THE UNIVERSITY OF AUCKLAND

FIRST SEMESTER, 2006
Campus: Tamaki

COMPUTER SCIENCE

COMPSCI 220: Algorithms and Data Structures

(Time allowed: ONE hour)

NOTE: Attempt *all* questions. Write answers in the boxes below the questions. You may continue your answers onto the “overflow” page provided at the back if necessary. Marks for each question are shown in brackets after the question. The use of calculators is NOT permitted.

SURNAME:

FORENAME(S):

STUDENT ID:

Section:	A	B	Total
Possible marks:	50	50	100
Awarded marks:			

QUESTION/ANSWER SHEETS FOLLOW

Surname: ______________________________

Forename(s): ______________________________

A. (Algorithm Analysis)

1. Assume that each of the expressions below gives the processing time $T(n)$ spent by an algorithm for solving a problem of size n. Select the dominant term(s) having the steepest increase in n and specify the Big-Theta complexity of each algorithm. [12 marks]

Marking scheme: 2, 0, and -1 marks for each correct, absent, and incorrect answer, respectively.

Expression	Dominant term(s)	$\Theta(\ldots)$
$10\log_2(n^2) + 0.1(\log_2 n)^2$	$(\log_2 n)^2$	$\Theta((\log n)^2)$
$50n + n^2\log_2 n + 0.01n^3$	$0.01n^3$	$\Theta(n^3)$
$50n^{0.1} + 200\log_2 n$	$50n^{0.1}$	$\Theta(n^{0.1})$
$100n + 35n\log_2 n + 3n^{1.5}$	$3n^{1.5}$	$\Theta(n^{1.5})$
$n^2\log_{16} n + n^2\log_2 n$	$n^2\log_{16} n$, $n^2\log_2 n$	$\Theta(n^2\log n)$
$n^3 + 3^n + n^6 + 6^n$	6^n	$\Theta(6^n)$

2. Prove that $f(n) = 0.5n^{1.5} + 50n\ln n$ is $\Omega(n\log n)$. [8 marks]

Hints: (*i*) $f(n)$ is $\Omega(g(n))$ if there exist a positive factor $c > 0$ and a positive threshold $n_0 > 0$ such that $f(n) \geq cg(n)$ for all $n \geq n_0$; (*ii*) $\ln z$ denotes the natural logarithm of z; (*iii*) useful formulae: $\frac{d\ln z}{dz} = \frac{1}{z}$; $\frac{dz^a}{dz} = az^{a-1}$.

The simplest solution: $0.5n^{1.5} > 0$ for $n \geq 1$, therefore $f(n) = 0.5n^{1.5} + 50n\ln n > 50n\ln n$ for $n \geq 1$ (i.e. $c = 50$ and $n_0 = 1$), and $f(n)$ is $\Omega(n\log n)$.

More complicated solution: $f(n) = 0.5n^{1.5} + 50n\ln n = n\ln n\left(50 + 0.5\frac{n^{0.5}}{\ln n}\right)$. If

$n \to \infty$, the fraction in the parentheses tends to infinity because in line with the Limit Rule

$$\lim_{n\to\infty}\left(\frac{n^{0.5}}{\ln n}\right) = \lim_{n\to\infty}\left(\frac{\frac{d}{dn}n^{0.5}}{\frac{d}{dn}\ln n}\right)$$
$$= \lim_{n\to\infty}\left(\frac{0.5n^{-0.5}}{n^{-1}}\right) = \lim_{n\to\infty}\left(0.5n^{0.5}\right) = \infty$$

Therefore, there always exists the factor $c = 50 + 0.5\frac{n_0^{0.5}}{\ln n_0}$ such that for $n > n_0$ $f(n) \geq cn\ln n$.

Any valid proof that $f(n) \geq cn\log n$ holds for all $n \geq n_0$ is acceptable.

Surname: ______________________________

Forename(s): ______________________________

3. Prove that $f(n) = 10n^2 + 5n + 15$ is not $O(n)$. [7 marks]

Hint: (*i*) $f(n)$ is $O(g(n))$ if there exist a positive factor $c > 0$ and a positive threshold $n_0 > 0$ such that $f(n) \leq cg(n)$ for all $n \geq n_0$.

A possible solution: $f(n) = 10n^2 + 5n + 15 = n\left(10n + 5 + \frac{15}{n}\right)$. Because the first term in parentheses tends to infinity when $n \to \infty$, there exists no positive constant factor $c > 0$ for any threshold n_0 such that $f(n) \leq cn$ for all $n \geq n_0$.

Any valid proof that $f(n) \leq cn$ cannot hold for all $n \geq n_0$ is acceptable.

A frequent error is to show that $f(n)$ is $O(n^2)$ and claim that therefore if is not $O(n)$. Because the Big-Oh notation specifies an asymptotic upper bound, the former relationship does not guarantee the latter one which has to be proven by itself.

4. Show basic steps of solving the recurrence $T(n) = \frac{2}{n}\left(T(0) + T(1) + \ldots + T(n-1)\right) + 1$; $T(0) = 0$. [10 marks]

Hint: A useful formula: $\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{n(n+1)} = \frac{n}{n+1}$.

Solution 1 (by "telescoping"): In line with the given recurrence,

$$\begin{array}{lcl} nT(n) & = & 2\left(T(0) + T(1) + \ldots + T(n-2) + T(n-1)\right) + n \\ (n-1)T(n-1) & = & 2\left(T(0) + T(1) + \ldots + T(n-2)\right) + (n-1) \end{array}$$

Therefore, $nT(n) - (n-1)T(n-1) = 2T(n-1) + 1$, i.e. $nT(n) = (n+1)T(n-1) + 1$, or $\frac{T(n)}{n+1} = \frac{T(n-1)}{n} + \frac{1}{n(n+1)}$. Solving the recurrence by telescoping, one obtains:

$$\begin{array}{lcl} \frac{T(n)}{n+1} & = & \frac{T(n-1)}{n} + \frac{1}{n(n+1)} \\ \frac{T(n-1)}{n} & = & \frac{T(n-2)}{n-1} + \frac{1}{(n-1)n} \\ \ldots & & \ldots \\ \frac{T(1)}{2} & = & \frac{T(0)}{1} + \frac{1}{1\cdot 2} \end{array}$$

or $\frac{T(n)}{n+1} = 0 + \frac{1}{1\cdot 2} + \ldots + \frac{1}{(n-1)n} + \frac{1}{n(n+1)} = \frac{n}{n+1}$, i.e. $T(n) = n$.

Solution 2 (by math induction): $T(1) = 2T(0) + 1 = 1$; $T(2) = \frac{2}{2}(0+1) + 1 = 2$; $T(3) = \frac{2}{3}(0+1+2) + 1 = 3$, $T(4) = \frac{2}{4}(0+1+2+3) + 1 = 4$, etc. Our hypothesis is $T(n) = n$.
Base case: $T(0) = 0$.
Inductive hypothesis: if $T(i) = i$ for $i = 0, 1, \ldots, n-1$, then $T(n) = n$.
Proof: $T(n) = \frac{2}{n}\left(0 + 1 + \ldots + (n-1)\right) + 1 = \frac{2}{n}\frac{(n-1)n}{2} + 1 = n - 1 + 1 = n$, i.e. just what is to be proved. Therefore, $T(n) = n$ for all $n \geq 0$.

Surname: ____________________

Forename(s): ____________________

5. Prove that the worst-case time complexity of insertion sort for an array of n data items is $\Theta(n^2)$. [10 marks]

The worst case for insertion sort is when an array of the size n is sorted in the inverse order. In this case at each step $i = 1, \ldots, n-1$, the algorithm makes i data comparisons and i data moves in order to insert each current array element $a[i]$ into its current correct position $i = 0$. The total number of these operations is $T(n) = 2\sum_{i=1}^{n-1} i = (n-1)n$, i.e. $T(n) = n^2 - n$.

Because $n^2 > T(n)$ for all $n \geq 1$ and $T(n) > 0.5n^2$ for $n \geq 2$ (as $n - 1 > 0.5n$ for $n \geq 2$), then $T(n)$ is $\Theta(n)$.

Solutions that take account of only comparisons or moves are equally valid.

6. Show steps of deleting the root node from the heap of size 11 and restoring the heap of size 10. [3 marks]

Position	1	2	3	4	5	6	7	8	9	10	11
Heap value	50	45	35	31	40	24	18	30	20	10	8
Root deleting	8	45	35	31	40	24	18	30	20	10	–
Heap restoring	45	8	35	31	40	24	18	30	20	10	–
	45	40	35	31	8	24	18	30	20	10	–
	45	40	35	31	10	24	18	30	20	8	–

There were misprints (the wrong positions of 30 and 31) in the heap that did not affect the solution; those who noticed and corrected the misprints received a bonus: +0.5 mark.

Surname: ____________________

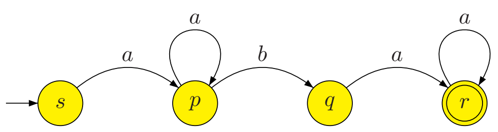
Forename(s): ____________________

B. (Automata and Pattern Matching)

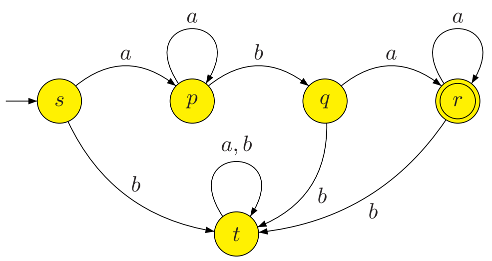
7. The statements in the following table are true or false. Mark the truth (true or false) of each statement in the third column. The first entry is an example. [10 marks]

0	Every language accepted by a DFA is infinite	false
1	Every finite language is accepted by some DFA	true
2	Every NFA is a DFA	false
3	The language $\{a^n b^n \mid n > 0\}$ is not accepted by any NFA	true
4	It is algorithmically decidable whether a DFA M accepts infinitely many strings	true
5	It is not algorithmically decidable whether an NFA N accepts finitely many strings	false

8. (a) Design an NFA N accepting the language $L = \{a^n b a^m \,:\, n, m > 0\}$. [5 marks]



(b) Design a DFA M such that $L(M) = L(N)$. [5 marks]



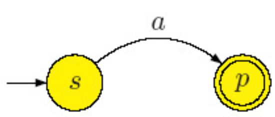
Surname: ________________________________

Forename(s): ________________________________

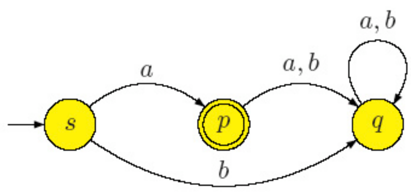
(c) Is the DFA M minimal? [5 marks]

The DFA M is minimal because: $\equiv_0 = \{\{s,q\},\{p\}\}$, but $s \not\equiv_1 q$ as $p = \delta(s,a) \not\equiv_0 \delta(q,a) = q$.

9. (d) Construct an NFA N that accepts the language $A = \{a\} \subset \{a,b\}^*$. [5 marks]



(e) Construct a DFA M such that $L(M) = L(N)$. [5 marks]



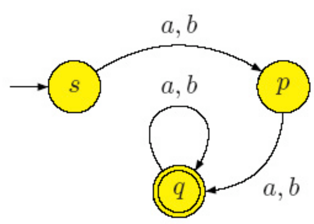
Surname: ____

Forename(s): ____

(f) Show that there is an algorithm which receives as input a DFA M' over the alphabet $\{a, b\}$ and decides whether $L(M') = \{a\}$ or $L(M') \neq \{a\}$. Clearly state all results you use. [10 marks]

It is known that there is an algorithm deciding whether two DFAs accept the same language. The language $\{a\}$ is accepted by the DFA M. So, we can apply the above algorithm to the DFAs M and M' to decide whether $L(M) = L(M')$, that is, $L(M) = \{a\}$.

10. Construct a DFA accepting $L = \{w \in \{a, b\}^* \mid |w| \geq 2\}$. [5 marks]



Surname: ______________________________

Forename(s): ______________________________

Additional work page