FIT2004: Tutorial 2 (held in Week 5)
Covering concepts from Weeks 3 & 4

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 1 mark worth for this Tute towards **active** participation. 0.5 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 0.5 mark is for participating during the rest of the tutorial.

Instructions to Tutors:

1. The purpose of the tutorials is not to solve the practical exercises!

2. The purpose is to check answers, and to discuss particular sticking points, not to simply make answers available.

1. Derive the average case complexity of the quick select algorithm.

*2. The Fibonacci sequence is usually defined as follows:

```
fibonacci(1) = fibonacci(2) = 1
fibonacci(n) = fibonacci(n-1) + fibonacci(n-2),  where n>2
```

It is easy to write (**a highly inefficient**) recursive program to compute the nth number in the Fibonacci sequence. For instance:

```
1 fibonacci(n)
2 {
3  if( n <= 2 ) return 1;
4  else return fibonacci(n-1) + fibonacci(n-2);
5 }
```

- What is the computational complexity of this algorithm? Attempt to prove it formally!

- Can you write a more efficient version that is NOT iterative, but instead single-recursive (rather than double-recursive as in the version above)?

- What is the time complexity of such a single-recursive implementation?

∗ **3.** Given the following algorithm:

```
1     for i from Lo1 to Hi1 do
2        for j from Lo2 to Hi2 do
3            body()
4        end_for
5     end_for
```

How many times is **body()** executed for the following values

```
Lo1=1, Hi1=10, Lo2=i, Hi2=10,
Lo1=0, Hi1= 9, Lo2=0, Hi2= 9,
Lo1=1, Hi1=n, Lo2=i-2, Hi2=i+2,
Lo1=0, Hi1=n, Lo2=i, Hi2=2*i ?
```

∗ **4.** Given the following algorithm how many times are

a) **base()**

b) **body()**

executed for the calls, **r(10)**, **r(5)**, and **r(1)**?

```
1     function r(n)
2      { if( n > 0 )
3         { body();
4            r(n-1);
5         }
6       else
7            base();
8      }
```

Why?

5. A country issues N coins whose denominations (in the increasing order of worth) are $\{D_1, D_2, \cdots D_N\}$. The lowest denomination is $D_1 = 1$. Given a value of V, construct a dynamic programming recurrence to find the **minimum number of coins** that add up to the stated value of V. For example, if the coins denominations are $\{1, 5, 6, 9\}$ and value $V = 12$, the minimum number of coins required is two (of value 6 each).

-=o0o=-
END
-=o0o=-

2