HW1: Parallel Prime Number Sieve

个人信息：

明友芬 15331242

林泽云 15331207

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

```
yunalin@yunalin-X455LD:~/Desktop/primesieve$ ./bin/primeSieve
Please input the number between 0 and 9: 9
total num: 512000
process_number: 357
process 357: 510581 510583 510589 510611 510613 510617 510619 510677 510683 510691 510707 510709 510751 510767 510773 510793 510803 510817 510823 510827 510847 510889 510907 510919 510931 510941 510943 510989 511001 511013 511019 511033 511039 511057 511061 511087 511109 511111 511123 511151 511153 511163 511169 511171 511177 511193 511201 511211 511213 511223 511237 511243 511261 511279 511289 511297 511327 511333 511337 511351 511361 511387 511391 511409 511417 511439 511447 511453 511457 511463 511477 511487 511507 511519 511523 511541 511549 511559 511573 511579 511583 511591 511603 511627 511631 511633 511669 511691 511703 511711 511723 511757 511787 511793 511801 511811 511831 511843 511859 511867 511873 511891 511897 511909 511933 511939 511961 511963 511991 511997
process 0: 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211 223 227 229 233 239 241 251 257 263 269 271 277 281 283 293 307 311 313 317 331 337 347 349 353 359 367 373 379 383 389 397 401 409 419 421 431 433 439 443 449 457 461 463 467 479 487 491 499 503 509 521 523 541 547 557 563 569 571 577 587 593 599 601 607 613 617 619 631 641 643 647 653 659 661 673 677 683 691 701 709 719 727 733 739 743 751 757 761 769 773 787 797 809 811 821 823 827 829 839 853 857 859 863 877 881 883 887 907 911 919 929 937 941 947 953 967 971 977 983 991 997 1009 1013 1019 1021 1031 1033 1039 1049 1051 1061 1063 1069 1087 1091 1093 1097 1103 1109 1117 1123 1129 1151 1153 1163 1171 1181 1187 1193 1201 1213 1217 1223 1229 1231 1237 1249 1259 1277 1279 1283 1289 1291 1297 1301 1303 1307 1319 1321 1327 1361 1367 1373 1381 1399 1409 1423 1427 1429 1433 1439 1447 1451 1453 1459 1471 1481 1483 1487 1489

TIME_COST : 0.002313(sec)

process 356: 509147 509149 509203 509221 509227 509239 509263 509281 509287 509293 509297 509317 509329 509359 509363 509389 509393 509413 509417 509429 509441 509449 509477 509513 509521 509543 509549 509557 509563 509569 509573 509581 509591 509603 509623 509633 509647 509653 509659 509681 509687 509689 509693 509699 509723 509731 509737 509741 509767 509783 509797 509801 509833 509837 509843 509863 509867 509879 509909 509911 509921 509939 509947 509959 509963 509989 510007 510031 510047 510049 510061 510067 510073 510077 510079 510089 510101 510121 510127 510137 510157 510179 510199 510203 510217 510227 510233 510241 510247 510253 510271 510287 510299 510311 510319 510331 510361 510379 510383 510401 510403 510449 510451 510457 510463 510481 510529 510551 510553 510569
TIME_COST : 0.002246(sec)

process 355: 507713 507719 507743 507757 507779 507781 507797 507803 507809 507821 507827 507839 507883 507901 507907 507917 507919 507937 507953 507961 507971 507979 508009 508019 508021 508033 508037 508073 508087 508091 508097 508103 508129 508159 508171 508187 508213 508223 508229 508237 508243 508259 508271 508273 508297 508301 508327 508331 508349 508363 508367 508373 508393 508433 508439 508451 508471 508477 508489 508499 508513 508517 508531 508549 508559 508567 508577 508579 508583 508619 508621 508637 508643 508661 508693 508709 508727 508771 508789 508799 508811 508817 508841 508847 508867 508901 508903 508909 508913 508919 508931 508943 508951 508957 508961 508969 508973 508987 509023 509027 509053 509063 509071 509087 509101 509123 509137
TIME_COST : 0.002272(sec)
```

```
process 3: 4357 4363 4373 4391 4397 4409 4421 4423 4441 4447 4451 4457 4463 4481 4483 4493 4507 4513 4517 4519 4523 4547 4549 4561 4567 4583 4591 4597 4603 4621 4637 4639 4643 4649 4651 4657 4663 4673 4679 4691 4703 4721 4723 4729 4733 4751 4759 4783 4787 4789 4793 4799 4801 4813 4817 4831 4861 4871 4877 4889 4903 4909 4919 4931 4933 4937 4943 4951 4957 4967 4969 4973 4987 4993 4999 5003 5009 5011 5021 5023 5039 5051 5059 5077 5081 5087 5099 5101 5107 5113 5119 5147 5153 5167 5171 5179 5189 5197 5209 5227 5231 5233 5237 5261 5273 5279 5281 5297 5303 5309 5323 5333 5347 5351 5381 5387 5393 5399 5407 5413 5417 5419 5431 5437 5441 5443 5449 5471 5477 5479 5483 5501 5503 5507 5519 5521 5527 5531 5557 5563 5569 5573 5581 5591 5623 5639 5641 5647 5651 5653 5657 5659 5669 5683 5689 5693 5701 5711 5717 5737 5741 5743 5749 5779
TIME_COST : 0.002627(sec)

process 2: 2927 2939 2953 2957 2963 2969 2971 2999 3001 3011 3019 3023 3037 3041 3049 3061 3067 3079 3083 3089 3109 3119 3121 3137 3163 3167 3169 3181 3187 3191 3203 3209 3217 3221 3229 3251 3253 3257 3259 3271 3299 3301 3307 3313 3319 3323 3329 3331 3343 3347 3359 3361 3371 3373 3389 3391 3407 3413 3433 3449 3457 3461 3463 3467 3469 3491 3499 3511 3517 3527 3529 3533 3539 3541 3547 3557 3559 3571 3581 3583 3593 3607 3613 3617 3623 3631 3637 3643 3659 3671 3673 3677 3691 3697 3701 3709 3719 3727 3733 3739 3761 3767 3769 3779 3793 3797 3803 3821 3823 3833 3847 3851 3853 3863 3877 3881 3889 3907 3911 3917 3919 3923 3929 3931 3943 3947 3967 3989 4001 4003 4007 4013 4019 4021 4027 4049 4051 4057 4073 4079 4091 4093 4099 4111 4127 4129 4133 4139 4153 4157 4159 4177 4201 4211 4217 4219 4229 4231 4241 4243 4253 4259 4261 4271 4273 4283 4289 4297 4327 4337 4339 4349
TIME_COST : 0.002744(sec)

process 1: 1493 1499 1511 1523 1531 1543 1549 1553 1559 1567 1571 1579 1583 1597 1601 1607 1609 1613 1619 1621 1627 1637 1657 1663 1667 1669 1693 1697 1699 1709 1721 1723 1733 1741 1747 1753 1759 1777 1783 1787 1789 1801 1811 1823 1831 1847 1861 1867 1871 1873 1877 1879 1889 1901 1907 1913 1931 1933 1949 1951 1973 1979 1987 1993 1997 1999 2003 2011 2017 2027 2029 2039 2053 2063 2069 2081 2083 2087 2089 2099 2111 2113 2129 2131 2137 2141 2143 2153 2161 2179 2203 2207 2213 2221 2237 2239 2243 2251 2267 2269 2273 2281 2287 2293 2297 2309 2311 2333 2339 2341 2347 2351 2357 2371 2377 2381 2383 2389 2393 2399 2411 2417 2423 2437 2441 2447 2459 2467 2473 2477 2503 2521 2531 2539 2543 2549 2551 2557 2579 2591 2593 2609 2617 2621 2633 2647 2657 2659 2663 2671 2677 2683 2687 2689 2693 2699 2707 2711 2713 2719 2729 2731 2741 2749 2753 2767 2777 2789 2791 2797 2801 2803 2819 2833 2837 2843 2851 2857 2861 2879 2887 2897 2903 2909 2917
TIME_COST : 0.002936(sec)

TOTAL_TIME : 0.339486
yunalin@yunalin-X455LD:~/Desktop/primesieve$
```

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。