UNIVERSITY OF BRISTOL

May/June 2019 Examination Period

FACULTY OF ENGINEERING

First Year Examination for the Degree of Bachelor and Master of Engineering and Bachelor of Science

COMS-10007
Algorithms

TIME ALLOWED:
2 Hours

This paper contains *three* questions.
All answers will be used for assessment.
The maximum for this paper is *100 marks*.

Other Instructions:

1. **Calculators must have the Faculty of Engineering Seal of Approval.**

TURN OVER ONLY WHEN TOLD TO START WRITING

Important Information: Throughout this exam paper log() denotes the binary logarithm, i.e, $\log(n) = \log_2(n)$, and ln() denotes the logarithm to base e, i.e., $\ln(n) = \log_e(n)$. We also write $\log\log n$ as an abbreviation for $\log(\log(n))$ and $\log^c n$ as an abbreviation for $(\log n)^c$.

Q1. This question is about sorting.

(a) What does it mean for a sorting algorithm to be in-place? Give an example of an in-place sorting algorithm (only mention its name) and an example of a sorting algorithm that is not in-place (only mention its name). *[5 marks]*

(b) What does it mean for a sorting algorithm to be stable? Give an example of a stable sorting algorithm (only mention its name) and an example of a sorting algorithm that is not stable (only mention its name). *[5 marks]*

(c) Suppose that Quicksort is used for sorting an array A of n positive integers. The pivot plays a central role in Quicksort. Consider the following options as a choice for the pivot:

1. The element at position $\lceil n/2 \rceil$ ($\lceil . \rceil$ denotes the ceiling function).
2. The median.

For each option, give the worst-case runtime of Quicksort (no justification needed). *[4 marks]*

(d) In the lecture we proved a $\Omega(n \log n)$ time lower bound for sorting. Why does this not contradict the runtimes of Countingsort and Radixsort? *[5 marks]*

(e) Sort the following numbers using Radixsort:

$$219, 113, 736, 233, 176, 512 \ .$$

Show your working. *[5 marks]*

(f) Heapsort interprets an array as a complete binary tree. Consider the following array:

10	16	4	9	1	5	8	7	6	11

Draw the corresponding complete binary tree. Next, turn the tree into a heap by running Build-Heap(). Give the sequence of node exchanges and draw the resulting heap. *[6 marks]*

Q2. This questions concerns Big-O notation.

(a) Let $g : \mathbb{N} \to \mathbb{N}$ be a function. Define the set $O(g(n))$. *[5 marks]*

(b) State the racetrack principle. *[5 marks]*

(c) Give a formal proof of the statement:

$$5n^2 \in O\left(\frac{1}{10}n^3\right) \ .$$

[5 marks]

(d) Use the racetrack principle to prove the following statement:

$$4\log(n) + 3n \in O(n) \ .$$

Hint: The derivative of $\log n$ is $\frac{1}{\ln(2)n}$ and $0.5 \leq \ln(2) \leq 1$. *[7 marks]*

(e) Order the following sets so that each is a subset of the one that comes after it:

$$O\left(2^{\sqrt{\log n}}\right), O\left(\log^2 n\right), O(n!), O(2^n), O(\log\log n), O(n\log n), O\left(n^8\right) \ .$$

[5 marks]

(f) Give two functions f and g such that:

$$f(n) \in O(g(n)) \text{ and } 2^{f(n)} \notin O(2^{g(n)}) \ .$$

Briefly justify your answer. *[3 marks]*

Q3. This question concerns algorithmic design principles and recurrences.

(a) Describe an efficient algorithm in words (no code or pseudo-code) that finds the largest element in an array of n distinct numbers. What is the worst-case runtime of this algorithm? What is the best-case runtime of this algorithm? *[5 marks]*

(b) What is a divide-and-conquer algorithm? Give an example and briefly explain why it is a divide-and-conquer algorithm.

[7 marks]

(c) What is a dynamic programming algorithm? Give an example and briefly explain why it is a dynamic programming algorithm. *[7 marks]*

(d) Explain the substitution method for proving an upper bound on a recurrence.

[5 marks]

(e) Consider the following sequence defined inductively as follows:

$$P(0) = P(1) = P(2) = P(3) = 1, \text{ and } P(n) = P(n-2) + P(n-4) \text{ for every } n \geq 4 \ .$$

Further, consider the following algorithm for computing $P(n)$:

Algorithm 1 $\textsc{Seq}(n)$

```
Require: Integer n ≥ 0
  if n ≤ 3 then
    return 1
  else
    return SEQ(n − 2) + SEQ(n − 4)
  end if
```

Draw the recursion tree of the call $\textsc{Seq}(11)$. *[5 marks]*

(cont.)

(f) Let $T(n)$ be the number of times the function SEQ() (listed in Algorithm 1) is executed when calling SEQ(n) (including the call to SEQ(n)). Give a recursive definition of $T(n)$. *[5 marks]*

(g) Let $T(n)$ be the function defined in the previous exercise. Show that $T(n) \in O(C^n)$, for some constant C, using the substitution method. Use the guess $T(n) \leq k \cdot C^n - 1$, for constants k, C. Give the smallest constant C so that the previous statement is true and determine a suitable value for k on the way. *[6 marks]*

END OF PAPER