[**Open book and notes**]Show all of your work clearly in the space provided or on the additional page at the end of the exam. If the additional page is used, clearly identify to which exam question it is related. Be sure to **read each problem carefully**. You should answer all 7 questions. Note that the exam is double sided.

1. In lab 1 you implemented two search algorithms.
(a) (10 points) Briefly describe how each algorithm works.

(b) (10 points) What factors affected the degree of difference in search times for the two algorithms. (Points will be awarded based on the insightfulness and depth of your answer.)

2. (10 points) During the running of the procedure RANDOMIZED-QUICKSORT, how many calls are made to the random-number generator RANDOM in the best case? Justify your answer.

3. (5 points) What is the running time of heapsort on an array A of length n that is already sorted in decreasing order? Justify your answer.

4. (15 points) Use mathematical induction to show that the sum of any six consecutive squares leaves a remainder of seven when divided by 12.

5. (10 points) In problem 6.3-2 you were asked "Why do we want the loop index i in line 2 of BUILD-MAX-HEAP to decrease from $\lfloor length[A]/2 \rfloor$ to 1 rather than increase from 1 to $\lfloor length[A]/2 \rfloor$?" Give an example array that will produce an incorrect result if i in line 2 of BUILD-MAX-HEAP[1] increases instead of decreases.

[1] BUILD-MAX-HEAP is on page 133 of the text book.

6. (15 points) Let $f(n)$ and $g(n)$ be asymptotically non-**positive** functions. Consider the following statement:

$-\min(f(n), g(n)) = \Theta(-f(n) - g(n))$

If true, use the basic definition of Θ-notation to prove it. If false, provide a specific example for which the equation fails to be true.

7. 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) (5 points) Sort the numbers, and list the i largest.

(b) (10 points) Build a max-heap from the numbers, and call EXTRACT-MAX i times.

(c) (10 points) Use an order-statistic algorithm to find the i^{th} largest number, partition around that number, and sort the i largest numbers.

Additional work area for any problem. Clearly identify to which problem the work on this page is related.

Additional work area for any problem. Clearly identify to which problem the work on this page is related.