THE UNIVERSITY *of* ADELAIDE

# EXAMINATION FOR THE DEGREE OF BACHELOR AND MASTER OF ENGINEERING Semester 1, 2021

Algorithm & Data Structure Analysis

(COMP SCI 2201/7201)

**Instructions for Candidates:**

- This is an online examination conducted via **MyUni**.
- The **total time** allocated is **120** minutes including reading time, scanning time and uploading time.
- **Note that you need to finish answering, scanning and uploading within the given time**, late penalties will be applied for the late submissions.
- This is an **open** book examination, student **must not communicate with others during the exam**.
- It is permitted to check stackflow for existing discussions but posting questions is *NOT* permitted.
- Attempt **all six** questions.
- Marks for each question are **shown in brackets**.
- Begin each answer on **a new page**.
- **Answers to questions should be expressed clearly and written legibly. These aspects of presentation will be taken into account in assessment.**
- You need to clearly write your **name and student ID** at the top of the first page while scanning/photographing. A scan or a photograph of your solutions is acceptable, as long as they are **clearly readable**. The workings should be uploaded as a single document in **PDF** format.
- If you experience a problem and you are not sure what to do, contact the Online Exams Call Centre on **+61 8 8313 3311**.

**Question 1 begins on page 2.**

## Q1 Complexity Proof (10 x 2 = 20 marks)

- Given $f(n) \in \Theta(n)$, prove that $f(n) \in O(n^2)$.
- Given $f(n) \in O(n)$ and $g(n) \in O(n^2)$, prove that $f(n)g(n) \in O(n^3)$.

## Q2 Sorting Algorithm (10 marks)

You are asked to sort $n$ English words alphabetically. What is the best achievable complexity? (Note: we are talking about actual English words that can be found in a Marriam-Webster dictionary.) Describe the algorithm that achieves the best complexity.

## Q3 Binary Search Tree (30 marks)

Suppose we need a data container for storing multiple distinct integers, and we need to frequently run the following operations:

- searching, insertion, and deletion
- selection: identifying the i-th smallest integer

We decide to implement a binary search tree (just a normal binary search tree, not AVL), as binary search tree's average performance is quite good for searching, insertion, and deletion. Then, to make selection more efficient, we decide that our tree nodes will store three pieces of information besides the left and right pointers (the value itself, plus two additional pieces of book-keeping information):

- The integer value that we need to store
- How many nodes are in the left child branch
- How many nodes are in the right child branch

We call our new data structure *ADSA Binary Search Tree (ABST)*. Now you are asked to fill in the implementation details. We are looking for algorithms that are efficient and elegant. This is an open ended question without a specific standard answer.

- Given an ABST, how to find the i-th largest integer? Please provide pseudocode. (5 marks for correctness and 5 marks for efficiency and elegance.)
- How to perform insertion to an ABST? Please provide pseudocode. Note that you need to maintain the correctness of all book-keeping information, *i.e.*, for every node, you need to update how many nodes are in the child branches. (5 marks for correctness and 5 marks for efficiency and elegance.)

- How to perform deletion from an ABST? To make this problem easier, you only need to consider the case where the node to be deleted has only one child branch. Please provide pseudocode. (5 marks for correctness and 5 marks for efficiency and elegance.)

## Q4 Graph Algorithm (20 marks)

Given an unweighted undirected connected graph $G$, you are asked to write pseudocode to determine whether $G$ contains cycles. If you need to refer to an existing algorithm, then you need to also include that algorithm's pseudocode in your solution.

## Q5 Undecidable Problems (20 marks)

On our lecture slides, we mentioned two undecidable problems:

- Hello world: Does program Q, given input y, prints "Hello world" as its first output?
- Calls-foo: Does program Q, given input y, ever calls function foo()?

We define a new problem called **Calls-foo-twice**:

- Calls-foo-twice: Does program Q, given input y, calls function foo() at least twice?

Use reduction to prove that calls-foo-twice is also undecidable.

## Q6 NP-Completeness Proof (20 marks)

We know that the partitioning problem is NP-Complete. The partitioning problem is defined as follows:

**Partitioning Problem:** Given $n$ integers $x_1, x_2, \ldots, x_n$, determine whether it is possible to partition these $n$ integers into two disjoint sets so that the sums of these two sets are the same.
Formally, you need to determine whether $L$ and $R$ exist so that

$$L \cap R = \emptyset$$

$$L \cup R = \{1, 2, \ldots, n\}$$

$$\sum_{i \in L} x_i = \sum_{i \in R} x_i$$

For example, if there are five integers $\{1, 2, 2, 4, 5\}$, then we can partition them into $\{1, 2, 4\}$ and $\{2, 5\}$ (both sets sum to 7).

Your task is to use the fact that the partitioning problem is NP-Complete to prove that the three-way partitioning problem is also NP-Complete. The three-way partitioning problem is as follows:

**Three-way Partitioning Problem:** Given $n$ integers $x_1, x_2, \ldots, x_n$, determine whether it is possible to partition these $n$ integers into **three** disjoint sets so that the sums of these three sets are the same.
Formally, you need to determine whether $L$, $R$, $S$ exist so that

$$L \cap R = \emptyset; L \cap S = \emptyset; S \cap R = \emptyset$$

$$L \cup R \cup S = \{1, 2, \ldots, n\}$$

$$\sum_{i \in L} x_i = \sum_{i \in R} x_i = \sum_{i \in S} x_i$$

## End of Exam