csce750 — Analysis of Algorithms
Fall 2019 — Homework 03

Assigned: September 19
Due: October 1

This assignment covers material from the lectures on Chapters 6 and 7, in preparation for Quiz 3.

Page 153: Exercises 6.1-1, 6.1-4, 6.1-5

Page 156: Exercises 6.2-1 (list the element comparisons and swaps, in order), 6.2-5, 6.2-6

Page 159: Exercise 6.3-1 (list the element comparisons and swaps, in order)

Page 160: Exercises 6.4-1 (optional, because it repeats some parts of 6.2-1 and 6.3-1.)

Not in textbook: Blondie has an array $A[1, \ldots, n+m]$, in which $n > 0$ of the elements are '0' and the remaining $m > 0$ elements are '1'. He wants to find the index of one of the 0's, and he uses the following randomized algorithm:

```
FINDINDEXOFZERO(A)
  while true do
    i = random integer between 1 and n + m
    if A[i] = 0 then
      return i
    end if
  end while
```

Find a tight asymptotic bound, in terms of n and m on the **worst-case expected** run time of this algorithm. Then **simplify** that bound for the special case in which $n = m$.

Page 180: Exercises 7.3-2 (Write a recurrence for the number of random numbers generated, then solve that recurrence via the substitution method.)

Page 184: Exercises 7.4-1, 7.4-2

Not in textbook: Consider this randomized version of MERGESORT:

```
RANDOMIZEDMERGESORT(A, ℓ, r)
  if ℓ < r then
    m = random integer in the range {ℓ, ..., r − 1}
    RANDOMIZEDMERGESORT(A, ℓ, m)
    RANDOMIZEDMERGESORT(A, m + 1, r)
    MERGE(A, ℓ, m, r)
  end if
```

Recall that ℓ and r are the lower and upper limits of the portion of the array to be sorted. The only change from standard MERGESORT is that m is selected randomly, rather than dividing the array into two equal parts. Write and solve, using any appropriate method, a recurrence for the worst-case expected run time of this algorithm. Is this algorithm an improvement over the standard MERGESORT?