

The Australian National University - Faculty of Engineering and Information Technology

## COMP1200 - Perspectives on Computing - 2008

### Tutorial 2 (Week 8 (w/c Monday 28/04/08))

Preparation: You should attempt ALL 6 questions before the tutorial and bring to the tutorial class your written answers. Tutorial solutions should not be posted on the message board.

- What is the ASCII bit pattern for the character 'c'?
  - What is the ASCII bit pattern for the character 'C'?
  - What is the difference between them?
  - What is the difference in the ASCII bit pattern between the character '4' and the character '\$' or the character '5' and the character '%'?
  - Why do you think that this is the case?
- Consider the following recursive function:

$$f(1) = 1; \quad f(2) = 1; \quad f(3) = 1; \quad f(4) = 3; \quad f(5) = 5;$$

$$f(n) = f(n - 1) + 3 * f(n - 5);$$

Evaluate  $f(6)$ ,  $f(7)$ ,  $f(12)$  and  $f(15)$ .

- Give a *brief* proof of the following fact:

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}.$$

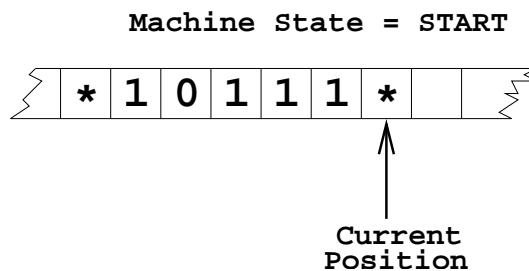
- Draw the parse tree for the expression

$$x \times y + x + z$$

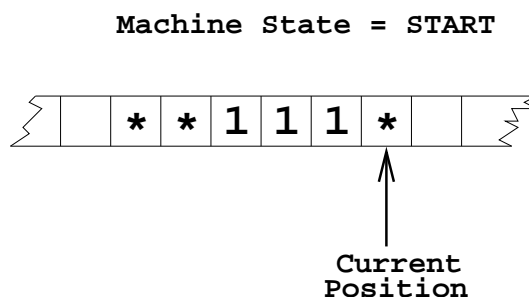
based on the syntax diagrams in Chapter 6.4 of the textbook (or equivalently slide 23 of lecture 6.1).

5. Refer to the instruction set of the Turing machine for incrementing a value given in lectures and in Chapter 11.2 of the textbook:

(a) What instructions are performed when the input tape looks like this:



(b) What instructions are performed when the input tape looks like this:



For both (a) and (b), after each instruction has been executed, indicate (i) the contents of each cell; (ii) the cell that is the current position; (iii) the *state* that the machine is in.

6. (a) Design an algorithm, illustrating it with pseudo-code, for the following problem:

Given two strings of characters,  $S_1$  and  $S_2$ , where the length of  $S_1$  is  $n$  and the length of  $S_2$  is  $O(n)$ , what is the first position in  $S_2$  that contains a character from  $S_1$  (answer should be zero if none of the characters in  $S_2$  appear in  $S_1$ ).

For example, if  $S_1$  was

This.is.easy!

and  $S_2$  was

Or-is-it?

then the result should be 4 (the first occurrence of a character in  $S_2$  that appears in  $S_1$  is the first "i" in position "4").

- (b) What is the worst-case time complexity (using Big-Oh notation) of your algorithm?
- (c) Is your algorithm optimal? Justify your answer.