School of Computing and Information Systems

COMP10002 Foundations of Algorithms

Sample Mid-Semester Test #3

Student Number:

Time Allowed: Thirty minutes.

Authorised Materials: None.

Instructions to Students: This paper counts for 10% of your final grade. All questions should be answered in the spaces provided on the test paper.

And yes, there will be more lines in each set of boxes in the real test.

Question 1 (7 marks).

The *mode* of a set of numbers is the element that occurs most frequently. For example, the mode of the numbers $[4, 2, 3, 7, 4, 5, 5, 2, 4]$ is the value 4. For the purposes of this question, if there are two values of equal maximum frequency, then the mode is not defined.

Write a function

```
int compute_mode(int A[], int n, int *mode)
```

that processes the `n` values stored in array `A` as follows:

(a) If there is any element in `A` that is a clear majority, the function should provide that value to the calling context using the pointer `*mode`, and the function return value should be the constant `MODE_DEFINED`;

(b) If there is no unique majority element in `A` (that is, if there is more than one distinct value that has the same maximum occurrence frequency), the predefined value `MODE_UNDEFINED` should be returned, and the pointer `*mode` should not be used. For example, on the array $[4, 2, 3, 7, 4, 5, 5, 2, 4, 5]$, there is no mode.

(c) The value `MODE_UNDEFINED` should also be returned if `n` $= 0$.

If you feel that decomposition of the task is appropriate, you may write and call other helper functions too. You do not need to define `MODE_DEFINED` and `MODE_UNDEFINED`; you may assume that they already have values.

You may not alter the array `A` in any way.

(Yes, you'll be given more lines than this in the actual test paper.)

Question 2 (3 marks).

Consider the following functions:

$$f_1(n) = 3n^2 + 10n\sqrt{n}$$

$$f_2(n) = 2n^2 \log\log n$$

$$f_3(n) = 2n^2 \log n + (\log n)^2$$

Using the *best* representative function in each case, categorize the asymptotic growth rate of the following combinations using the "big-Oh" notation:

(a) $f_1(n) + f_2(n)$

(b) $f_1(n) \times \sqrt{n} + f_2(n) \times \log n$

(c) $f_1(n) \times \log n - 1.5 \times f_3(n)$

(d) $f_3(n)/f_2(n)$

(e) Now add the two words "worst" and "best" to exactly two of boxes (a)–(d) to indicate, respectively, the asymptotic growth rate that is the greatest (that is, least desirable for an algorithm to possess), and the asymptotic growth rate that is the slowest (that is, most desirable for an algorithm). Each of the two words should be used exactly once.