Design and Analysis of Algorithms 2011
(Home work 4)

October 23, 2011

- Due on, November 4, before 12 noon.
- Late home works will not be accepted.
- Please give precise arguments for all statements that you write.
- Please do not hesitate to contact me if you do not understand the problems.
- Collaboration is discouraged, but not prohibited. It is recommended that you try to solve the problems on your own. You can discuss the questions with your colleagues but you should not copy solutions. Always write down your own answers. If copying is detected that may immediately lead to a grade less than 7. (**This would be followed strictly**)
- Credits would be given to partial solutions also.
- When you write an algorithm, you should briefly discuss the main idea of your algorithm, then write a pseudo code, argue about its correctness and state and prove the running time of your algorithm.
- The answers should be typed or written clearly and a hard copy is to be submitted.

1. [**10 points**] Given two sorted arrays A and B such that all elements in A and B are distinct and each contain n elements. Give an $O(\lg n)$ algorithm to find the median of all the elements present in A and B.

2. [**10 points**] Given an array A with n elements we would like to know whether there is an element x in A which occurs more than $n/2$ times. Give an $O(n)$ algorithm to do this.

3. [**10 points**] Given two sets A, B, we define their sum as

$$A + B = \{a + b : a \in A, b \in B\}.$$

Give a $O(n \log n)$ algorithm whose input is two sets $A, B \subseteq \{1, 2, \ldots, n\}$ and output is the set $A + B$.

4. [**10 points**] Given two arrays A and B of size n and a number c, design an algorithm which decides whether there exists $i, j \in \{1, 2, \ldots, n\}$, such that $A[i] + B[j] = c$. Your algorithm should run in $O(n \log n)$ time.

5. [**20 points**]Consider the following sorting algorithm called StrangeSort:

StrangeSort(A, i, j)

1. **if** $A[i] > A[j]$;
2. exchange $A[i] \leftrightarrow A[j]$
3. **if** $i + 1 \geq j$
4. **return**
5. $k \leftarrow \lfloor (j - i + 1)/3 \rfloor$
6. StrangeSort$(A, i, j - k)$
7. StrangeSort$(A, i + k, j)$
8. StrangeSort$(A, i, j - k)$

   (a) Argue that if $n =$ length$[A]$, then StrangeSort$(A, 1, n)$ correctly sorts the input array $A[1, 2, \ldots, n]$.

   (b) Give a recurrence for the worst case running time of StrangeSort and a tight asymptotic bound on the worst case running time.

6. [**20 points**] We discussed in class that following the FFT algorithm an analogous algorithm can be written to find the inverse DFT. Write the pseudo-code to compute inverse DFT in $\Theta(n \log n)$ time.

7. [**20 points**] This problem illustrates how to do the Fourier transform in modular arithmetic, for example, modulo 7.

   (a) There is a number ω such that all the powers $\omega, \omega^2, \ldots, \omega^6$ are distinct (modulo 7). Find this ω, also show that $\omega + \omega^2 + \ldots + \omega^6 = 0$.

   (b) Find the Fourier transform of the sequence $(0, 1, 1, 1, 5, 2)$ modulo seven. You have to construct an appropriate matrix using the value of ω found above and multiply the given vector with this matrix. Note all additions and multiplications are to be done modulo 7.

   (c) Write down the matrix necessary to perform the inverse Fourier transform. Verify that multiplying with this matrix returns the original sequence.