Introduction to Computer Science
HW #3
Due: 2016/04/27

Homework Rules:

Hand-written homework can be handed in **before lecture starts**. Otherwise, you may contact the TA in advance and then bring the hardcopy to the TA in BL-603 (please send e-mail in advance).

As for the programming part, you need to upload it to CEIBA before the deadline. The file you upload must be a **.zip** file that contains the following files:

README.txt

HW01_b04901XXX (a folder that contains all .cpp & .h as required),

1. Do not submit executable files (.exe) or objective files (.o, .obj). Files with names in wrong format will not be graded. You must **remove any system calls**, such as system ("pause"), in your code if any.
2. In README.txt, you need to describe which compiler you used in this homework and how to compile it (if it is in a "project" form).
3. In your .cpp files, we suggest you write comments as detailed as you can. If your code does not work properly, code with comments earns you more partial credits.

Chapter 4 Review Problems (6 pts each)

21, 25, 28, 40

Chapter 5 Review Problems (8 pts each)

39, 49, 50, 53

Programming Problem (44%)

First, VERY IMPORTANT: check whether sizeof(unsigned long long int) or sizeof(unsigned long int) is 8. If not, use another computer.

Write two pieces of code:

(a) cipher.cpp reads the file "plain.txt" containing one string (length < 10000) and "public_key.txt" containing *N* and *e*. cipher.cpp should then output "secret.txt" as integers encrypted by RSA. The encoding concatenates 2 chars into one integer. For example, "AB" would be encoded as (65*2^8+66)=16,706. If only one char remains, put it to leftmost. For example, “A” would be encoded as $65*2^8$=16,640.

(b) decipher.cpp reads the file “secret.txt” and “private_key.txt” containing *N* and *d*. decipher.cpp should then output “message.txt” with content same as “plain.txt”.

Note: Be careful about overflow, signed/unsigned, and eof() problem. “Npqphied.txt” contains two more (*N*,*e*,*d*) sets for you to test.

Bonus (5%)

Write the following function:

unsigned long long int findD(**unsigned long long int** e, **unsigned long long int** phi)

The function returns *d*, where *de* ≡ 1 (mod *phi*). Save the function into bonus.cpp. No main().
Note: You need to use Euclidian algorithm. Enumeration won’t earn any credit.