Computer Science CSC 238H — Wednesday June 10 1998
St. George Campus — University of Toronto

Homework Assignment #2
Due: Wednesday July 8 1998 at 6:10pm

Don't forget to attach the completed COVER PAGE *to the* front *of your assignment!*

1. [7 marks] Consider the following argument, adapted from a logic exercise by Lewis Carroll (Rev. C. L. Dodgson).
Premises:
1. No hedgehogs can read.
2. No one subscribes to the *Times* of London unless he is well-educated.
3. Malcolm, who is a hedgehog, subscribes to the *Times*.
4. Those who cannot read are not well-educated.

Conclusion:
Malcolm cannot waltz.

Determine whether this argument is valid (*i.e.* whether the conclusion is true whenever all of the premises are true). If so, give an informal proof in English, and explain which proof techniques you are using. If not, describe an interpretation which makes the premises true but the conclusion false. The interpretation should describe your domain and which elements of the domain are hedgehogs, which elements are well-educated, etc.

2. [18 marks] Let β and ψ be two *propositional* formulae. Assume all symbols used in β and ψ are from the set $\{$B, C, D, E, H, I, K, O, X, $\sim, \vee, \wedge, \bot, \top, (,)\}$. Let $\scalebox{1}[-1]{$\beta$}$ and $\scalebox{1}[-1]{$\psi$}$ represent the formulae obtained by reflecting β and ψ in a horizontal axis and then interpreting $\backsim$ as $\sim$.

For example, if β is the formula $\sim \mathrm{H} \vee (\mathrm{X} \wedge \mathrm{E}) \vee \bot$,

then $\scalebox{1}[-1]{$\beta$}$ is the formula $\sim \mathrm{H} \wedge (\mathrm{X} \vee \mathrm{E}) \wedge \top$.

Prove that if $\beta \Leftrightarrow \psi$, then $\scalebox{1}[-1]{$\beta$} \Leftrightarrow \scalebox{1}[-1]{ψ}$.

(This should help you to remember the equivalences on page 10 of the textbook; in most cases they come in pairs where one equivalence in the pair can be obtained by reflecting the other.)

Hint: Consider the truth table for the example formula above:

H	X	E	$\sim \mathrm{H} \vee (\mathrm{X} \wedge \mathrm{E}) \vee \bot$
$\bot$	$\bot$	$\bot$	$\top$
$\bot$	$\bot$	$\top$	$\top$
$\bot$	$\top$	$\bot$	$\top$
$\bot$	$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$	$\bot$
$\top$	$\bot$	$\top$	$\bot$
$\top$	$\top$	$\bot$	$\bot$
$\top$	$\top$	$\top$	$\top$
H	X	E	$\sim \mathrm{H} \vee (\mathrm{X} \wedge \mathrm{E}) \vee \bot$

Now let's reflect the whole truth table in a horizontal axis:

H	X	E	$\sim$H $\wedge$ (X $\vee$ E) $\wedge$ $\top$
$\bot$	$\bot$	$\bot$	$\bot$
$\bot$	$\bot$	$\top$	$\top$
$\bot$	$\top$	$\bot$	$\top$
$\bot$	$\top$	$\top$	$\top$
$\top$	$\bot$	$\bot$	$\bot$
$\top$	$\bot$	$\top$	$\bot$
$\top$	$\top$	$\bot$	$\bot$
$\top$	$\top$	$\top$	$\bot$
H	X	E	$\sim$H $\wedge$ (X $\vee$ E) $\wedge$ $\top$

Notice that the second table is still a correct truth table! (Check this yourself.) Use this observation to formulate a lemma that relates the value of any formula β under a particular tva to the value of the formula $\reflectbox{$\beta$}$ under the *opposite* tva. Two tva's are opposite if each propositional variable gets opposite values in the two tva's. Prove your lemma by induction, and then use it to solve this problem.

3. [6 marks] Show that $\sqrt[3]{n}\log_e n \in O(\sqrt{n})$. You may use the following fact from first-year calculus without proof: $\forall c[c > 0 \rightarrow \exists x_0 \forall x(x > x_0 \rightarrow \log_e x < cx)]$.

4. [14 marks] Let $f(n) = \sum_{i=1}^{n}(\log_2 i)^2$. Let $g(n) = \sum_{i=1}^{n}(10\log_3(i^2))$. Answer yes or no to each of the following. Justify your answers.

(a) Is $f \in O(g)$?

(b) Is $f \in \Omega(g)$?

(c) Is $g \in \Theta(f)$?

5. [30 marks] You are given an array $A[1..n]$ sorted into non-decreasing order. Each element in the array is from the set $S = \{1, 2, \ldots, n+1\}$.

[10] **(a)** Design an algorithm that returns an element of S that does *not* appear in the array A. (Do not give a recursive algorithm.) Your algorithm should run in $O(\log n)$ time.

[5] **(b)** State a loop invariant for your algorithm.

[9] **(c)** Prove that your answer to part (b) is an invariant.

[2] **(d)** Use your invariant to prove that your algorithm correctly solves the problem.

[4] **(e)** Prove that your algorithm runs in $O(\log n)$ time. What are the elementary steps that you are counting when you do your analysis of the running time?

6. [25 marks] The following block of pseudocode computes the (integer part of) the square root of x. When reading the code, it may help to think of the numbers being represented in binary. The code fills in one bit of the square root at a time, starting from the most significant bit and working towards the least significant bit. Each iteration of the loop computes one bit of the answer. All variables in the code are integer variables.

```
sqrt(x) % precondition: x ∈ ℤ⁺
    n ←number of bits in binary representation of x
    y ← 0
    y2 ← 0
    t ← 2^(⌊n/2⌋ + 1)
    u ← 2^(2 ⌊n/2⌋)
    do while t > 1
(*)     q ← y2 + y · t + u
        t ← ⌊t/2⌋
        if x ≥ q then
            y ← y + t
            y2 ← q
        end if
        u ← ⌊u/4⌋
    end loop
    return y % postcondition: y = ⌊√x⌋
end sqrt
```

[3] **(a)** Perform a trace of the algorithm for the input $x = 500$. *I.e.*, write down a table that gives the values of the variables $y, y2, t, u$ and q after line (*) is executed in each iteration of the loop.

[5] **(b)** Give a tight bound on the running time of the algorithm, assuming that each line of code can be executed in constant time. Use big-Θ notation to give the bound in terms of n, the length of the input.

[2] **(c)** Write down a *simple* equation that gives the value of $y2$ in terms of the other variables. Your equation should be true at the beginning of each iteration of the loop.

[10] **(d)** Show that the equation you wrote down in part (c), as well as the following statements are true at the beginning of each iteration of the loop:
$u = t^2/4$
$y^2 \leq x < (y+t)^2$
t is a power of 2.

[2] **(e)** Show that the postcondition is satisfied when the loop terminates.

[3] **(f)** What would the running time of the algorithm be if the cost of each addition was $\Theta(n)$ instead of $\Theta(1)$? Assume all the other operations still require constant time. Use big-Θ notation to express your answer.

Remark: Assuming all the numbers are represented in binary notation, this algorithm can be efficiently implemented using only addition and elementary bit operations. Even binary multiplication and division are not really needed, since one can multiply or divide a number by a power of two simply by shifting the bits of the number to the left or to the right.