**中山大学数据科学与计算机学院本科生实验报告**

**（2016学年春季学期）**

课程名称：**Algorithm design**  任课教师：张子臻

|  |  |  |  |
| --- | --- | --- | --- |
| 年级 | **15** | 专业（方向） | **软件工程(移动信息工程)** |
| 学号 | **15352408** | 姓名 | **张镓伟** |
| 电话 | **13531810182** | Email | **709075442@qq.com** |
| 开始日期 |  | 完成日期 |  |

目录

[1.实验题目 3](#_Toc482891082)

[Soj 1009 Mersenne Composite N 3](#_Toc482891083)

[Soj 1020 高精度 3](#_Toc482891084)

[Soj 1259 Sum of Consecutive Primes 3](#_Toc482891085)

[Soj 1240 Faulty Odometer 3](#_Toc482891086)

[Soj 1231 The Embarrassed Cryptography 3](#_Toc482891087)

[Soj 1203 The Cubic End 3](#_Toc482891088)

[Soj 1119 Factstone Benchmark 4](#_Toc482891089)

[Soj 1500 Prime Gap 4](#_Toc482891090)

[2.实验目的 4](#_Toc482891091)

[3.程序设计 4](#_Toc482891092)

[Soj 1009 Mersenne Composite N 4](#_Toc482891093)

[Soj 1020 Big Integer 7](#_Toc482891094)

[Soj 1259 Sum of Consecutive Primes 7](#_Toc482891095)

[Soj 1240 Faulty Odometer 8](#_Toc482891096)

[Soj 1231 The Embarrassed Cryptography 9](#_Toc482891097)

[Soj 1203 The Cubic End 10](#_Toc482891098)

[Soj 1119 Factstone Benchmark 11](#_Toc482891099)

[Soj 1500 Prime Gap 12](#_Toc482891100)

[5.实验总结与心得 13](#_Toc482891101)

[附录、提交文件清单 13](#_Toc482891102)

# 1.实验题目

**数论专题**

Soj 1009 Mersenne Composite N

**题意：**判断所有质数i<=k，2^i-1是否是质数，不是的话就要将它分解质因数输出来。

**约束：**k<=63

Soj 1020 高精度

**题意：**题意：输入n个整数bi（1 <= i <= n)，以及一个大整数x，输出一个n元组(x mod b1,x mod b2,…,x mod bn)。

**约束：** n <= 100， 1 < bi <= 1000 (1 <= i <= n) 大整数x的位数，m <= 400并且非负

Soj 1259 Sum of Consecutive Primes

**题意：**给出一个正整数n，求出它有多少种方法可以表示成连续的素数的和。 例如53 = 5 + 7 + 11 + 13 + 17 = 53，共有两种方法。

**约束：**数字大小2<=n<=10000

Soj 1240 Faulty Odometer

**题意：**有个损坏的里程表，不能显示数字4，会从数字3直接跳到数字5。给出里程表的读数x，求出实际里程。

**约束：** 1<=x<=999999999

Soj 1231 The Embarrassed Cryptography

**题意：**给出两个正整数K和L，问K是否存在小于L的质因数，有的话则找出最小的质因数。

**约束:** 4 <= K <= 10100 ， 2 <= L <= 106

Soj 1203 The Cubic End

**题意：**题目给出了一个有趣的现象：如果一个数字串，以1， 3， 7， 9结尾，则会有一个数，它的三次方以这个数字串结尾，且它自己长度不会超过这个数字串。

**约束:** 数字串长度1<=n<=10

Soj 1119 Factstone Benchmark

**题意：**1960年发行了4位计算机，从此以后每过10年，计算机的位数变成两倍。输入某一个年份，求出在这个年份的最大的整数n使得n!能被一个字表示

**约束：**年份1960<=n<=2160，且n%10 == 0

Soj 1500 Prime Gap

**题意**：给出一个正整数k，找到与之相邻的两个素数，并求出两个素数之差。如果不存在两个相邻的素数则输出0。

**约束：**1<=k<=1299709

# 2.实验目的

1.学习数论的基础知识，并会做简单的数论题。

2.提高运用课堂所学知识解决实际问题的能力。

# 3.程序设计

Soj 1009 Mersenne Composite N

**解题思路：**这题由于只要判断的数其实很少，完全可以打表水过去，但是作为一个退役的OIer，是不能这么没有出息的。为了解决这道题，我们先知道以下三个知识点：

**1.**费马小定理：假如p是质数，且gcd(a,p)=1，那么 a(p-1)≡1（mod p）。即：假如a是整数，p是质数，且a,p互质(即两者只有一个公约数1)，那么a的(p-1)次方除以p的余数恒等于1。这个定理是很常用的，比如我经常用来求a在mop p 下的逆元，就是a(p-2)mod p

**2**.米勒-拉宾(Miller-Rabin)素数检验。由费马小定理，我们可以猜想出一个验证素数的方法。由于除了2之外其他素数都是奇数，那么我们猜测假如数n满足2^(n-1) mod n=1 那n不就是素数了吗。但人们已经找到了反例。不过如果不满足2^(n-1) mod n=1，那么n一定为合数这个是可以确定的。米勒-拉宾(Miller-Rabin)素数检验就是为了降低费马小定理检验素数出错的概率而提出的，内容如下：

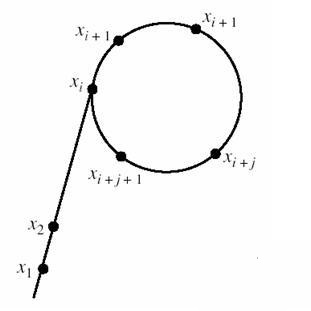
底数由2变为a(a为小于n的任意整数)不断地提取指数n-1中的因子2，把n-1表示成d\*2^r（其中d是一个奇数）那么我们需要计算的东西就 变成了a的d\*2^r次方除以n的余数。于是，a^(d \* 2^(r-1))%n要么等于1，要么等于n-1。如果a^(d \* 2^(r-1))%n等于1，定理继续适用于a^(d \* 2^(r-2))，这样不断开方开下去，直到对于某个i满足a^(d \* 2^i) mod n = n-1或者最后指数中的2用完了得到的a^d mod n=1或n-1。这样费马小定理加强成如下的形式：这样，Fermat小定理加强为如下形式：

尽可能提取因子2， 把n-1表示成d\*2^r，如果n是一个素数，那么或者a^d mod n=1，或者存在某个i使得a^(d\*2^i) mod n=n-1 ( 0<=i<r ) （注意i可以等于0，这就把a^d mod n=n-1的情况统一到后面去了）

从上也可看出米勒-拉宾检验是一个概率算法，对于数n，我们用越多的a去检验，正确概率越高。

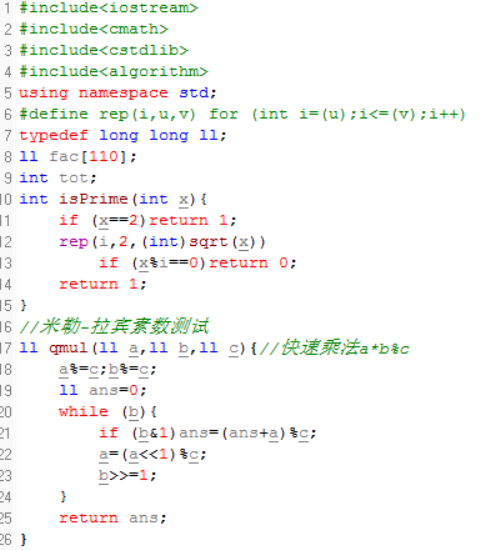
3. Pollard rho大整数质因数分解。大数分解最简单的思想也是试除法，就是从2到sqrt(n)，一个一个的试验，直到除到1或者循环完，最后判断一下是否已经除到1了即可。但是这样的做的复杂度是相当高的。一种很妙的思路是找到一个因子（不一定是质因子），然后再一路分解下去。这就是基于Miller\_rabin的大数分解法Pollard\_rho大数分解。这个算法找一个因子是一种随机化算法随机取一个x1，由x1构造x2，使得｛p可以整除x1-x2 && x1-x2不能整除n｝则p=gcd（x1-x2，n），结果可能是1也可能不是1。如果不是1就找寻成功了一个因子，返回因子；如果是1就寻找失败，那么我们就要不断调整x2，具体的办法通常是x2=x2\*x2+c（c是自己定的或者随机也可以）直到x2出现了循环==x1了表示x1选取失败重新选取x1重复上述过程。

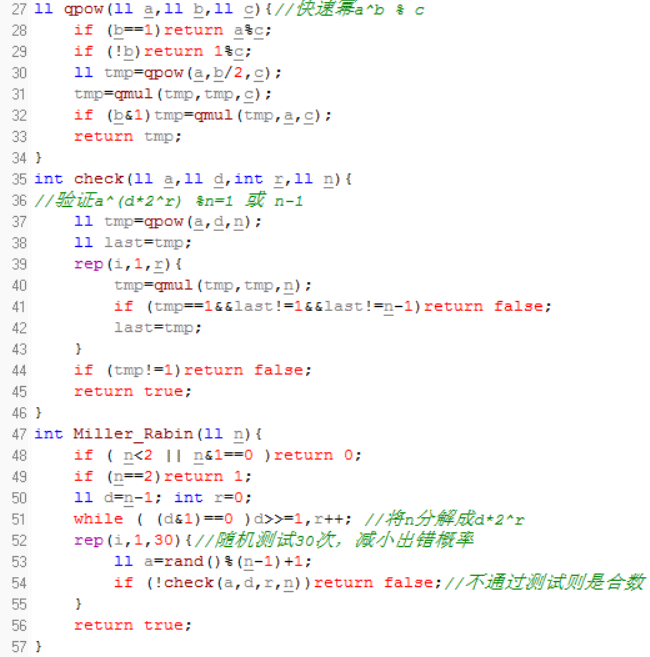
因为x1和x2再调整时最终一定会出现循环，形成一个类似希腊字母rho的形状，故因此得名。

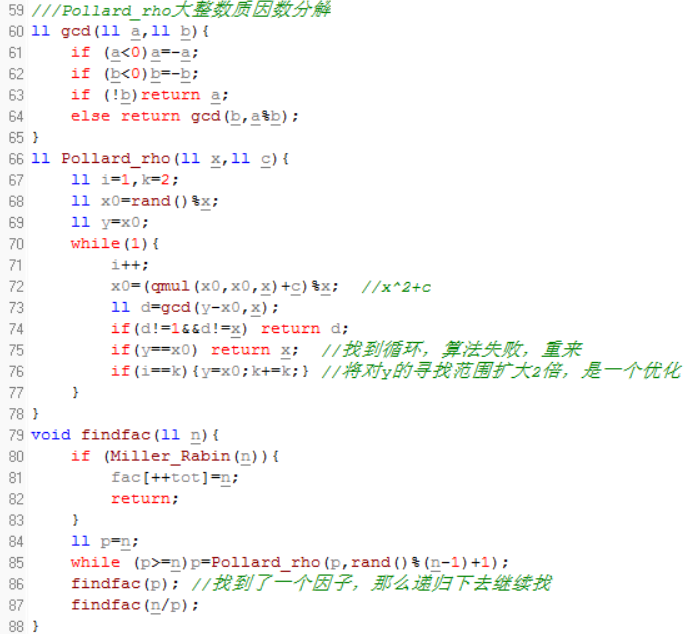


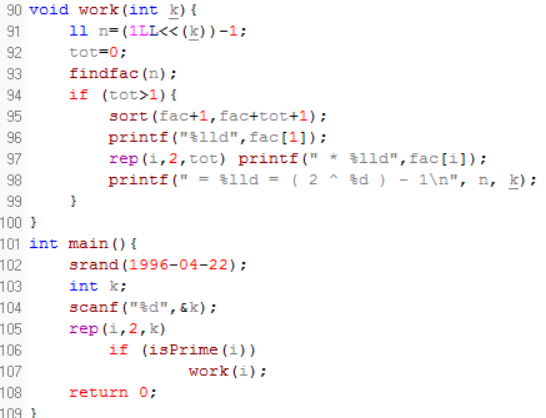
最后回到这题，就是枚举每个质数i<=k，用米勒-拉宾测试判断2^i-1是不是质数，不是的话就用Pollard\_rho算法分解。

**代码：**









Soj 1020 Big Integer

**解题思路：**我们知道：(a+b)%m=(a%m+b%m)%m

(a\*b)%m=(a%m)\*(b%m)%m

而一个n位数，假设从高位到低位依次为X(n-1)，X(n-2)，… ，X0，则它可以分解为X(n-1)\*10n-1 + X(n-2)\*10n-2 +…+ X0\*100 。设f[i]= X(i)\*10i %m,0<=i<n,那么这题的答案就是



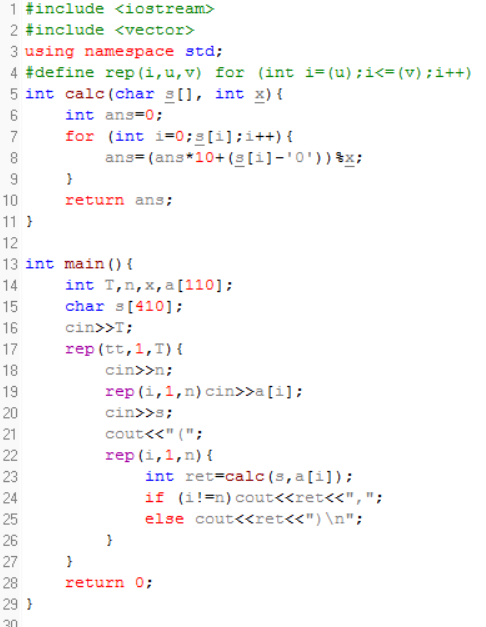
由观察得每一个位乘上的10的某次方是从低位开始一次比一次多乘一个10，所以这题也可以写成迭代的形式更方便我们写代码：

g[0]=Xn-1%m g[i]=(g[i-1]\*10+Xn-1-i)%m

最终答案就是g[n-1].

**时间复杂度：**O(n)

**代码：**

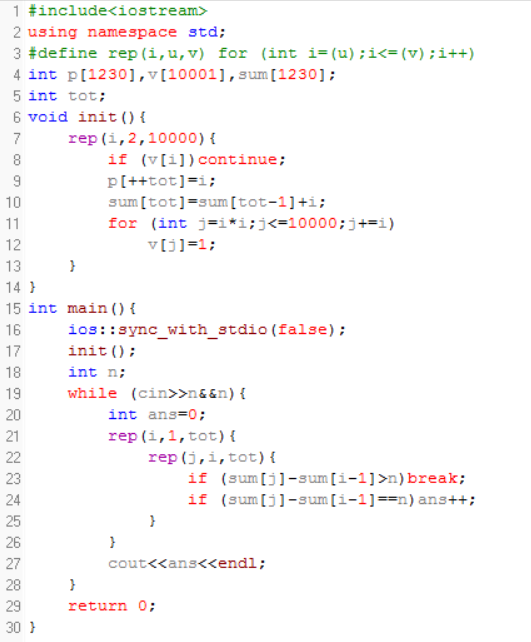


Soj 1259 Sum of Consecutive Primes

**解题思路:**要将一个数分成连续一段素数的和，显然我们首先要知道素数有什么。所以我们一开始先快速筛出10000以内的所有素数。然后发现只有1229个。那么对于给出的每个数n，我们可以枚举连续一段素数的起点和终点，判断该段之和是否为n，这一步可以先求出素数的前缀和sum，那么[i,j]这段的素数和就是sum[j]-sum[i-1]，这样可以方便不少。最后我们统计有多少段素数满足要求即可。

**时间复杂度：**O(n^2)

**代码:**

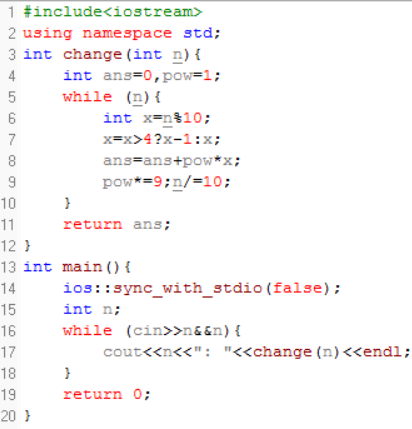


Soj 1240 Faulty Odometer

**解题思路:**这个损坏的里程表只能显示“012356789”这9个数字，它其实就是显示一个9进制数。我们将读数转换成一个真的9进制数，即判断读数的每一位，如果该位小于4，则不变，如果大于4，该位上的数字减一。然后就是简单的九进制转十进制的问题了。假设该n位9进制数从高位到低位依次为X(n-1)，X(n-2)，… ，X0，那么转换成10进制的方法就是X(n-1)\*9n-1 + X(n-2)\*9n-2 +…+ X0\*90

**时间复杂度:**O(n)

**代码:**



Soj 1231 The Embarrassed Cryptography

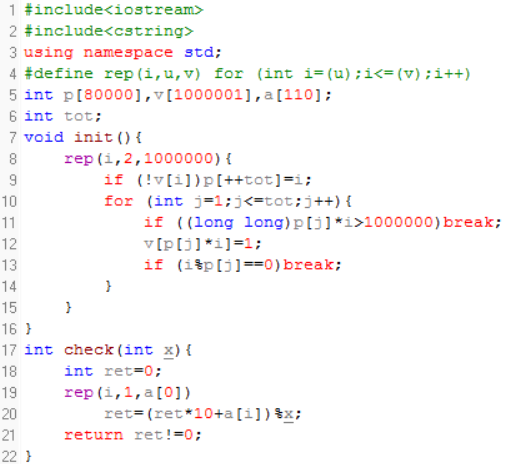
**解题思路:**首先使用O(n)线性筛素数的方法筛出10^6以内的所有素数。从小到大枚举每一个小于L的素数x，判断能否被K整除。但是K是个大整数，显然不能直接用K%x去判断。将K拆成各位数字与10的幂之和的形式X(n-1)\*10n-1 + X(n-2)\*10n-2 +…+ X0\*100。那么K%x的结果就相当于这个多项式中每一项%x之后的和再%x。由于10的幂的连续性，我们同样可将这个做法写成一个递推式：

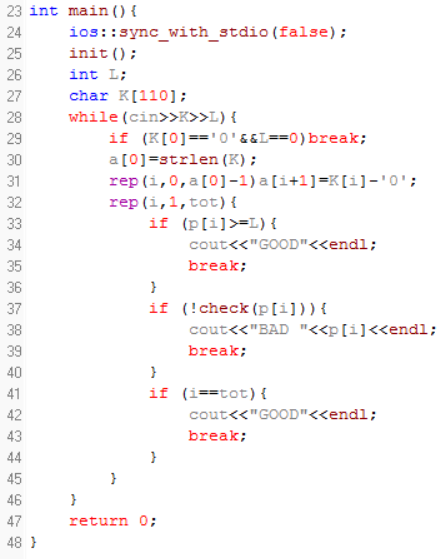
g[0]=Xn-1%x g[i]=(g[i-1]\*10+Xn-1-i)%m

最后只要看g[n-1]是否为0，若为0则说明K能被x整除，输出“Bad x”并退出枚举。若到最后都没有x能整除K，则输出“GOOD”

**时间复杂度:**O(n)

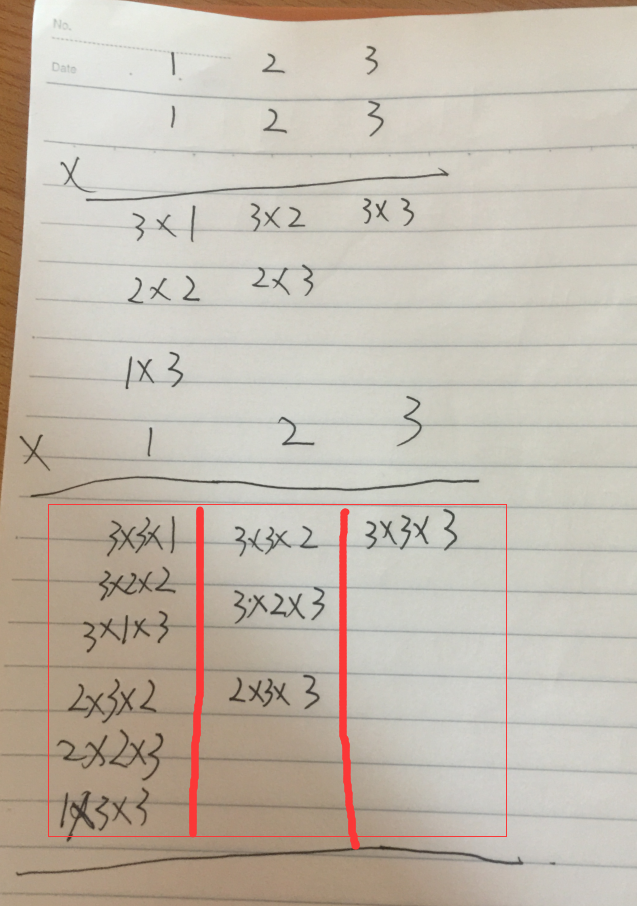
**代码:**





Soj 1203 The Cubic End

**解题思路:**由于数字串长度n很小，只有10，那么一个明显的想法就是从低位到高位从小到大枚举每一位的数字，再检查是否满足条件。那么现在第一个问题就是如何算出三次方之后某一位的数字是多少。我现在以求123的三次方的末三位来做说明：



从图中可以看出，结果的末三位就是红框中对应那一列的数相加。我们现在设x表示第一个123的第x位，设y表示第二个123的第y位，设z表示第三个123的第z位。那么上图整理后第i位的结果与x、y、z的关系如下表：

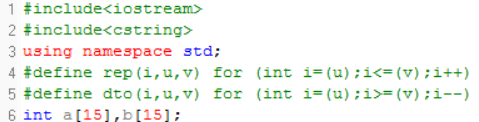
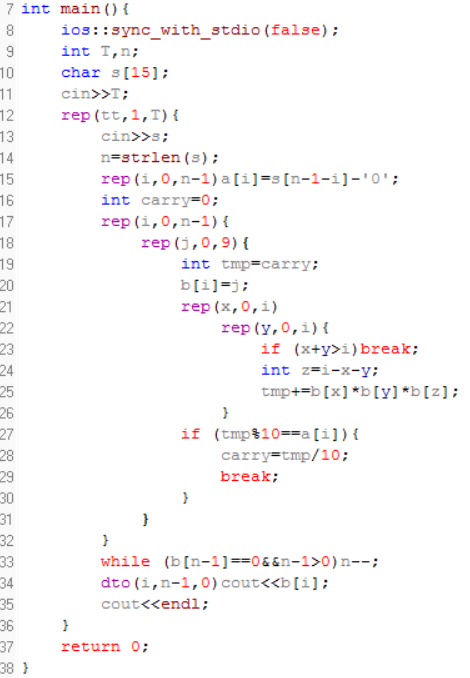
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i | x | y | z | xyz对应位乘积 |
| 0 | 0 | 0 | 0 | 3\*3\*3 |
| 1 | 1 | 0 | 0 | 3\*3\*2 |
| 0 | 1 | 0 | 3\*2\*3 |
| 0 | 0 | 1 | 2\*3\*3 |
| 2 | 2 | 0 | 0 | 3\*3\*1 |
| 1 | 1 | 0 | 3\*2\*2 |
| 0 | 2 | 0 | 3\*1\*3 |
| 1 | 0 | 1 | 2\*3\*2 |
| 0 | 1 | 1 | 2\*2\*3 |
| 0 | 0 | 2 | 1\*3\*3 |

假设现在枚举到第i位(最低位为第0位)，数字为b[i]，那么结合上表我们不难发现，结果的第i位数字，就是所有的b[x]\*b[y]\*b[z]之和再加上低位的进位之后再%10，其中（x+y+z=i）。

有了上述结论，我们只有从小到大枚举每一位的数字，判断是否符合要求，一旦某位找到一个数字符合要求，就退出当前位的枚举而转向下一位数字的枚举，从而保证最后求出的数字是最小的。

**时间复杂度:**O(n^3)

**代码:**

Soj 1119 Factstone Benchmark

**解题思路:**这题很有意思。计算机一开始有4位，每10年位数乘2。也就是说第1960<=x<=2160年的位数len就是4\*2(x-1960)/10=2(x-1960/10)+2。题目要我们求一个最大的n使得n!可以在第x年用len位表示。而len位可以表示的范围为[0,2len-1]也就是有：

n!<2^len

显然如果我们想算出n！与2^len比较是不可行的，因为容易算出len最大为2^22，而2^(2^22)太大了。但是如果我们发现虽然2^len比较大，但是len还是在可接受范围内，因为2^22是四百多万。所以我们对上式两边取对数得

log­2(n!) < len

由于C++标准库中没有以2为底的对数函数只有以10为底的对数函数log，所以我们的等式再变形为：

lg(n!)/lg(2)<len

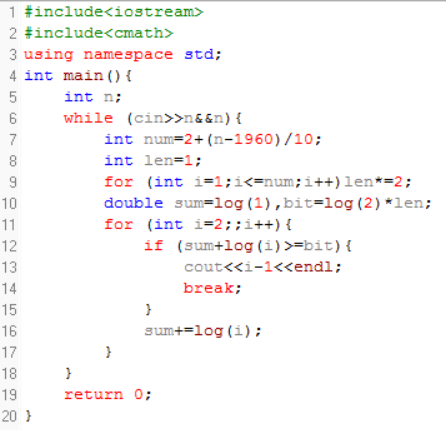
lg(n!)<lg(2)\*len

lg(1)+lg(2)+…+lg(n)<lg(2)\*len

如此就转换成了求一个最大前缀和不超过一个值的简单问题了。只要从小到大枚举n去求即可。

**时间复杂度：**O(n)

**代码:**

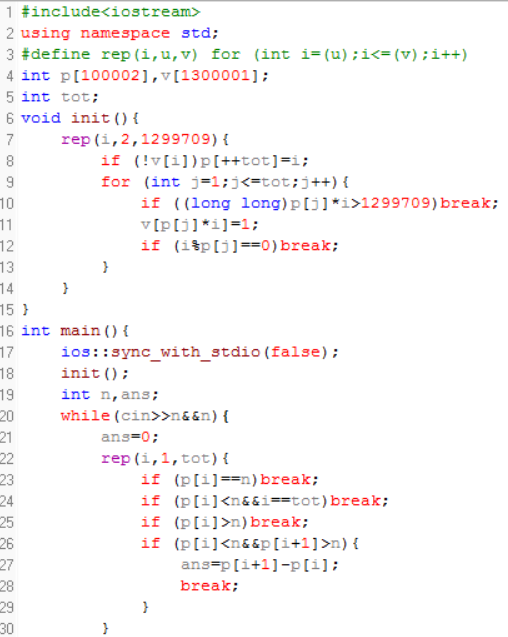


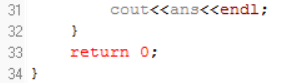
Soj 1500 Prime Gap

**解题思路:**由于数的范围达到百万级别。所以我们选择先使用线性筛素数的方法筛出1-1299709之内的所有素数，共100000个。对给定的每个数，遍历素数数组找到其相邻的两个素数求得它们的差就好。要注意小于数组开头素数和大于数组末尾素数的数以及本身就是素数的数就输出0.

**时间复杂度：**O(n)

**代码:**





**4.程序运行与测试**

8道题均通过成功运行并通过sicily的测试。



# 5.实验总结与心得

这次数论专题仅是接触数论中很基本的知识，题目基本上都与质数有关。在这里我复习了一下如何快速线性筛素数。对于1009这道题，还顺便复习了一下费马小定理以及米勒-拉宾素数测试算法，这两者还是很好用的。但是至于Pollard\_rho大整数质因数分解这个算法则是我第一次接触。它其实是采用了生日悖论去寻找因素。我们知道在[1,1000]中随机取一个数取得x的概率为1/1000，但是如果随机取两个数i和j，它们的差值为x的概率就要大一些。而如果我们从中选取k个数，这k个数中两两差值为x的概率显然又比在[1,1000]中取两个差值为x的数的概率要高。这其实就是生日问题，相当于在一堆人中选两个人，他们生日相同的概率是多少。Pollard\_rho算法不需要盲目地找这样的两个数，只需要一个一个地生成并检查梁旭的两个知道找到我们想要的。在这里又不得不感叹前人真是聪明。

# 附录、提交文件清单

实验报告.pdf

1009.cpp

1020.cpp

1259.cpp

1240.cpp

1231.cpp

1203.cpp

1119.cpp

1500.cpp