第1次实验

1. 判断整数n是否为质数

写出判断整数n是否为质数的函数，在主函数中验证

【分析】

质数的定义是除了1和本身之外没有其他约束，因此判断n是否为质数，根据定义直接判断从2到n-1是否存在n的约数即可。

算法伪代码如下：

bool isPrime\_1( int n )

{

若n小于2的数不是质数，返回0；

循环检测从2到n-1的整数是否有n的因子，若有返回0； // 用求余判断

返回 1; //都除不尽则是质数

}

上面的判断方法效率极低，对于每个数n，其实并不需要从2判断到n-1，我们知道，一个数如果可以进行因数分解，那么分解时得到的两个数一定是一个小于等于sqrt(n)，一个大于等于sqrt(n)，因此，上述代码中并不需要遍历到n-1，只需遍历到sqrt(n)即可，因为若sqrt(n)左侧找不到约数，那么右侧也一定找不到约数。

改进后的算法如下：

bool isPrime\_1( int n )

{

若n小于2的数不是质数，返回0；

循环检测从2到**sqrt(n)**的整数是否有n的因子，若有返回0； // 用求余判断

返回 1; //都除不尽则是质数

}

* **实验1：根据上面改进后的算法编写程序，并在main函数验证**

1. 列举整数N的所有质因子

利用上面的函数，编程列举出整数N的所有质因子

【分析】

质因子的定义是能整除给定正整数的质数，也就是质数的因子。要找出N的所有质因子，可以在找N的因子的同时，去判断该因子是否是质数。

求出一个数的所有因子的代码：

void printFactor(int m) {

for (int i = 2; i <= m / 2; i++)

{

if (m%i == 0)

{

printf("%3d ", i);

}

}

}

结合上面判断质数的函数，可以列举出整数N的所有质因子。

* **实验2：编写程序列举整数N的所有质因子，并在main函数验证**

1. 筛法求质数表程序加注释

* **实验3：**

将书中筛法求质数表的函数抄写下来，对其中每个for循环的作用加上注释

1. 一元多项式除法

* **实验4：**

按书上对一元多项式的存储形式，写出一元多项式除法的程序，并将书上除法的例子数据（除以）代入，输出结果

【分析】

一元多项式除法可以仿照数字的竖式除法，用减法来实现带余项的除法。其演算过程如 下：

（1）把被除式、除式按变量作降幂排列，并把所缺的项用零补齐；

（2）用被除式的第一项除以除式第一项，得到商式的第一项；

（3）用商式的第一项去乘除式，把积写在被除式下面（同类项对齐），消去相等项， 把不相等的项结合起来；

（4）把减得的差当作新的被除式，再按照上面的方法继续演算，直到余式为零或余式 的次数低于除式的次数时为止。

实现思路：

（1）设 P(X)为被除式、Q(x)为除式、R(x)为商式，若 P(X)最高次项为 m 次，Q(x) 最高次项为 n 次，则 R(x) 最高次项为 m-n 次；

（2）若用数组 a、 b、 c 存储 P(X)、Q(x)、 R(x)，则第一次运算用 a[m]/b[n]，并将结果放到 c[m-n]中，然后从数组 a 最高次开始依次减去 c[m-n]乘以数组 b；

（3）第二次运算用新的数组 a 和数组 b 继续上面的运算过程（此时 m 变为 m-1）；

（4）重复执行第（2）和（3）步，直到数组 a 全为 0 或者数组 a 中的最高次项（即不为 0 的最大下标）小于数组 b 中的最高次项。

下面给出多项式相除函数：

void div(float a[], int m, float b[], int n, float c[])

{

int i, j;

for (i = m - n; i >= 0; i--) {

c[i] = a[i + n ] / b[n ]; // 商式系数

for (j = n; j >= 0; j--) {

a[i + j] = a[i + j] - b[j] \* c[i];

}

}

}

1. （选作）求N！的尾部有连续多少个零（N>=50）

【分析】

N较大时，N！无法直接计算出来（超界了）。

如果N的阶乘为K和10的M次方的乘积，那么N!末尾就有M的0。如果将N的阶乘分解后，那么N的阶乘可以分解为： 2的X次方，3的Y次方，5的Z次方，.....的乘积。由于10 = 2 \* 5,所以M只能和X和Z有关，每一对2和5相乘就可以得到一个10，于是M = MIN(X,Z)，不难看出X大于Z，因为被2整除的频率比被5整除的频率高的多。所以可以把公式简化为M=Z。

由上面的分析可以看出，只要计算出Z的值，就可以得到N!末尾0的个数。那么Z的值如何得到呢？

**方法一：**

依次计算2、3、…….、N中每个数字包含多少个因子5，再把这些数字加起来即可。伪代码可写作：

Count = 0; //记录因子5总数

for( k=5; k<=N; k=k+5) // 仅考查5的倍数

{

计算k包含的因子5的个数，存入m； // 循环

Count =count + m;

}

**方法二：**

Z = N/5 + N /(5\*5) + N/(5\*5\*5)+.....，直到N/（5的K次方）等于0. 这里的除法是整除。

举例说明上述公式的来源：

数字N=126，那么包含因子5的小于等于126的数字是5、10、15、20、…、125。其中每个数字含有因子5的个数可能不同。比如10、15含有1个因子5，而25、50含有2个因子5。因此我们作如下图形。



图中将每个数字含有因子5的个数竖向拆分成若干个1排列起来。因此将上图所有1加起来就是126！中因子5的总数。显然有：

N/5表示图中最下面一行1的数目；N/(5\*5)表示图中中间一行1的数目；N/(5\*5\*5) 表示图中最顶层一行1的数目；于是126！中因子5的总数等于N/5 + N /(5\*5) + N/(5\*5\*5)。

以此类推，有N！中5的总数等于N/5 + N /(5\*5) + N/(5\*5\*5)+.....，直到N/（5的K次方）等于0。

* **实验5：**

按照上面方法1或2编写程序计算N！尾部有几个0。N由用户输入。