# 第1次编程练习报告

姓名：张丛 学号：2113662 班级：信安一班

##### **编程练习1——埃氏筛**

* **源码部分：**

#include<iostream>

using namespace std;

const int N = 1000000;

void eratosthenes(int\* p) //埃氏筛函数

{

for (int i = 2; i < ceil(sqrt(N)); i++)

{

int j = 2 \* i;

for ( p[j]!=0; j <= N; j+=i)

{

p[j] = 0;

}

}

return;

}

int main()

{

int\* p = new int[N + 1];

for (int i = 0; i <= N; i++)//初始化

{

p[i] = i;

}

p[1] = 0;

eratosthenes(p);

int count = 0; //计数

for (int i = 2; i <= N; i++)

{

if (p[i] != 0)

{

cout << p[i] << " ";

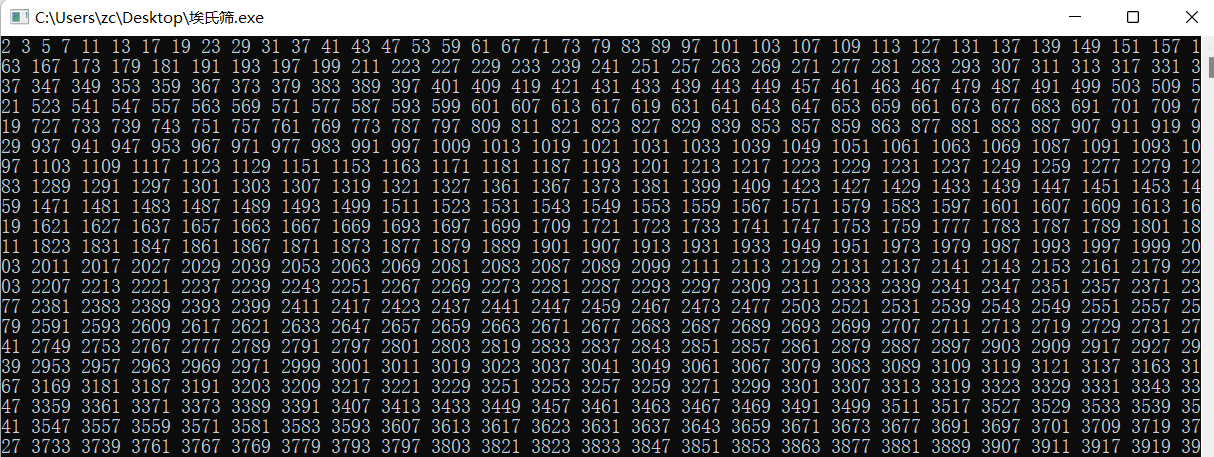
count++;

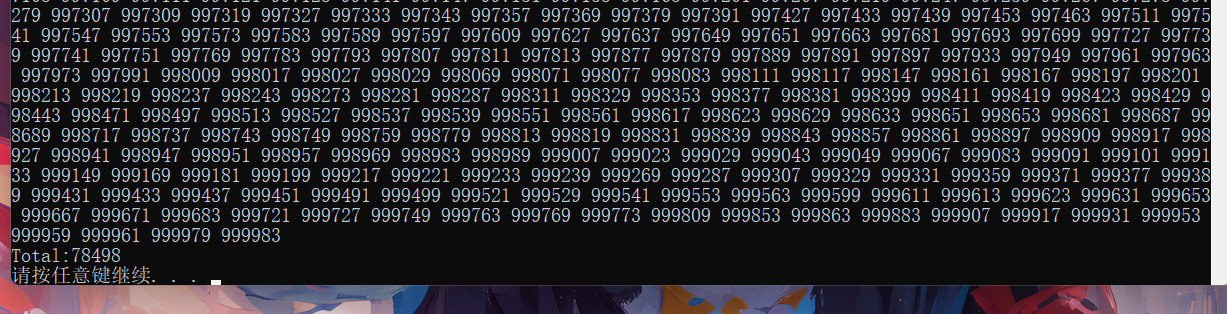
}

}

cout << endl << "Total:" << count << endl;

return 0;

* }
* **说明部分：**
* 主要算法在埃氏筛函数部分：自然数组初始化完成后，从2开始，将2的倍数置为零，然后3的倍数置为零，到数字4时，4已被筛掉（包括4的倍数），再筛5的倍数，以此类推，最终未置零的即为素数。
* **运行示例：**
* 



* **其他：**

埃氏筛要优于一般的素数算法。

但埃氏筛存在重复筛的情况，时间复杂度为O(n\*log n),若数据过大，仍会浪费太多性能。

##### **编程练习2——最小公倍数和最大公因数**

* **源码部分：**

#include<iostream>

using namespace std;

int Maxfactor(int a, int b)//求最大公因数

{

if (b < 0)

{

b \*= -1;

}

if (a < 0)

{

a \*= -1;

}

if (b > a)

{

int temp = b;

b = a;

a = temp;

}

int r, q;

while (true)

{

r = a % b;

q = (a - r) / b;

if (r != 0)

{

a = b;

b = r;

}

else

{

break;

}

}

return b;

}

int Minmul(int a, int b)//利用最大公因数求最小公倍数

{

int g = Maxfactor(a, b);

int m = a \* b / g;

return m;

}

int main()

{

int a = 9876, b = 6789;

cout << "a=" << a << endl << "b=" << b << endl;

int g = Maxfactor(a,b);

int m = Minmul(a, b);

cout << "gcd(a,b)=" << g << endl;

cout << "lcm(a,b)=" << m << endl;

system("pause");

return 0;

* }
* **说明部分：**

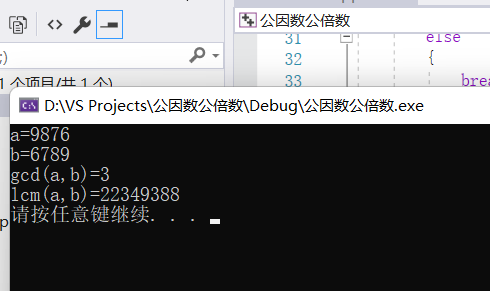
利用辗转相除法求最大公因数：

下一次循环中，被除数为上一次的除数，除数为上一次的余数。

当余数为0时，跳出循环，返回当次除数（即上一个余数）为最大公因数。

考虑到负数，以及传值的大小不确定，函数中有取绝对值、交换a和b等操作。

* **运行示例：**



##### **编程练习3——算术基本定理**

* **源码部分：**

#include<iostream>

using namespace std;

bool Isprime(int n)//判断是否是素数

{

if (n == 0 || n == 1)

{

return false;

}

for (int i = 2; i <=sqrt(n); i++)

{

if (n % i == 0)

{

return false;

}

}

return true;

}

int main()

{

int n;

cout << "Please input n(n>0):";

cin >> n; //输入

int N = n;

int\* prime = new int[n + 1];

int k = 0;

for (int i = 2; i < n; i++) //素数数组prime

{

if (Isprime(i))

{

prime[k] = i;

k++;

}

}

int\* count = new int[k + 1]; //用于存储幂指数

for (int i = 0; i < k + 1; i++)

{

count[i] = 0;

}

for (int i = 0; i <= k; i++) //得到标准分解式

{

while (true)

{

if (n % prime[i] == 0)

{

n = n / prime[i];

count[i]++;

if (n == 1)

{

break;

}

}

else

{

break;

}

}

}

cout << N << "=";

int flag = 0; //控制输出格式\*

for (int i = 0; i <= k; i++) //打印输出

{

if (count[i])

{

if (flag == 0)

{

flag = 1;

cout << prime[i] << "^" << count[i];

}

else

{

cout << "\*"<<prime[i] << "^" << count[i];

}

}

}

cout << endl;

system("pause");

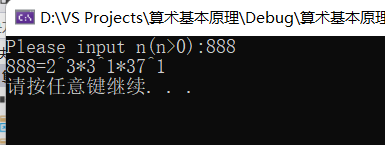
return 0;

* }
* **说明部分：**

得到标准分解式部分：

遍历小于n的素数数组；对于prime[i],如果此素数能整除n，则count[i]++存储幂指数，n=n/prime[i]更新n，并且继续循环直到不能整除，再遍历下一个素数。

另外，需要注意输出格式。

* **运行示例：**
* 
* 