FIT1008 – Intro to Computer Science
Tutorial 5

Semester 1, 2017

Objectives of this tutorial

- To understand Big O notation.
- To be able to find the best and worst case time complexity for simple algorithms.
- To be able to determine whether a sorting method is stable.

Exercise 1

Write a version of bubble sort in Python which stops as soon as the list is sorted. Give the best and worst case time complexity for your algorithm, and explain why.

Exercise 2

Write a version of bubble sort that alternates left-to-right and right-to-left passes through the list. This algorithm is called *shaker sort*.

- What is best and worst case time complexity.
- Is this sorting method stable?

Exercise 3

(a) Write a recursive method for computing the sum of the digits of a number. For example, for number 979853562951413, the sum of its digits is $9+7+9+8+5+3+5+6+2+9+5+1+4+1+3=77$. To do this you can use integer division by 10 (//10) which returns an integer with the same digits except the last one, and reminder by 10 (% 10), which returns the last digit.

(b) Determine its complexity, in Big-O notation.

Definition: The *digital root* of a decimal integer is obtained by adding up its digits, and then doing the same to *that* number, and so on,

until you get a single digit, which is the digital root of the number you started with.
For example, to find the digital root of 979853562951413, we calculate:
sum of digits =
$9+7+9+8+5+3+5+6+2+9+5+1+4+1+3 = 77$, then
sum of digits = $7+7 = 14$, then sum of digits = $1+4 = 5$. Now we have just one digit, 5, so that's the digital root of the number we started with.

(c) Write a recursive method to compute the digital root of a positive integer.

(d) Determine its complexity, in Big-O notation.