Homework 8

Due: At the beginning of class, Wednesday, December 7.

Problem 1 (40 points)
Using the C or C++ language, and using the Integer class bignum package available with the g++ compiler on umbc8, implement the following two procedures:

- The Miller-Rabin probabilistic primality test.
- An efficient method for solving any system of linear Diophantine equations of the form:

$$\begin{aligned} y &\equiv b_1 \bmod n_1 \\ y &\equiv b_2 \bmod n_2 \\ &\vdots \\ y &\equiv b_k \bmod n_k. \end{aligned} \tag{1}$$

You must use the C or C++ language, and your implementation must handle integers that consist of up to 4096 decimal digits.

Bignums
The *Bignum* data-type represents arbitrarily large integers. Such integers are used extensively in modern cryptology—for example, they are needed to implement the RSA cryptosystem. The usual double-precision integer data type in C is not adequate for dealing with large integers because it cannot represent the very long integers used in cryptography.

Resources
At UMBC there are two bignum packages available: the Integer class that comes with the g++ compiler, and the gmp package which was recently installed. You are free to use any package of your choice; however, if you do not use the g++ package, then do not expect to receive help.

See the CMSC-443 WWW home page for information about the g++ package. In particular, study the file Bignum Information/sample.tex for a detailed example of how to use the g++ package. Note that this package uses operator overloading (you can use many of the usual C operators on bignums).

For more about the Miller-Rabin test, see Chapter 33 in *Introduction to Algorithms* by Cormen, Leiserson, Rivest (the CMSC-441 text).

What to hand in
Hand in each of the following five items:

a) A well-documented listing of all source code that your wrote.

b) Proof that your two procedures work correctly. It is your responsibility to offer convincing empirical proof. Among other examples, solve the system of Diophantine equations given in HW6. Check each example.

c) For any integer $k \geq 1$, the kth *Fermat Number* is the integer $F_k = 2^{2^k} + 1$. Run the Miller-Rabin on each of the first 12 Fermat numbers. According to your test, which are prime? Similarly, test the Carmichael numbers 561, 1105, and 1729.

d) Empirically measure the time it takes to perform integer addition, multiplication, and division as a function of the length (in bits) of the bignums. Draw a graph of your observed running times. Interpret your graphs.

e) Write a brief essay describing your work on this homework assignment. In this essay, you should identify what engineering difficulties you faced, explain how you solved them, and discuss the implications of your engineering choices.

f) *(Optional)* Implement another cryptographic application involving Bignums. For example, you might implement the RSA cryptossytem. Clearly describe and discuss any such additional work.