Problem Set 2: Due by October 16, 2006

Problem 1. (40 points)

Complex Division z_1/z_2. Let $z_1 = a + ib$ and $z_2 = c + id$ be two complex numbers. You may assume that $z_2 \neq 0$, i.e. $c^2 + d^2 \neq 0$, since otherwise the division $z_1/z_2 = z_1/0$ can not be performed. Complex addition/subtraction and multiplication are defined also at the end of the notes for Subject 3 (page 15). The symbol i is handled as an indeterminate in a polynomial operation except that $i^2 = -1$. Therefore instead of having i^3 in a calculation we simplify it to $i^3 = ii^2 = -i$. The division of two complex numbers is defined as follows.

$$\begin{aligned}\frac{a+ib}{c+id} &= \frac{(a+ib)(c-id)}{(c+id)(c-id)} \\ &= \frac{ac+ibc-iad+bd}{c^2+d^2} \\ &= \frac{ac+bd}{c^2+d^2} + \frac{(bc-ad)i}{c^2+d^2}\end{aligned}$$

Apparently that quotient z_1/z_2 is a complex number z_3 such that $z_3 = \frac{ac+bd}{c^2+d^2} + \frac{(bc-ad)i}{c^2+d^2}$. Thus to compute z_3 from z_1 and z_2 we apparently need 6 multiplications (Ms) and 2 divisions (Ds) for a total of 8 M/Ds. Can you decrease this bound further? Explain.

Problem 2. (40 points)

In class we presented an alternative to the Strassen's method that finds the product of two $n \times n$ matrices A and B using the following sequence of multiplication operations recursively

$$\begin{aligned}m_1 &= (A_{12} - A_{22}) * (B_{21} + B_{22}) \\ m_2 &= (A_{11} + A_{22}) * (B_{11} + B_{22}) \\ m_3 &= (A_{11} - A_{21}) * (B_{11} + B_{12}) \\ m_4 &= (A_{11} + A_{12}) * B_{22} \\ m_5 &= A_{11} * (B_{12} - B_{22}) \\ m_6 &= A_{22} * (B_{21} - B_{11}) \\ m_7 &= (A_{21} + A_{22}) * B_{11}\end{aligned}$$

(where A_{ij} and B_{ij} are the $n/2 \times n/2$ submatrices of A and B), and combining the products m_i's as follows to derive the $C_{11}, C_{12}, C_{21}, C_{22}$ of the result $C = A \times B$.

$$\begin{aligned}C_{11} &= m_1 + m_2 - m_4 + m_6 \\ C_{21} &= m_6 + m_7 \\ C_{12} &= m_4 + m_5 \\ C_{22} &= m_2 - m_3 + m_5 - m_7.\end{aligned}$$

Show that indeed the C_{ij} are such that

$$\begin{aligned}C_{11} &= A_{11}B_{11} + A_{12}B_{21} \\ C_{12} &= A_{11}B_{12} + A_{12}B_{22} \\ C_{21} &= A_{21}B_{11} + A_{22}B_{21} \\ C_{22} &= A_{21}B_{12} + A_{22}B_{22}.\end{aligned}$$

Problem 3. (40 points)

a. What is the largest k such that if you can multiply 3×3 matrices using k multiplications (not assuming commutativity of multiplication), then you can multiply $n \times n$ matrices in time $o(n^{\lg 7})$? What would the running time of this algorithm be?

b. V. Pan has discovered a way of multiplying 68×68 matrices using 132464 multiplications, 70×70 matrices using 143640 multiplications, 72×72 matrices using 155424 multiplications. Which method yields the best asymtotic running time when used in a divide and conquer matrix multiplication algorithm? How does it compare to Strassen's algorithm?

Problem 4. (40 points)

Evaluating a polynomial A(x) of degree-bound n at a given point x_0 can also be done by dividing $A(x)$ by the polynomial $(x-x_0)$ to obtain the quotient polynomial $q(x)$ of degree-bound $n-1$ and a remainder r, such that $A(x) = q(x)(x-x_0)+r$. Clearly $A(x_0) = q(x_0)(x_0 - x_0) + r = r$. Show how to compute the remainder r and the coefficients of $q(x)$ in time $\Theta(n)$ from x_0 and the coefficients of A.

Problem 5. (40 points)

Show that interpolation can be done in $O(n^2)$ time.

Hint: Read Exercise 30.1-5 of CLRS on page 830, and think of Problem 4.

Problem 6. (50 points)

Derive a point value representation for $A^{rev}(x) = \sum_{j=0}^{n-1} a_{n-1-j}x^j$ from a point-value representation for $A(x) = \sum_{j=0}^{n-1} a_j x^j$, assuming that none of the points in the point-value representation of $A(x)$ is 0.

Programming

Problem P1. (100 points)

Implement the Karatshuba-Ofman algorithm. The input/output will be decimal (base-10) integers. Whether you plan to maintain the long integers in binary or not it's up to you.

```
// The following is pseudocode
// Whether you provide an interface in which n3 is an integer, a pointer to an
// integer or a reference it's up to you.
ReadKaratsuba(file digit1, longint d1, int n1);
MultiplyKaratuba(longint d1, int n1,longint d2, int n2, longint d3, int n3);
WriteKaratsuba(longint d1, int n1, file digit1);
```

A file contains on its first line an (in fact two) integers indicating the number of (decimal) digits that will follow. The remaining lines contain the digits. Line breaks, spaces, tab might exist but are ignored. For example

```
10 10
12345 56
789
```

stores integer the 10-digit integer `1234556789`. Note that the length in digits is given twice in the first line. The first integer is indeed the length; the second is to note the decimal-point in the case that the number is not an integer. Thus viewing the input as real number we get `1234556789.` by including a decimal point to the right of the 10-th digits (the second 10 of the first line).

I should be able to test your code from the command line with an operation of the form

```
% ./karatshuba  file1 file2 file3
```

where `file1, file2` contain the two integers that will be multiplied and `file3` will store the product in the same format as that described earlier.

The first function reads a long integer from a file. The second function uses the internal representation and the algorithm presented in class for binary numbers (adapt it as fit) to speed up multiplication faster than $\Theta(n^2)$. The last function prints an internally-stored integer into a file in the standard format described in this assignment. Make sure that in the latter case you never print more that 50 decimal digits per line.

Problem P2. (100 points)

Implement integer inversion of arbitrarily long integers. Input are binary integers stored in a file (see previous problem). For this problem the input is always binary integers not decimal. You might reuse code for multiplication (P1) for this problem. Storing non integers might require the slight modification to the file format of P1; now the second integer hold the decimal point position and might not be equal to the first.

```
ReadBinary(file digit1, longint d1, int n1);
InvertBinary(longint d1, int n1,longint d2, int n2, int decimalpointposition2, int accuracy);
WriteBinary(longint d1, int n1, int decimalpointposition1, file digit1);
VerifyBinary(longint d1, int n1,longint d2, int n2, int decimalpointposition2, file digit3);
```

Function `ReadBinary` reads a binary integer stored in file `digit1`. It behaves similarly to the corresponding function of the previous assignment (except that the former dealt with decimal integers). `InvertBinary` applies Newton's method described in class to find the inverse of the input up to `accuracy` digits of precision. `decimalpointposition2` might or might not be needed for the output stored in `d2` of `n2` bits. You should transform such output into a file through function `WriteBinary`; make sure though where you place the decimal point. Function `VerifyBinary` multiplies the input and the output and stores the result into file `digit3`.

Besides the four functions, I should be able to run your program from the command line as follows.

```
% ./inverse file1  file2 file3
```

where `file1` is being read by a call of `ReadBinary` and inverted by a call to `InvertBinary` with the output of inversion written in `file2` and the verificiation in `file3`.