CSc 220: Algorithms
Homework 2
Due in Class on Thursday September 24

Return the homework written on sheet(s) of paper with your name and CSc220 written at the top of each sheet. Please staple multiple sheets together. Remember that collaboration is allowed, but that you must write the solution on your own. Also you must acknowledge all collaborators and all sources (other than the textbook) in your solutions. Each problem is worth 10 points.

Problem 1: Given a set $A$ of $n$ distinct integers we want to find the median of $A$, i.e. the element $a \in A$ such that

$$|\{x \in A : x \leq a\}| - |\{x \in A : x > a\}| \leq 1$$

(the above is a fancy way to state that $a$ is the element such that $A$ has the same number of elements which are smaller than $a$ and larger than $a$, with the difference of 1 allowed to account for an odd number $n$ of total elements).

- Give a deterministic $\Theta(n \log n)$ algorithm to find the median of $A$ [2pts];
- Give a randomized algorithm that finds the median in expected $O(n)$ time. [8pts]

Later in the class we will show a deterministic $O(n)$ algorithm to find the median, which is described in the textbook. You will not get credit by presenting that algorithm as a solution for this homework.

Problem 2: A different way to randomize Quick-Sort is to use the deterministic version of Quick-Sort over a 'randomized' array, according to the following pseudo-code

```
Permute-Quick-Sort(A)
B ←− Random-Permute(A);
Return Quick-Sort(B)
```

- Under what conditions on the procedure Random-Permute will Permute-Quick-Sort run in $O(n \log n)$ steps? [2pts]
- Consider the following procedure

```
Shift-Permute(A)
n ← |A|;
s ← Random(1, n);
For i = 1 To n
   j ← s + i mod n;
   B[j] ← A[i];
Return B
```

What is the expected running time of Permute-Quick-Sort if you use Shift-Permute in place of Random-Permute? [4pts]

- Give your own implementation of Random-Permute that will make Permute-Quick-Sort run in $O(n \log n)$ steps. [4pts]

Problem 3: Let $a$ and $b$ be two $n$ bit numbers (assume for simplicity that $n$ is a power of 2).

- Describe the "grade school" algorithm to multiply $a$ and $b$ and show that it requires $O(n^2)$ steps; [2pts]
- Describe a divide-and-conquer algorithm with an asymptotically faster running time. [8pts]