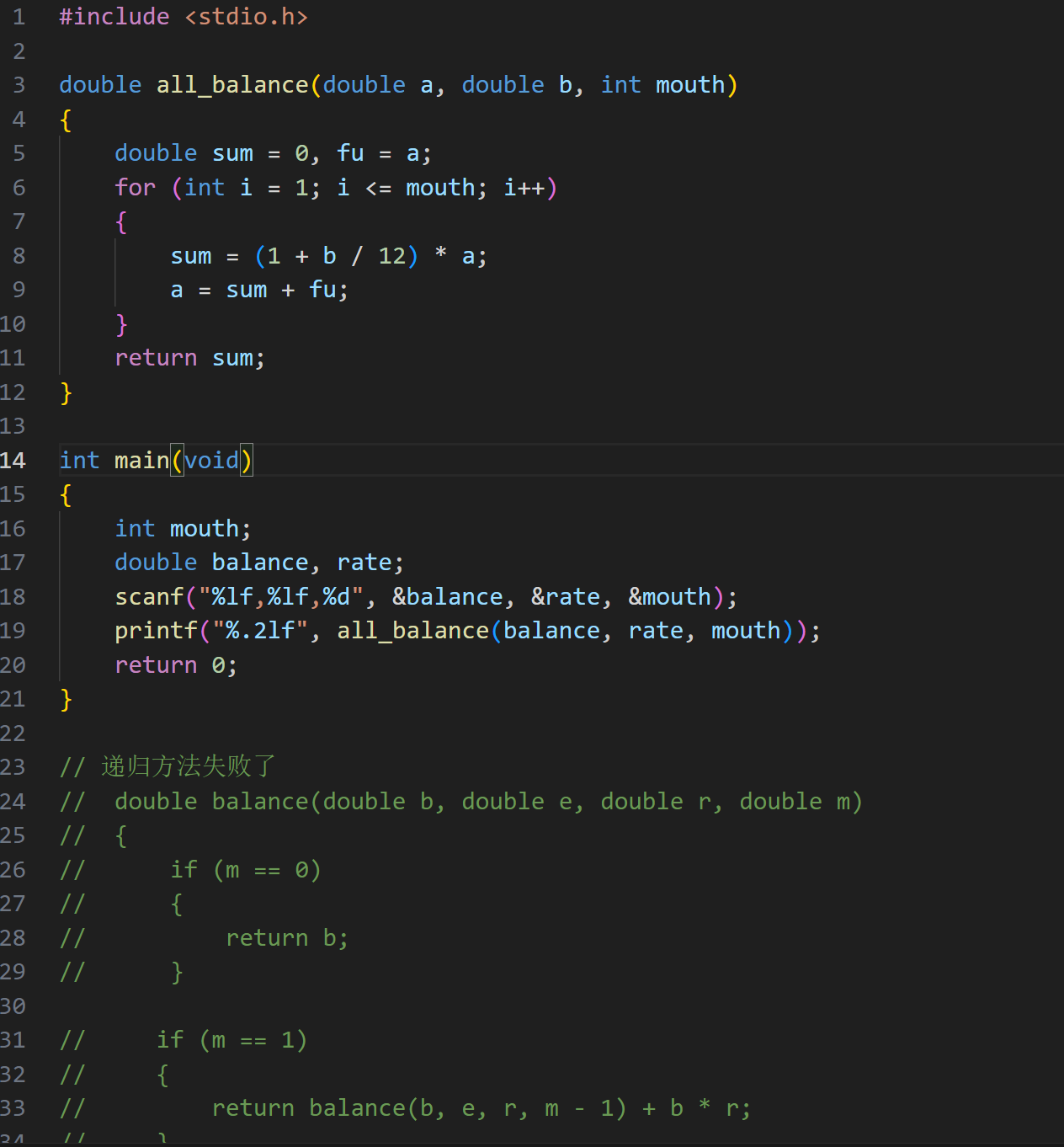
功能函数：

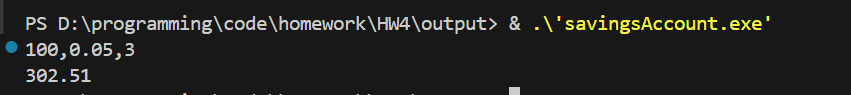


主函数：



实验代码、验证与结论：







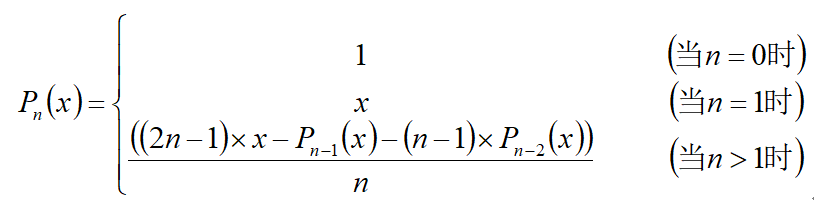
总结与心得体会：

这道题我犯的第一个错误就是把月存入数的值给用变量的值覆盖掉了，导致第三次循环结果就与样例差了0.01，后面结果误差越来越大，后来用演算纸演算了一下才找到原因。

我一开始想用递归算法，后面的结果也是有问题的，一会儿重新修改一下，看看能不能找到问题所在。

实验题目5：求解勒让德多项式

使用递归方法求n阶勒让德多项式的值，递归公式为：



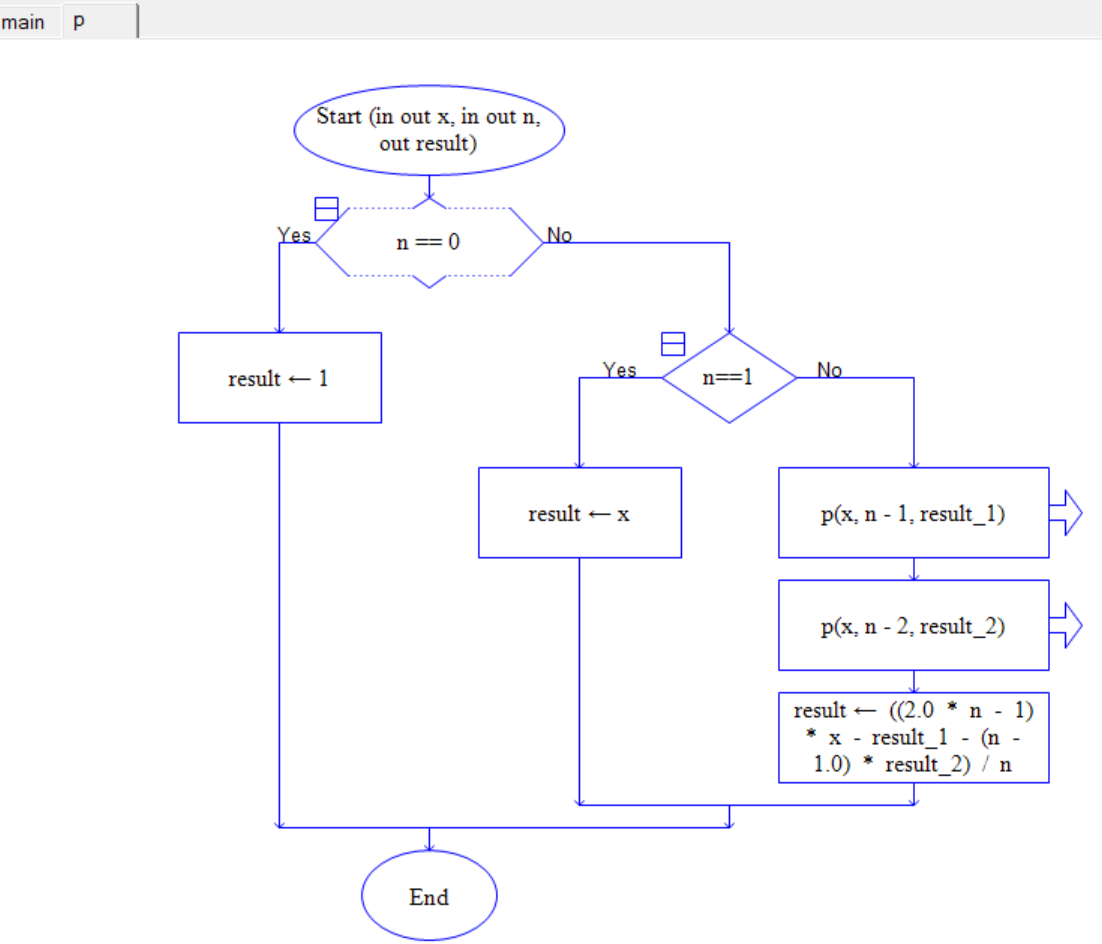
其中：n和x为int型；Pn(x)为float型。

设计思想：

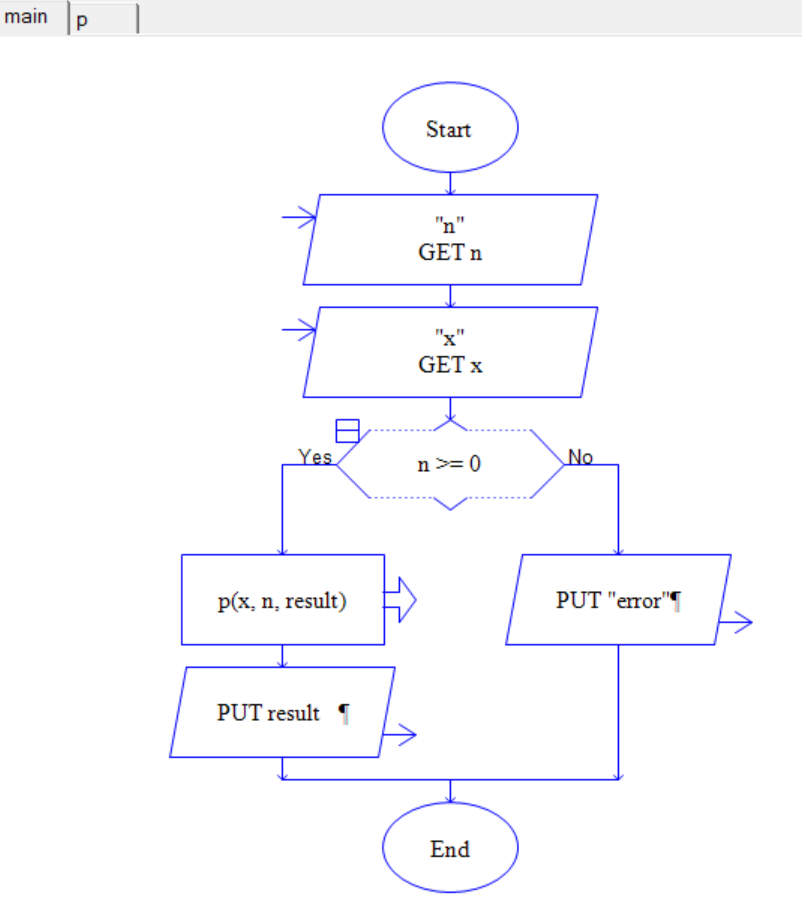
照着公式抄就可以，注意这道题要用到递归。

Raptor算法流程图如下，

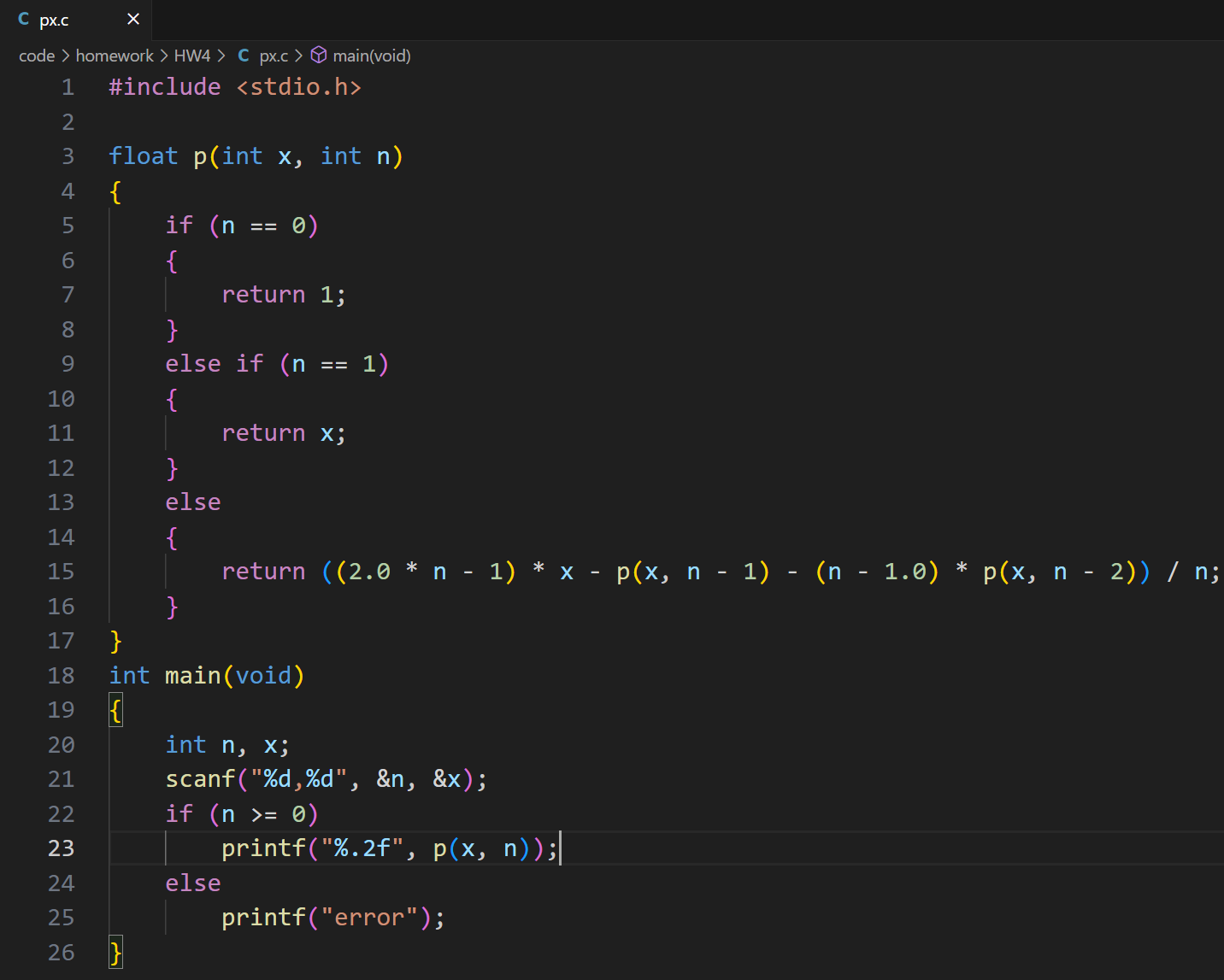
递归函数：

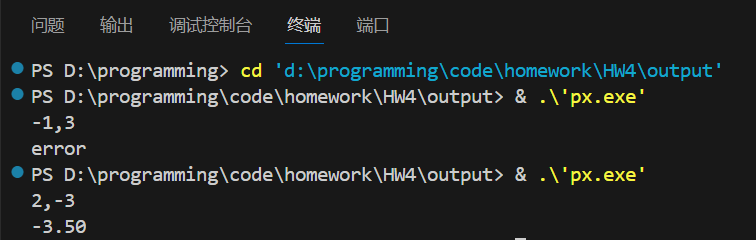


主函数：



实验代码、验证与结论：







总结与心得体会：

递归其实就是函数里调用函数。不过，递归必须要有结束条件，否则程序就会崩溃。

实验题目6：闰年问题

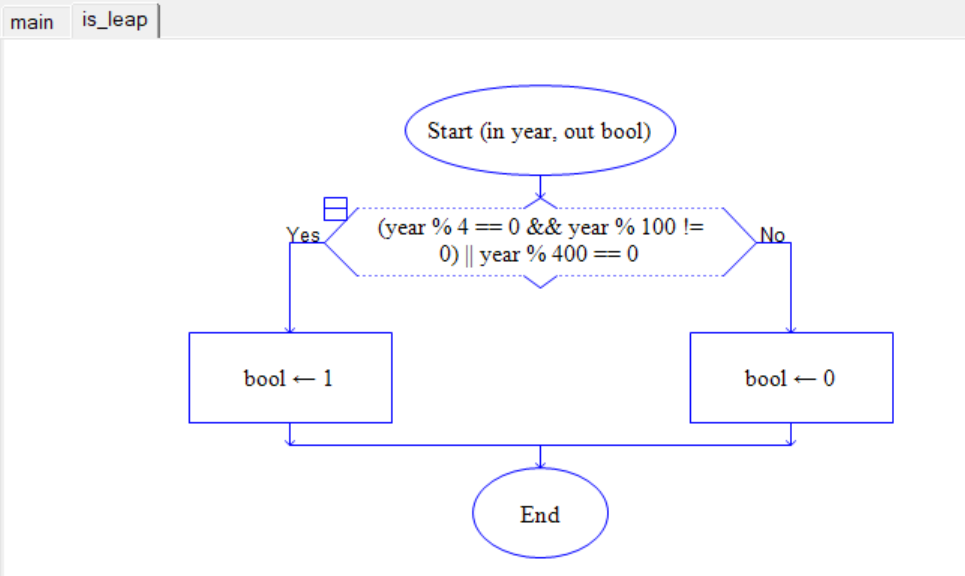
定义一个带参数的宏，以判别某个年份是否为闰年。其中年份year应从键盘输入。

设计思想：

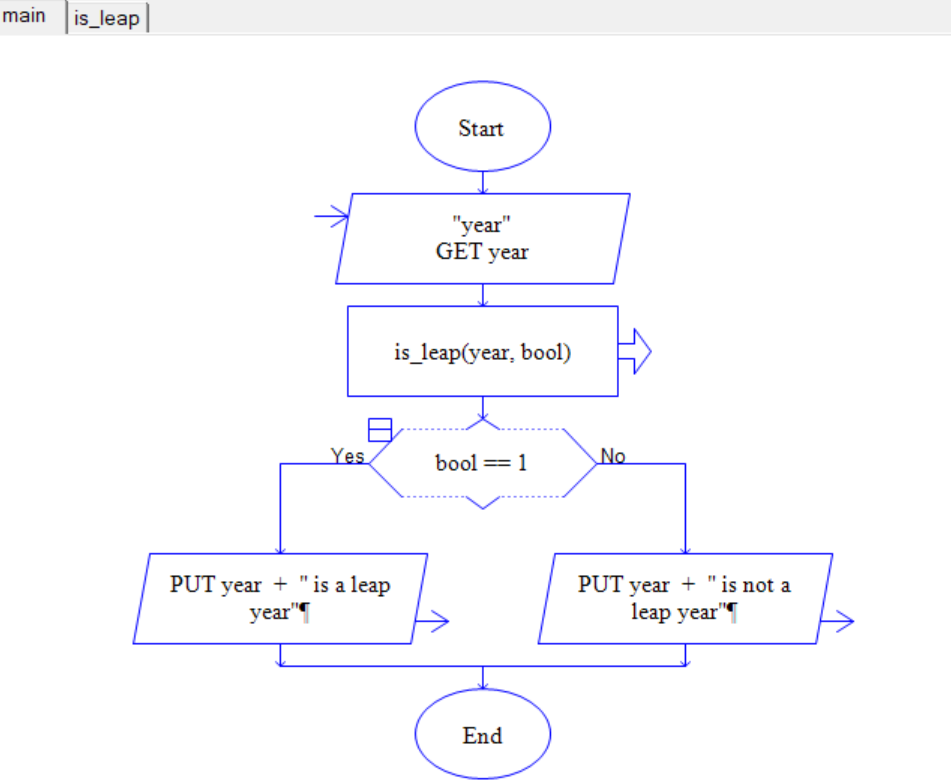
上网检索，得知：四年一闰，百年不闰，四百年再闰。

那就简单多了嘛，raptor算法流程图如下，

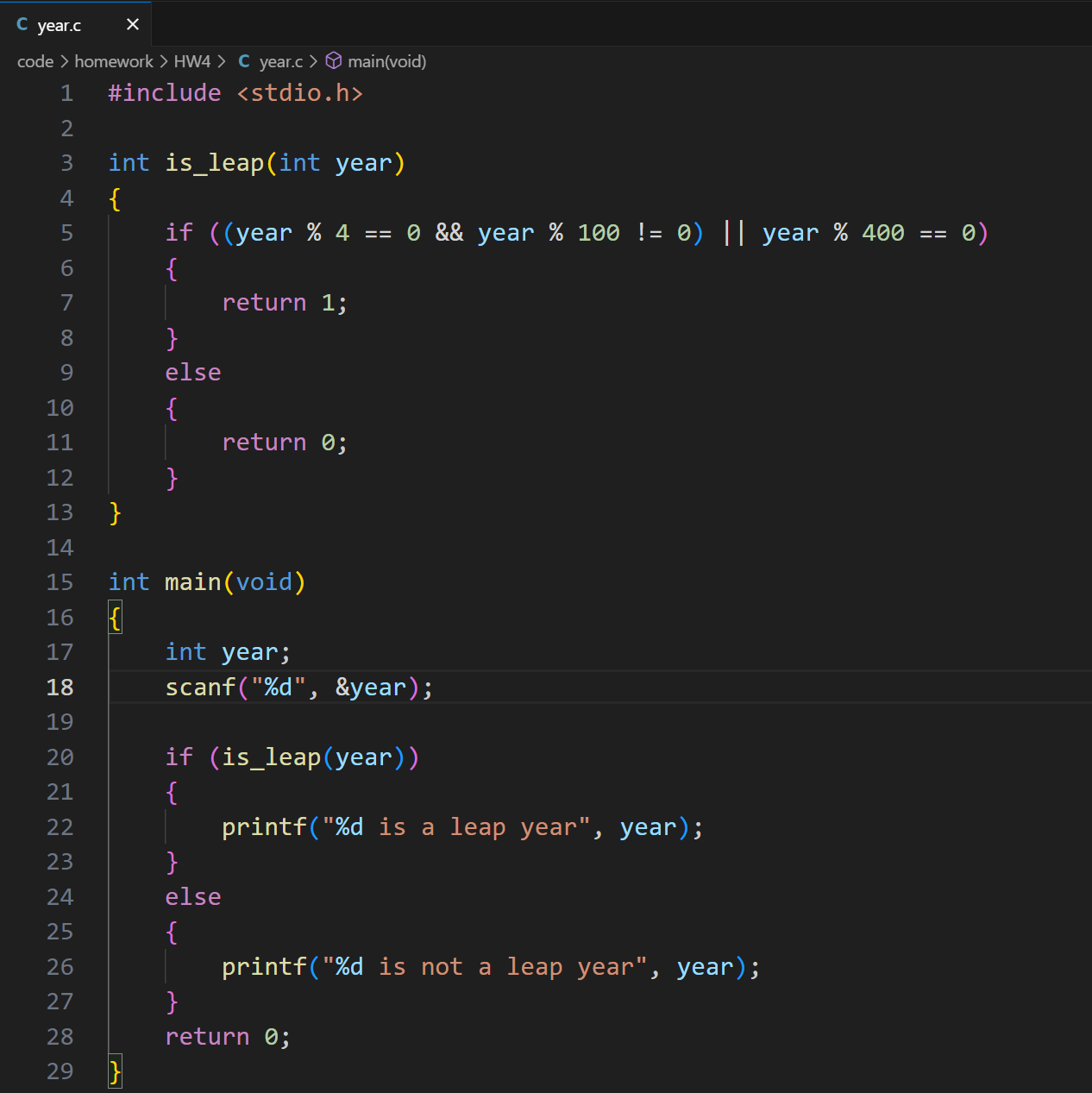
功能函数：

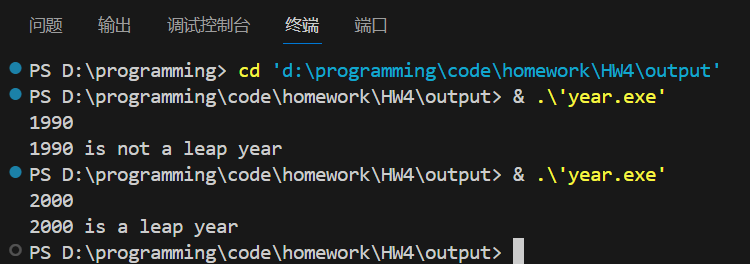


主函数：



实验代码、验证与结论：







总结与心得体会：

在复杂的逻辑运算与算术运算中，多加几个括号来表示优先级更不容易出错，可读性也更强。