ECE G205 Fundamentals of Computer Engineering
Fall 2004

Homework 3: Due by Wednesday October 6 2004

- This homework contains 4 problems. They allow you to earn 100 points.
- Show your work, as partial credit can be given. You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. **Be neat**.
- **No late submissions will be accepted.**
- Only homework returned in a 9in $\times$ 12in envelope will be accepted. (If you cannot find such envelope, ask the Instructor.) Please, write your name and the class name (ECE G205) on the envelope (write clearly, please).
- For the first three problems an e-mail to the TA should be sent that contains the code and the executable of a (single) program that implements the solutions to the problems as functions.
- **Remote students** should send an e-mail of the (typed) solutions to the TA independently of whether code is required or not.

Write your name here: _____

- **Problem # 1 [20 points]**. Consider the problem of sorting n integers stored in an array A by first finding the smallest element of A and exchanging it with $A[1]$. Then find the second smallest element of A, and exchange it with $A[2]$. Continue in this manner for the first $n-1$ elements of A. **(a)** Write both pseudo-code and a C++ program for this algorithm. **(b)** What loop invariant does this algorithm maintain? **(c)** Why does it need to run for only the first $n-1$ elements, rather than for all n elements? **(d)** Give the best-case and the worst-case running times of this algorithm in Θ notation.

- **Problem # 2 [40 points]**. Write the pseudo code and implement in C++ an *exponential time* algorithm for sorting an array of n integers, i.e., an algorithm that requires a number of steps which is an exponential function of n.

- **Problem # 3 [20 points]**. Implement Merge-Sort in C++ using the C++ Standard Template Library.

- **Problem # 4 [30 points]**. Let $T_{MS}(n)$ be the worst-case time complexity of Merge Sort. Prove that $T_{MS}(n)$ is nondecreasing. (Hint: use strong induction.)