ARIZONA STATE UNIVERSITY

CSE 310, All Sections — **Data Structures and Algorithms** — Fall 2015

Assignment #2

Available Wednesday, 09/16/2015; due electronically Friday, 10/09/2015

This assignment covers Chapters 6 (Heaps), 7 (Quicksort), and 9 (Medians and Order Statistics) of the text [100 marks total].

Submit electronically, before midnight on Friday, 10/09/2015 using the submission link on Blackboard for Assignment #2, a file named `yourFirstName-yourLastName.pdf` containing your solution to this assignment (a .doc or .docx file is also acceptable, but .pdf is preferred).

1. **Heaps.** [10 marks total]
   - (a) What are the minimum and maximum number of elements in a heap of height h? [2 marks]
   - (b) Show that an n-element heap has height $\lfloor \log n \rfloor$. [2 marks]
   - (c) Show that in any subtree of a max-heap, the root of the subtree contains the largest value occurring anywhere in that subtree. [4 marks]
   - (d) Is the sequence $\langle 23, 17, 14, 6, 13, 10, 1, 5, 7, 12 \rangle$ a max-heap? Why or why not? [2 marks]
2. **Heaps.** [10 marks total]
   - (a) Illustrate the operation of MAX-HEAPIFY$(A, 3)$ on the array $A = \langle 27, 17, 3, 16, 13, 10, 1, 5, 7, 12, 4, 8, 9, 10 \rangle$. [2 marks]
   - (b) Illustrate the operation of BUILD-MAX-HEAP(A) on the array $A = \langle 5, 3, 17, 10, 84, 19, 6, 22, 9 \rangle$. [3 marks]
   - (c) You are given a list of numbers for which you need to construct a min-heap. How would you use an algorithm for constructing a max-heap to construct a min-heap? [5 marks]
3. **Heapsort.** [10 marks]
   Argue the correctness of HEAPSORT using the following loop invariant.

   At the start of each iteration of the **for** loop, the subarray $A[1 \ldots i]$ is a max-heap containing the i smallest elements of $A[1 \ldots n]$, and the subarray $A[i+1 \ldots n]$ contains the $n - i$ largest elements of $A[1 \ldots n]$.
4. **Quicksort.** [20 marks total]
   The QUICKSORT algorithm contains two recursive calls to itself. After QUICKSORT calls PARTITION, it recursively sorts the left sugary and then it recursively sorts the right subarray. The second recursive call in QUICKSORT is not really necessary; we can avoid it by using an iterative control structure. This technique, called *tail recursion*, is provided automatically by good compilers. Consider the following version of QUICKSORT, which simulates tail recursion:

Algorithm 1 function TAIL-RECURSIVE-QUICKSORT(A, p, r)

```
1: while $p < r$ do
2:     $q$ = PARTITION$(A, p, r)$
3:     TAIL-RECURSIVE-QUICKSORT$(A, p, q - 1)$
4:     $p = q + 1$
5: end while
```

   - (a) Argue that TAIL-RECURSIVE-QUICKSORT$(A, 1, n)$, where n is the number of elements in array A, correctly sorts the array A. [10 marks]

(b) Compilers usually execute recursive procedures by using a stack that contains pertinent information, including the parameter values, for each recursive call. The information for the most recent call is at the top of the stack, and the information for the initial call is at the bottom. Upon calling a procedure, its information is pushed onto the stack; when it terminates, its information is popped. If we assume that array parameters are represented by pointers, the information for each procedure call on the stack requires $O(1)$ stack space. The *stack depth* is the maximum amount of stack space used at any time during a computation.

Describe a scenario in which TAIL-RECURSIVE-QUICKSORT's stack depth is $\Theta(n)$ on an n element input array. [10 marks]

5. **Partition.** [10 marks]
The colours of the Dutch national flag are red (R), white (W), and blue (B). The Dutch national flag problem is to rearrange an array of n characters R, W, and B so that all the Rs come first, the Ws come next, and the Bs come last. Design a linear in-place algorithm for this problem and show that your algorithm is $O(n)$. *Hint:* Use the idea of the PARTITION algorithm of QUICKSORT.

6. **Select.** [10 marks]
In the algorithm SELECT, the input elements are divided into groups of 5. Will the algorithm work in linear time if they are divided into groups of 7? Argue that SELECT does not run in linear time if groups of 3 are used.

7. **Largest i numbers in sorted order.** [30 marks total]
Given a set of n numbers, we wish to find the i largest in sorted order using a comparison-based algorithm. Find the algorithm that implements each of the following methods with the best asymptotic worst-case running time, and analyze the running times of the algorithms in terms of n and i.

   (a) Sort the numbers, and list the i largest. [10 marks]

   (b) Build a max-priority-queue from the numbers, and call EXTRACT-MAX i times. [10 marks]

   (c) Use an order-statistic algorithm to find the ith largest number, partition around that number, and sort the i largest numbers. [10 marks]