

# FIT2004: Tutorial 1 (held in Week 3)

**Objectives:** The tutorials, in general, give practice in problem solving, in analysis of algorithms and data-structures, and in mathematics and logic useful in the above.

**Instructions to the class:** Prepare your answers to the questions **before** the tutorial! It will probably not be possible to cover all questions unless the class has prepared them in advance. There is 0.5 mark worth for this Tute towards active participation. 0.25 marks is towards answering the starred questions (\*) indicated below, before attending your assigned tutorial. You will have to hand in your work on these starred questions to your tutor at the very start of the tutorial. Remaining 0.25 mark is for participating during the rest of the tutorial.

**Instructions to Tutors:**

- i. The purpose of the tutorials is not to solve the practical exercises!
- ii. The purpose is to check answers, and to discuss particular sticking points, not to simply make answers available.

1. Use mathematical induction to prove the following:

$$\sum_{i=0}^N ar^i = a + ar + ar^2 + ar^3 + \dots + ar^N = \frac{a(r^{N+1} - 1)}{r - 1}$$

2. \* Listed below is the abstract data type definition for a binary-tree data structure (similar to what's been introduced in the lecture 1.2):

```
type tree e = nilTree | fork e * (tree e) * (tree e)
```

Further, below are definitions of two algorithm on this data structure:

M applicable to trees of any kind, defined as:

```
M(nilTree) = nilTree
| M(fork(e,L,R)) = fork(e, M(R), M(L))
```

SUM on trees of numbers (integers or floating-point):

```
SUM(nilTree) = 0
| SUM(fork(e,L,R)) = e + SUM(L) + SUM(R)
```

Your task in this tutorial is prove some properties of the algorithms:

- (a) Prove **formally** that  $M(M(T)) = T$ , for every tree  $T$ .
- (b) Prove **formally** that  $SUM(M(T)) = SUM(T)$ , for every tree  $T$  of numbers.

3. \* Solve the following recurrence relationship:

$$T(N) = \begin{cases} 2 * T(N-1) + a, & \text{if } N > 0 \\ b & \text{if } N = 0. \end{cases}$$

4. \* Show that the following recurrence relation has a solution  $T(N) = b + c \log_2 N$ .

$$T(N) = \begin{cases} T(N/2) + c, & \text{if } N > 0 \\ b & \text{if } N = 1. \end{cases}$$

5. Let  $F(n)$  denote the  $n$ -th number in the Fibonacci series. Prove the following by induction.

$$\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n = \begin{bmatrix} F(n+1) & F(n) \\ F(n) & F(n-1) \end{bmatrix}$$

6. Let  $F(n)$  denote the  $n$ -th number in the Fibonacci series. Prove that we can compute  $F(2k)$  and  $F(2k+1)$  using  $F(k)$  and  $F(k+1)$  as follows.

$$F(2k) = F(k)[2F(k+1) - F(k)] \tag{1}$$

$$F(2k+1) = F(k+1)^2 + F(k)^2 \tag{2}$$

-END-