COM 5335 ASSIGNMENT #1

DUE BY 3AM 11/18/2016 (Mon)

10% penalty applies to 1-day late submissions received between 3:01AM 11/18 and 3:00AM 11/18. No submission will be accepted after 3:01 AM 11/19/2019.

Objective

Implement the Miller-Rabin primality test and the Rabin Public-Key Cryptosystem.

Description

You need to deal with big number arithmetic. You can use software packages/classes for this part or implement in your own way.

There are 2 things you need to do before implementing the Rabin public-key cryptosystem.

- (1) Miller-Rabin primality test
- (2) 256-bit prime number generation

Prime number generation involves pseudorandom number generation. Here you can use any pseudorandom number generating functions provided in any library (such as srand() in <cstdlib>) to do this assignment. Remember, in practice, pseudorandom number generators need to be cryptographically secure and srand() is NOT a good choice.

Miller-Rabin as well as any other primality tests are EXPENSIVE, so it is better to avoid running it as much as possible. You can significantly reduce your program's running time if you DO TRIAL DIVISION FOR SMALL PRIMES. IT IS STRONGLY RECOMMENDED. Below is a list of small primes up to 1000.

3	5	7	11	13	17	19	23	29	31
41	43	47	53	59	61	67	71	73	79
89	97	101	103	107	109	113	127	131	137
149	151	157	163	167	173	179	181	191	193
199	211	223	227	229	233	239	241	251	257
269	271	277	281	283	293	307	311	313	317
337	347	349	353	359	367	373	379	383	389
401	409	419	421	431	433	439	443	449	457
463	467	479	487	491	499	503	509	521	523
547	557	563	569	571	577	587	593	599	601
613	617	619	631	641	643	647	653	659	661
677	683	691	701	709	719	727	733	739	743
757	761	769	773	787	797	809	811	821	823
829	839	853	857	859	863	877	881	883	887
911	919	929	937	941	947	953	967	971	977
991	997								
	41 89 149 199 269 337 401 463 547 613 677 757 829 911	41 43 89 97 149 151 199 211 269 271 337 347 401 409 463 467 547 557 613 617 677 683 757 761 829 839 911 919	41 43 47 89 97 101 149 151 157 199 211 223 269 271 277 337 347 349 401 409 419 463 467 479 547 557 563 613 617 619 677 683 691 757 761 769 829 839 853 911 919 929	41 43 47 53 89 97 101 103 149 151 157 163 199 211 223 227 269 271 277 281 337 347 349 353 401 409 419 421 463 467 479 487 547 557 563 569 613 617 619 631 677 683 691 701 757 761 769 773 829 839 853 857 911 919 929 937	41 43 47 53 59 89 97 101 103 107 149 151 157 163 167 199 211 223 227 229 269 271 277 281 283 337 347 349 353 359 401 409 419 421 431 463 467 479 487 491 547 557 563 569 571 613 617 619 631 641 677 683 691 701 709 757 761 769 773 787 829 839 853 857 859 911 919 929 937 941	41 43 47 53 59 61 89 97 101 103 107 109 149 151 157 163 167 173 199 211 223 227 229 233 269 271 277 281 283 293 337 347 349 353 359 367 401 409 419 421 431 433 463 467 479 487 491 499 547 557 563 569 571 577 613 617 619 631 641 643 677 683 691 701 709 719 757 761 769 773 787 797 829 839 853 857 859 863 911 919 929 937 941 947	41 43 47 53 59 61 67 89 97 101 103 107 109 113 149 151 157 163 167 173 179 199 211 223 227 229 233 239 269 271 277 281 283 293 307 337 347 349 353 359 367 373 401 409 419 421 431 433 439 463 467 479 487 491 499 503 547 557 563 569 571 577 587 613 617 619 631 641 643 647 677 683 691 701 709 719 727 757 761 769 773 787 797 809 829 839 853 857 859 863 877 911 919 929	41 43 47 53 59 61 67 71 89 97 101 103 107 109 113 127 149 151 157 163 167 173 179 181 199 211 223 227 229 233 239 241 269 271 277 281 283 293 307 311 337 347 349 353 359 367 373 379 401 409 419 421 431 433 439 443 463 467 479 487 491 499 503 509 547 557 563 569 571 577 587 593 613 617 619 631 641 643 647 653 677 683 691 701 709 719 727 733 757 761 769 773 787 797 809 811	41 43 47 53 59 61 67 71 73 89 97 101 103 107 109 113 127 131 149 151 157 163 167 173 179 181 191 199 211 223 227 229 233 239 241 251 269 271 277 281 283 293 307 311 313 337 347 349 353 359 367 373 379 383 401 409 419 421 431 433 439 443 449 463 467 479 487 491 499 503 509 521 547 557 563 569 571 577 587 593 599 613 617 619 631 641 643 647 653 659 677 683 691 701 709 719 727

Implement Rabin Cryptosystem. Users are asked to input two 128-bit primes p and q, and a 96-bit plaintext in hex. Do 16-bit repetition padding at the end as described in the class. For decryption, users are asked to input the ciphertext as well as p and q. Sample I/O is shown below.

Output your 256-bit prime number

Sample I/O (Input shown in bold face.)

<Rabin Encryption>

p = daaefe65 2cad1614 f17e87f2 cd80973f

q = f9998862 6723eef2 a54ed484 dfa735c7

n = pq = d5375c87 792a4ac9 135966b6 d1689939 c249ed22 452f77d6 3fa82d67 e95e9cf9

Plaintext: be000bad bebadbad bad00deb deadface deafbeef add00add bed00bed

Ciphertext = 205651dd a3fced3e 74e9c50a 61342e29 b6b8e14e 85ce5666

```
7b341c78 cc2965cb

<Rabin Decryption>
Ciphertext = 5452361a db4c34be 04a5903a e00793bc 1086e887 ebed06e2
3ffba0b4 a4348cc0
Private Key:
p = d5e68b2b 5855059a d1a80dd6 c5dc03eb
q = c96c6afc 57ce0f53 396d3b32 049fe2d3
Plaintext = 00000000 12345678 87654321 12345678 87654321
87654321
```

Grading

Your program MUST BE compatible with Dev C/C++ or GNU C/C++ compilers. If you are using other compilers, please make sure your final program is compatible. You will get no points if your program is not compilable using the abovementioned compilers. If your program is compilable but the result is not completely correct, you'll still get partial credits. Your program should be well-commented, well-structured, and easy to understand. You may lose up to 30% of points if you fail to do so.

Submission

Put all your source codes in a folder containing main functions, function implementations, class definitions, or compilation instructions, if any. Compress them as a single zip file. DO NOT submit executable files. Name your zip file as your student ID number (i.e. 100012345.zip). Submit your source code on iLMS at http://lms.nthu.edu.tw.