The University of Nottingham

SCHOOL OF COMPUTER SCIENCE

A LEVEL 2 MODULE, SPRING SEMESTER 2008-2009

DERIVATION OF ALGORITHMS

Time allowed TWO hours

Candidates must NOT start writing their answers until told to do so

Answer FOUR out of six questions

Marks available for sections of questions are shown in brackets in the right-hand margin.

No calculators are permitted in this examination.

Dictionaries are not allowed with one exception. Those whose first language is not English may use a standard translation dictionary to translate between that language and English provided that neither language is the subject of this examination. Subject specific translation dictionaries are not permitted.

No electronic devices capable of storing and retrieving text, including electronic dictionaries, may be used.

DO NOT turn examination paper over until instructed to do so

Question 1:

Let E be a binary relation symbol.
Is the following proposition a tautology?

$$\begin{aligned}&\forall x, \forall y, (\mathsf{E}(x,y) \vee \mathsf{E}(y,x))\\ &\quad \wedge \exists x, \forall y, (\mathsf{E}(y,x) \rightarrow y = x)\\ &\quad\quad \rightarrow \exists x, \forall y, (\mathsf{E}(x,y) \rightarrow y = x)\end{aligned}$$

(a) Write a natural language translation of the two premises and the conclusion of the proposition, when the domain consists of the nodes of a graph and the atomic proposition $\mathsf{E}(x, y)$ is interpreted as *"there is an edge from node x to node y"*. (10)

(b) If the proposition is a tautology, give a full derivation of it in Fitch-style natural deduction.
If it is not a tautology, give a graph countermodel of it. Write the truth table for E in your countermodel. (15)

Question 2:

(a) Compute the weakest precondition P of the following program with respect to the given postcondition:

$$\{P\} \left|\begin{array}{l} \mathsf{z} := \mathsf{x} - \mathsf{y};\\ \textsf{if } \mathsf{z} < 0 \textsf{ then}\\ \quad \mathsf{x} := \mathsf{y}\\ \textsf{else}\\ \quad \mathsf{x} := 2 \cdot \mathsf{y} - \mathsf{x}\\ ;\\ \mathsf{y} := \mathsf{z} + \mathsf{x} \end{array}\right| \{\mathsf{y} = \min(x_0, y_0)\}$$

(15)

(b) Prove that the weakest precondition P that you found is implied by the precondition $\{\mathsf{x} = x_0 \wedge \mathsf{y} = y_0\}$. (5)

(c) Does the specification still hold if we use the following precondition?

$$\{\mathsf{x} = x_0 + 1 \wedge \mathsf{y} = x_0 \wedge y_0 \geq x_0\}$$

If it does, prove it; if it doesn't, choose values for x_0 and y_0 that make it false. (5)

Question 3:

(a) Compute the weakest precondition P of the following program fragment with respect to the given postcondition (even is the boolean function on integers returning true on even arguments and false on odd ones) :

$$\begin{array}{l|l|l} \{P\} & \begin{array}{l}\mathsf{if\ even(n)\ then\ (}\\ \quad \mathsf{x} := \mathsf{x} \cdot \mathsf{x};\\ \quad \mathsf{n} := \mathsf{n}/2\\ \quad)\\ \mathsf{else\ (}\\ \quad \mathsf{y} := \mathsf{x} \cdot \mathsf{y};\\ \quad \mathsf{n} := \mathsf{n} - 1\\ \quad)\end{array} & \{\mathsf{y} \cdot \mathsf{x}^{\mathsf{n}} = a\} \end{array} \tag{10}$$

Using the result you arrived at in part (a), derive an algorithm p, containing the program fragment given above, that computes exponentiation:

$$\{\mathsf{x} = x_0 \wedge \mathsf{n} = n_0 \geq 0\}\ p\ \{\mathsf{y} = x_0^{n_0}\}.$$

(b) What are the invariant and the variant of your algorithm? (10)

(c) Write down the algorithm. (5)

Question 4:

(a) Construct an algorithm satisfying the given specification. The algorithm must contain a while loop with the invariant provided:

$$\left|\begin{array}{l|l|l} & & \{\mathsf{n} \geq 0\} \\ & ? & \\ & \vdots & \\ \{\mathsf{invariant} : I\} & \mathsf{while}\ ?\ \mathsf{do}\ (& \\ & \quad ? & \\ & \quad \vdots & \\ & \quad) & \\ & & \{\mathsf{y} = \mathsf{n} \cdot 2^{\mathsf{n}+1}\} \end{array}\right|$$

where the invariant of the algorithm must be:

$$I \equiv \mathsf{k} \leq \mathsf{n} + 1 \wedge \mathsf{z} = 2^{\mathsf{k}-1} \wedge \mathsf{y} = (\mathsf{k} - 1) \cdot 2^{\mathsf{k}}.$$

Your algorithm should involve only multiplications and/or additions; exponentiation should not be used as a primitive operation. You are not allowed to change the value of n inside the algorithm. (15)

(b) Complete the tableau proof of correctness of your algorithm. (10)

Question 5:

You are going to give a formal proof of correctness of the following program with respect to the given specification:

$$
\begin{array}{|l|l|}
\hline
 & \{\mathsf{x} \geq 0\} \\
\mathsf{k} := 0; & \\
\mathsf{y} := 0; & \\
\mathsf{z} := 0; & \\
\textsf{while } \mathsf{k} < \mathsf{x} \textsf{ do } (& \\
\quad \mathsf{z} := \mathsf{z} + 2 \cdot \mathsf{y} + \mathsf{k}; & \\
\quad \mathsf{y} := \mathsf{y} + 2 \cdot \mathsf{k} + 1; & \\
\quad \mathsf{k} := \mathsf{k} + 1; & \\
\quad \mathsf{z} := \mathsf{z} + \mathsf{y}; & \\
\quad) & \\
 & \{\mathsf{z} = \mathsf{x}^3\} \\
\hline
\end{array}
$$

a) What invariant would you choose to prove the partial correctness of the while loop? (8)

b) Using the invariant that you chose in (a), give a proof of the specification by completing the proof tableau. (8)

c) Write the non-trivial logical implications that still need to be proved to have a complete proof of correctness according to Hoare logic. (5)

d) Write a *variant* expression for the while loop. Explain how to use it to prove that the loop terminates. (4)

Question 6:

You are going to construct a program that satisfies a given specification. There are two input variables: a natural number n and an array of integers $\vec{\mathsf{a}}$ of length n. (The elements of the array are $\mathsf{a}[0], \ldots, \mathsf{a}[n-1]$.) The output variable is an array of integers $\vec{\mathsf{b}}$ also of length n.

This is the specification:

$$\{\mathsf{n} \geq 0\}\ P\ \{\forall i, 0 \leq i < n \rightarrow \mathsf{b}[i] = \max\{\mathsf{a}[i], \ldots, \mathsf{a}[n-1]\}\}$$

You are not allowed to modify the input variables n and $\vec{\mathsf{a}}$ inside the program P. You can use the binary maximum operation that maps any two integers x and y to the largest of them, $\max(x, y)$.

a) What should the *invariant* and the *variant* for the loop of your program be? (10)

b) Write down the program. (15)