CS 667 : Homework 3(Due: Mar 14, 2013)

Problems 1-6 are for 200pts. You may replace some of them with Problem 7 or 8 for a total of 200.

Problem 1. (20 points)
This problem assumes a WORD model of computation. The Fibonacci sequence is given by the following recurrence $F_{n+1} = F_n + F_{n-1}$ for $n \geq 1$ and $F_0 = F_1 = 1$.
(a) Show how to compute F_n in $O(n)$ time, and $O(n)$ space.
(b) Show how to compute F_n in $O(n)$ time and $\Theta(1)$ space.
(c) Given a $k \times k$ matrix A show how you can find A^n in $O(k^3 \lg n)$ time.
(d) Can you improve the obvious time bound in (a)/(b)? In particular prove that F_n can be computed in $O(\lg n)$ time.
Hint: You may need to use the result of part (b), i.e. formulate the F_n as a matrix problem. The discussion of problem 31-3 (CLRSE3e) in the Problem section of the Chapter on Number-Theoretic Algorithms might offer you some insight on this.

Problem 2. (20 points)
Choose k out of n (i.e. n choose k): $\binom{n}{k}$ is defined as

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

where $k!$ is defined as $k! = k(k-1)(k-2)\ldots 1$.
Find $\binom{n}{k}$ in $O(n^2)$ additions/subtractions. Multiplications and divisions are not allowed.
If you are not very familiar with the properties of $\binom{n}{k}$, a good reading is section C.1 or 6.1 (new or older version of the textbook) including the exercises. A way to derive the answer is hidden somewhere there.

Problem 3. (40 points)
Consider two sets A and B each having n integers in the range from 0 to $30n$. We wish to compute the Cartesian sum of A and B defined by

$$C = \{x + y : x \in A \ \ and \ \ y \in B\}$$

Note that the integers in C are in the range from 0 to 30n. We want to find the elements of C and the number of times each element of C is realized as a sum of elements in A and B. Suppose that we have a black-box function `AProduct` that takes as input two polynomials of degree m and returns its product in time $O(m \lg m)$. Show how you could use `AProduct` to find C in time $O(n \lg n)$. **Hint.** Represent A and B as polynomials of degree about $15n$.

Problem 4. (40 points)
Somewhere on page 920 (or 844) of Chapter 30 of CLRS3e (or CLRS2e) there is a problem 30-1 that discusses Karatsuba-based polynomial multiplication. Solve part (b) for the only case of upper-half and lower-half. This assumes you know how to solve part (a) as well. We will do part (c) in class.

Problem 5. (40 points)
Evaluating a polynomial A(x) of degree-bound n at a given point c can also be done by dividing $A(x)$ by the polynomial $(x - c)$ to obtain the quotient polynomial $q(x)$ of degree-bound $n - 1$ and a remainder r, such that $A(x) = q(x)(x - c) + r$. Clearly $A(c) = q(c)(c - c) + r = r$. Show how to compute the remainder r and the coefficients of $q(x)$ in time $\Theta(n)$ from c and the coefficients of A.

Problem 6. (40 points)
(a) **Model Times.** Let the cost of multiplying two polynomials of degree a and b be $a \cdot b$. Find a schedule for multiplying the six polynomials listed below that is of the lowest possible total cost. The six polynomials are $f_1, \ldots f_6$ of degrees $1, 2, 3, 2, 4, 5$ respectively and we are interested in finding the product $f = f_1 f_2 f_3 f_4 f_5 f_6$. Prove that the schedule you provide is of the lowest possible cost.
(b) **Model Plus.** Let the cost of multiplying two polynomials of degree a and b be $a+b$ (note the difference from Problem 1). Find a schedule for multiplying the six polynomials listed below that is of the lowest possible total cost. The six polynomials are $f_1, \ldots f_6$ of degrees $1, 2, 3, 2, 4, 5$ respectively and we are interested in finding the product $f = f_1 f_2 f_3 f_4 f_5 f_6$. Prove that the schedule you provide is of the lowest possible cost.

Problem 7. (40 POINTS)
Implement Shamir's secret sharing scheme. The interface below is just for illustrative purposes and to describe the parameters

```
shamirc(secret k, parties p, reconstruct  r, file-out file-name) //Create
  // Returns  a file-name that contains one per-line the individual secrets
  // assigned to each of the p parties. file-name thus has p lines one for each party

secret shamirr(parties p, reconstruct  r, file-in  file-name)  // Reconstruct
  // Uses file-name with at least r lines but no more than p to reconstruct
  // the secret s that is returned.

  // Details are left to you for implementation.
```

The interface will be through the command-line. (Below `shamirc, shamirr` are program names, and in the context of C/C++ not necessarily class names.)

```
% ./shamirc k p r out-file
or
% java shamirc k p r out-file
```

will call the corresponding function and generate some output in file out-file.

```
% ./shamirr p r my-file
or
% java shamirr p r my-file
```

will use my-file (containing lines of out-file) to return in the standard output the secret.
Be careful. Numbers can grow large! The secret is a positive 32-bit integer `int` or `unsigned int`.

Problem 8. (40 POINTS)
Implement traditional and Karatsuba-Ofman polynomial multiplication. The input/output will be two polynomials stored in two different files. The first line contains the degree of the polynomial, say n. The second line contains the number of (non-zero) terms t of the polynomial. If the polynomial is full (no zero term) then should read $t = n + 1$. The following lines contain the terms of the polynomial two integers per row the first the degree of the term and the second, the coefficient. Thus the following file represents polynomial $x^{10} + x - 1$. Among the different variants of Karatsuba, implement the one indicated in Problem 4 of this homework.

```
10
3
10 1
1  1
0 -1
```

```
% ./hw3 karatsuba  file1 file2 file3
% ./hw3 nsquared   file1 file2 file3
or
% java hw3 karatsuba file1 file2 file3
% java hw3 nsquared   file1 file2 file3
or
```

where `file1, file2` contain the two polynomials that will be multiplied and `file3` will store the product in the same format as that described earlier. Algorithm `nsquared` is the traditional quadratic algorithm.
Parameters n and t would/could grow larger than 1,000,000.