# 第1次编程练习报告

姓名：齐明杰 学号：2113997 班级：信安二班

##### 编程练习1——筛素数

* **源码部分：**
* #include <iostream>
* #include <cmath>
* **using** **namespace** std;
* **const** **int** N = 1000000;
* **bool** vis[N];
* **void** Eratosthenes(**int** n){
* **int** t = sqrt(n);
* **for**(**int** i=2;i<=t;i++){
* **if**(!vis[i]){
* **for**(**int** j = 2\*i;j<=n;j+=i){
* vis[j] = **true**;
* }
* }
* }
* }
* **int** main(){
* printf("Please input the range: ");
* **int** a,b,cnt=0; **char** s;
* scanf("%d%c%d", &a, &s, &b);
* vis[0] = vis[1] = **true**;
* Eratosthenes(N);
* **for**(**int** i=a;i<=b;i++){
* **if**(!vis[i]) cnt++,printf("%d, ",i);
* }
* printf("\nTotal: %d",cnt);
* system("pause");
* **return** 0;
* }
* **说明部分：**

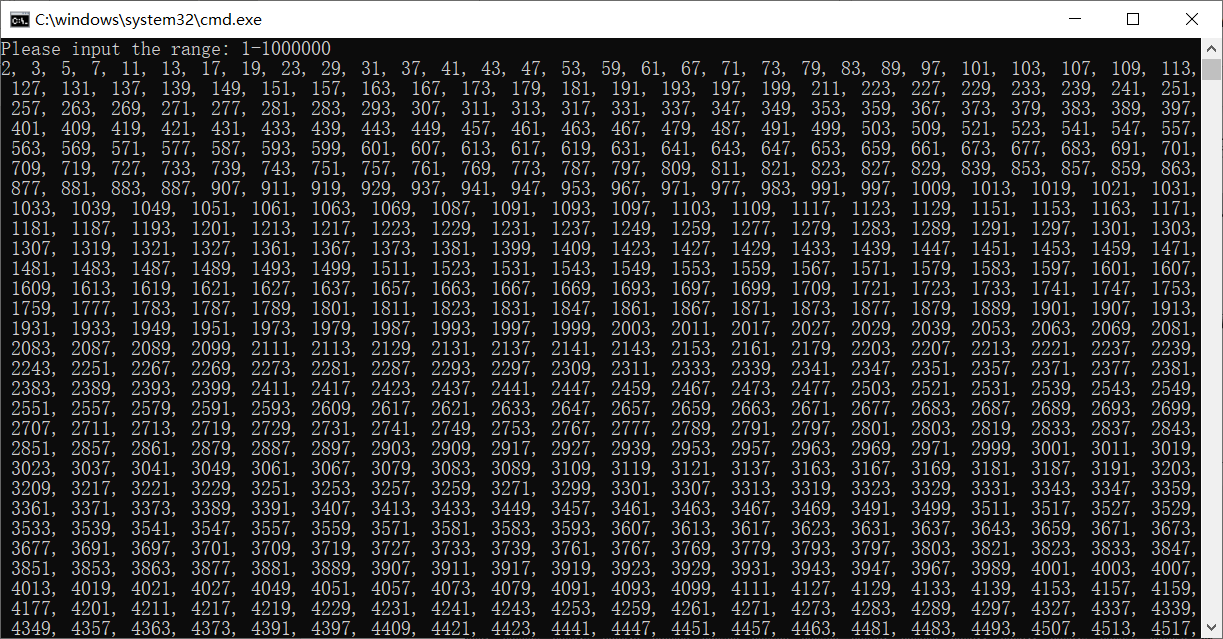
bool vis[N]是用于筛选素数的数组，对应值为false的为素数，true的为合数。一开始vis[N]全部初始化为false.

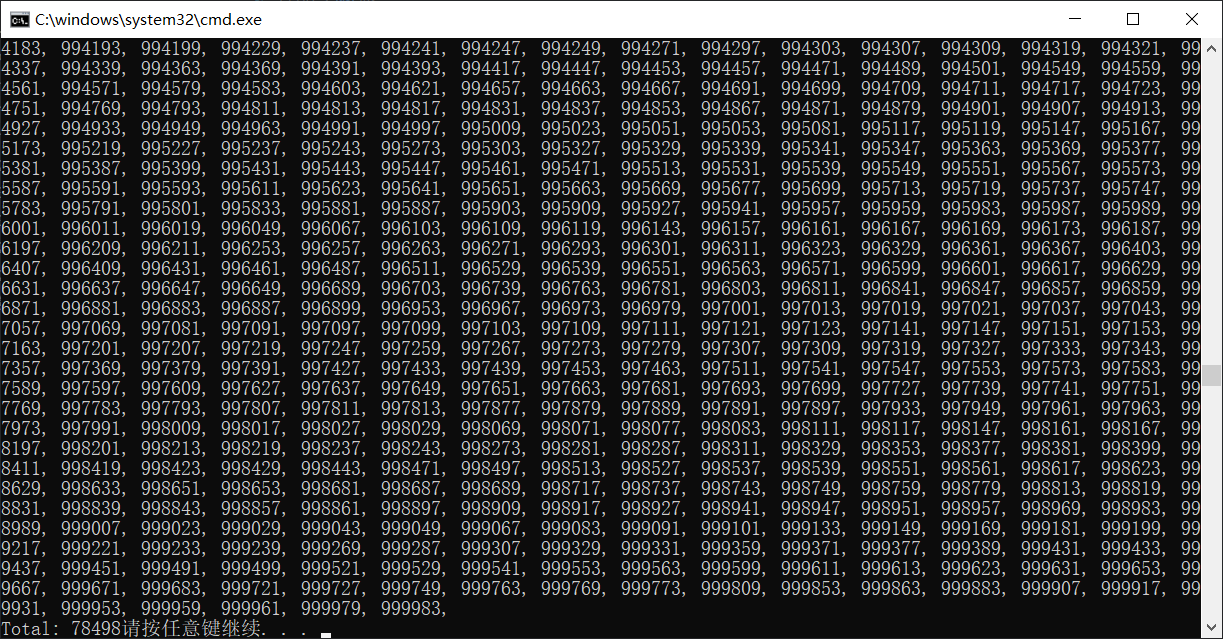
void Eratosthenes(int n)函数用于将1-n内的素数筛选出来，由于只需筛选出根号n以内的素数，再筛选出其各个素数的倍数，故

取int t = sqrt(n)，令i从2遍历到t，对于每一个i，由于从2开始迭代筛选素数，故若当前i满足vis[i]=false，则表明i必为素数，将其倍数依次删除即可。重复以上的过程，在动态寻找素数的过程中，找到了即标记其倍数为合数。最终vis[N]内仍为false值的为素数。

在main函数中，首先0和1不是素数，故vis[0]和vis[1]初始化为true，然后使用埃氏筛法筛选后，遍历vis数组找出素数即可。

* **运行示例：**





* **其他：**

a. Eratosthenes算法利用合数的质数因子来筛出这个合数，而普通筛素数需要对每个数字判断其是否为素数，是暴力枚举，其复杂度比埃氏筛算法复杂度高，因此埃氏算法通常会更快。

b. Eratosthenes筛法可以通过分段进行优化，将范围分成更小的部分，然后逐个分别筛选素数，从而避免内存不足的问题，即采用递归的思想。

c.求更大级别的素数，该方法由于很多数字会被重复筛去，如6被2筛去，又会被3筛去等等，操作重复，效率不高，因此对更大级别的素数筛选不适用，会导致时间超时，同时由于判断素数的数组长度有限，空间超限也是面对大数字存在的问题，如需要优化，最好采用欧拉筛法。

##### 编程练习2——计算gcd和lcm

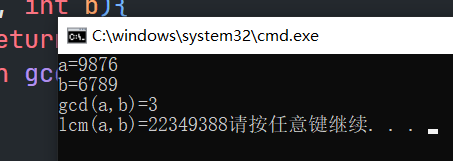
* **源码部分：**
* #include <iostream>
* **using** **namespace** std;
* **int** gcd(**int** a, **int** b){
* **if**(b==0) **return** a;
* **else** **return** gcd(b,a%b);
* }
* **int** lcm(**int** a, **int** b){
* **return** a\*b/gcd(a,b);
* }
* **int** main(){
* **int** a,b;
* printf("a=");
* scanf("%d", &a);
* printf("b=");
* scanf("%d", &b);
* printf("gcd(a,b)=%d\n", gcd(a,b));
* printf("lcm(a,b)=%d", lcm(a,b));
* system("pause");
* **return** 0;
* }
* **说明部分：**

Int gcd(int a, int b)函数返回a和b的最大公因数，其采用辗转相除法进行计算。根据定理可知若a%b不等于0，即a/b=c余d，那么gcd(a,b)=gcd(b,d)即gcd(a,b)=gcd(b,a%b)。不断递归，最后

b=0表明上一轮满足a%b==0,故本轮的a（即上轮的b）即为所求最大公因数。

根据(a,b)[a,b]=a\*b，求出gcd(a,b)后可立马求出lcm(a,b)=a\*b/gcd(a,b)

* **运行实例：**



##### 编程练习3——实现算数基本定理

* **源码部分：**
* #include <iostream>
* **using** **namespace** std;
* **const** **int** MAXN = 2000000;
* **bool** vis[MAXN];
* pair<**int**, **int**> ans[200];
* **int** lens;
* **void** Eratosthenes(**int** n){
* **for**(**int** i=2;i<=n;i++){
* **if**(!vis[i]){
* **if**(n%i==0){
* ans[lens].first = i;
* **while**(n%i==0){
* ans[lens].second++;
* n/=i;
* }
* lens++;
* **if**(n==1) **break**;
* }
* **for**(**int** j = 2\*i;j <= n;j += i) vis[j] = **true**;
* }
* }
* }
* **int** main(){
* **int** n;
* printf("Please input n(n>0): ");
* scanf("%d",&n);
* printf("%d=",n);
* Eratosthenes(n);
* **for**(**int** i=0;i<lens;i++){
* printf("%d^%d", ans[i].first, ans[i].second);
* **if**(i<lens-1) printf("\*");
* }
* system("pause");
* **return** 0;
* }
* **说明部分：**

本题我使用vis[MAXN]来作为Eratosthenes筛法的标记数组，

pair<int,int> ans[200]用于存储分解的因子以及个数，例如：888=2^3\*3^1\*37^1，那么最终ans[0].first=2,ans[0].second=3, ans[1].first=3, ans[1].second=1, ans[2].first=37, ans[2].second=1。

我以埃氏筛的代码为基础，同时在对每次遇到的素数i都判断n%i是否为0，若为0则为n的一个素因子，加入答案中，然后通过n/i来消除该因子，若n仍含i这个因子，则继续n/i同时i的个数+1，直到n不再有i这个素因子为止，然后继续找下一个素因子。

此外，埃氏筛最外层循环为n而不是根号n是由于该函数不仅要完成筛选素数的任务，也要完成找因子的任务，而因子可能遍布在2到n之间。

* **运行实例：**

