Assignment #3
Due on Thursday 2/14/2019

1. In the selection problem discussed in class, we used an algorithm that computes the median of 5 and showed that it works in a worst-case linear time.

   (a) Repeat the problem using the median of 3 and argue that it does not work in linear time.

   (b) Repeat the problem using the median of 7 and show that it works in a linear time.

2. Given two sorted lists \( A[1..n] \) and \( B[1..n] \). We would like to find the median of the union of the two lists. For simplicity, assume that the union of \( A \) and \( B \) does not contain any duplicate items and that \( n \) is a power of 2. The median is the element at position \( n \), i.e., the one that is larger than \( n - 1 \) elements and less than \( n \) elements.

   (a) Propose a naive algorithm that finds the median in \( \Theta(n) \) running time.

   (b) Propose a divide-and-conquer algorithm that finds the median in \( \Theta(\log n) \) running time.

3. Suppose that we have \( n \) tasks to schedule on a computer with a single-core processor where task \( i \) takes \( t_i \) time units to finish. We would like to run all of the \( n \) tasks while minimizing the total waiting time for all tasks. Assuming that the first tasks starts at \( t = 0 \), the waiting time \( w_i \) for task \( i \) is the total time before it is started. For example, if we have three tasks with execution times \( t_1 = 5, t_2 = 3 \), and \( t_3 = 2 \) scheduled to run in the order \( \langle c_1, c_2, c_3 \rangle \), the waiting times are \( w_1 = 0, w_2 = 5 \), and \( w_3 = 5 + 3 = 8 \). If they are scheduled in the order \( \langle c_3, c_2, c_1 \rangle \), the waiting times become \( w_3 = 0, w_2 = 2 \), and \( w_1 = 2 + 3 = 5 \). Propose a greedy algorithm that finds the optimal scheduling for the \( n \) tasks with the minimum waiting time. Prove the optimality of the algorithm and establish its running time.