Parallel Prime Number Sieve

解题思路：

总的实现方法如下：

```
13  void primeSieve(int number);
14  void changeAllMultiples(vector<bool>& prime, int father_range, int process_number, ProcessPool* pool);
15  void ParentProcessHdl(int total, int process_number, int index, ProcessPool* mypool);
16  void ChildProcessHdl(int total, int process_number, int index, ProcessPool* mypool);
17  void printResult(vector<bool> nums, int index, int low_bound, int start);
```

```
Program Sieve;
const n=100;
var Prime: array [1..n] of boolean;
    i, num, loc: integer;
begin
   for i:=1 to n do
       Prime[i]:=true;
   for num:=2 to $\lfloor\sqrt(n)\rfloor$ do
       if Prime[num] then
          begin
             loc:=num+num
             while loc$\leq$ n do
                 begin
          Prime[loc]:=false;
                    loc:=loc+num;
                 end;
          end;
end.
```

按照老师提示的思路，将总的实现分为$\lceil \sqrt{n}/2 \rceil$个子进程，然后通过总数和进程数目的关系来计算每个进程计算的数字范围。子进程的创建使用 fork 实现，并且通过管道实现进程之间的通信，使用 filedes 数组来控制进程的读写操作。

关于时间，当进程被创建的时候设置一个开始时间，打印结束的时间作为结束时间，从而来得到每个进程的时间，并且计算所有进程执行结束的时间来得到总时间。

实验环境：

Ubuntu 16.04，c++

实现中遇到的问题：

（1）对于 pipe，write，read，fork，close，wait 等函数的使用不熟悉;

（2）对于整个要计算的数字范围要特别注意上下界的问题避免数组边界溢出;

（3）时间打印的时候要进行类型转换。

（4）对于输入的数字的限制，因为如果输入的数字过大的话，电脑会崩掉，所以输入的数

字范围只能是 0 ~ 9

实验结果截图：

（1）输入为-1

```
yunalin@yunalin-X455LD:~/Desktop/primesieve$ ./bin/primeSieve
Please input the number between 0 and 9: -1
Please input the number between 0 and 9: 0
```

（2）输入为 0

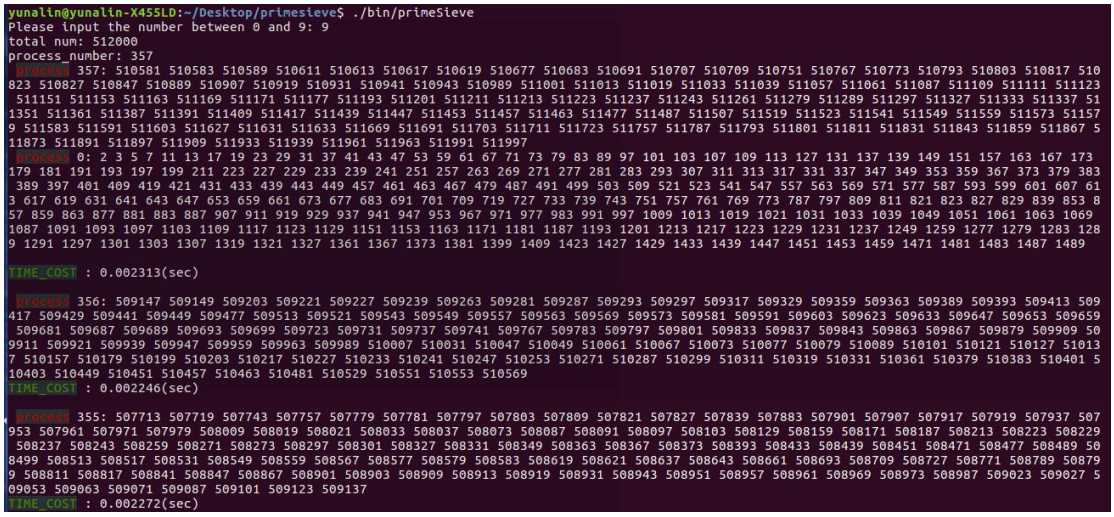
```
yunalin@yunalin-X455LD:~/Desktop/primesieve$ ./bin/primeSieve
Please input the number between 0 and 9: 0
total num: 1000
process_number: 15
 process 0: 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67

 process 15: 941 947 953 967 971 977 983 991 997
TIME_COST : 0.000619(sec)

 process 14: 877 881 883 887 907 911 919 929 937
TIME_COST : 0.000549(sec)

 process 13: 821 823 827 829 839 853 857 859 863
TIME_COST : 0.000532(sec)

 process 12: 757 761 769 773 787 797 809 811
TIME_COST : 0.000538(sec)

 process 11: 691 701 709 719 727 733 739 743 751
TIME_COST : 0.000529(sec)

 process 10: 631 641 643 647 653 659 661 673 677 683
TIME_COST : 0.000518(sec)

 process 9: 569 571 577 587 593 599 601 607 613 617 619
TIME_COST : 0.000542(sec)
```

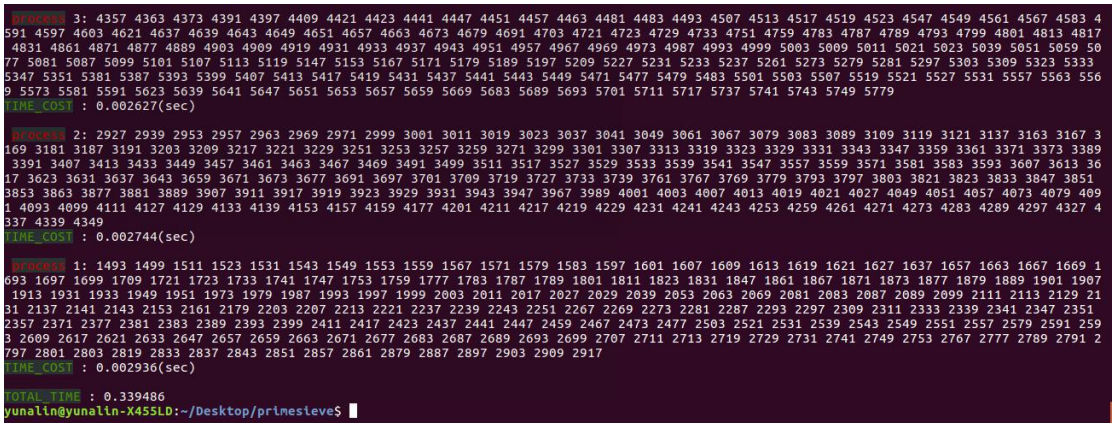
```
 process 8: 509 521 523 541 547 557 563
TIME_COST : 0.000485(sec)

 process 7: 443 449 457 461 463 467 479 487 491 499 503
TIME_COST : 0.000523(sec)

 process 6: 383 389 397 401 409 419 421 431 433 439
TIME_COST : 0.00049(sec)

 process 5: 331 337 347 349 353 359 367 373 379
TIME_COST : 0.000525(sec)

 process 4: 257 263 269 271 277 281 283 293 307 311 313 317
TIME_COST : 0.000475(sec)

 process 3: 197 199 211 223 227 229 233 239 241 251
TIME_COST : 0.000526(sec)

 process 2: 137 139 149 151 157 163 167 173 179 181 191 193
TIME_COST : 0.000507(sec)

 process 1: 71 73 79 83 89 97 101 103 107 109 113 127 131
TIME_COST : 0.000506(sec)

TOTAL_TIME : 0.006788
```

（3）输入为 10

```
yunalin@yunalin-X455LD:~/primesieve$ ./bin/primeSieve
Please input the number between 0 and 9: 10
Please input the number between 0 and 9: 1
```

（4）输入为 9





注：

程序运行方法：（1）进入 primeSieve 文件夹

（2）make

（3）./bin/primeSieve

另外有一个需要说明的问题是：运行系统装在固态硬盘上，所以可能运行的相对来说比较快。