| PAPER CODE NO. | EXAMINER : Prudence Wong |
|---|---|
| **COMP108** | DEPARTMENT : Computer Science Tel. No. 794-3694 |

THE UNIVERSITY
*of* LIVERPOOL

# MAY 2006 EXAMINATIONS

Bachelor of Arts : Year 1
Bachelor of Science : Year 1
No qualification aimed for: Year 1

## ALGORITHMIC FOUNDATIONS

**TIME ALLOWED : TWO hours**

---

**INSTRUCTIONS TO CANDIDATES**

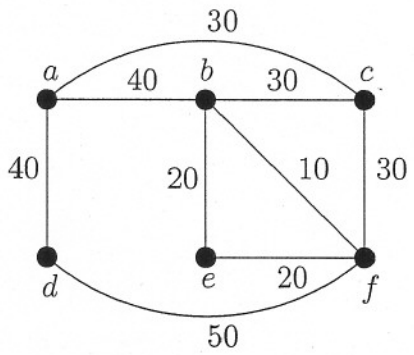
Answer **four** questions only.

- Candidates will be assessed on their best four answers.
- If you attempt to answer more questions than the required number of questions, the marks awarded for the excess questions will be discarded (starting with your lowest mark).
- All logarithms are to the base 2.

THE UNIVERSITY
*of* LIVERPOOL

## Question 1

Consider the following graph $G$. The label of an edge is the cost of the edge.




1A. Give the adjacency matrix of the graph $G$.

**[3 marks]**

1B. Briefly describe the *greedy* method.

**[2 marks]**

1C. Using *Kruskal's* algorithm, draw a *minimum spanning tree* (MST) of the graph $G$. Also write down the order in which the edges are selected.

Is the MST drawn unique? (i.e., is it the one and only MST for the graph?)

**[8 marks]**

1D. Using *Dijkstra's* algorithm, find the shortest paths from the vertex $a$ to *all* other vertices in the graph $G$.

Show the changes of the labels of the vertices step by step and give the order in which edges are selected.

N.B. There may be more than one solution. You only need to give one of the solutions.

**[12 marks]**

## Question 2

2A. State (without proof) the *order of magnitude*, in the form of $O(f(n))$, of the following functions. Use the simplest $f(n)$ possible in your answers.

I. $n^4 + n^3 \log^5 n + n^2 + 5$

II. $4\sqrt{n^5} + n^3 + 5n^2 + n$

III. $\sqrt{n} + \log n + 5$

IV. $2^n + n^3$

**[4 marks]**

2B. Show that the function $n^3 + 3n^2 \log n + 5n + 2$ is $O(n^3)$.

**[5 marks]**

2C. Suppose we are given $n$ numbers $A[0]$, $A[1]$, $\cdots$, $A[n-1]$. Write a pseudo code of an algorithm to sort the numbers in *descending* order (from the largest to the smallest).

What is the name of your sorting algorithm?

What is the time complexity (in big-O notation) of your algorithm?

**[6 marks]**

2D. Solve the following recurrence formula for any positive integer $n$ by the *iterative method.*

$$f(n) = \begin{cases} 4 & \text{if } n = 1, \\ 2f(n-1) + 4 & \text{if } n > 1. \end{cases}$$

(Hint: You can use the fact that $1 + x + x^2 + \ldots x^{n-1} = (x^n - 1)/(x - 1)$ for any positive integer $n$ and any real number $x \neq 1$.)

**[10 marks]**

## Question 3

3A. Briefly describe the idea of the *divide-and-conquer* technique.

**[3 marks]**

3B. Suppose $T(n)$ denotes the time complexity of the binary search algorithm on $n$ numbers.

I. Explain why $T(n)$ can be described by the following recurrence.

$$T(n) = \begin{cases} 1 & \text{if } n = 1, \\ T(\lfloor n/2 \rfloor) + 1 & \text{if } n > 1. \end{cases}$$

**[3 marks]**

II. Show that $T(n) = O(\log n)$ by the *substitution method*, i.e., to use Mathematical Induction.

**[7 marks]**

3C. Given a length-$n$ sequence $S$ of characters $S[0..(n-1)]$, a length-$m$ sequence of characters $T[0..(m-1)]$ is called a *substring* of $S$ if there exists some $0 \leq i \leq n - m$ such that $S[i..(i+m-1)] = T[0..(m-1)]$.
For example, if $S$ is ACGTACGGG, then ACGG is a substring appearing once in $S$, ACG is a substring appearing twice and ACC is not a substring of $S$ at all.

Design and write a pseudo code algorithm to determine the number of times that $T$ appears as a substring of $S$.

What is the worst case time complexity of your algorithm (in big-O notation)? Explain briefly.

**[12 marks]**

THE UNIVERSITY
*of* LIVERPOOL

## Question 4

4A. What are the *time complexities* (in big-O notation) of the following pseudo codes?

I.
$s = 0$
**for** $i = 1$ to $n$ **do**
  **for** $j = 1$ to $m$ **do**
    $s = s + j$
Output $s$

II.
$i = 0$
**while** $n > 1$ **do**
**begin**
  $n = n/2$
  $i = i + 1$
**end**
Output $i$

III.
**for** $i = 0$ to $n-1$ **do**
  **for** $j = 0$ to $n-1$ **do**
  **begin**
    $C[i, j] = 0$
    **for** $k = 0$ to $n-1$ **do**
      $C[i, j] = C[i, j] + A[i, k] \times B[k, j]$
  **end**

**[6 marks]**

4B. An *optimisation* problem can be turned into a *decision* problem if we add a parameter $k$; and then ask whether the optimal value in the optimisation problem is at most or at least $k$.

State the decision version of the following optimisation problems:

I. Given an undirected graph $G$, find the minimum number of colours that is needed to colour the vertices in $G$ such that no two adjacent vertices have the same colour.

II. Given $n$ items with weights and values, and a knapsack with capacity, find the most valuable subset of items such that the total weight does not exceed the capacity of the knapsack.

**[6 marks]**

4C. Which of the following problems is/are NP-complete problem(s)?

I. Finding MST in a weighted undirected graph.

II. Sorting $n$ numbers into ascending order.

III. Vertex Cover Problem.

**[3 marks]**

4D. In the Knapsack problem, we are given $n$ items each with a *weight* and a *value*, and a knapsack with a fixed *capacity*. The problem is to find the most valuable subset of items that can fit into the knapsack.

Describe an exhaustive search algorithm to find such a subset of items.

What is the time complexity of the algorithm (in big-O notation)? Explain briefly.

**[10 marks]**

THE UNIVERSITY
*of* LIVERPOOL

## Question 5

5A. Briefly describe the idea of the dynamic programming technique.

**[5 marks]**

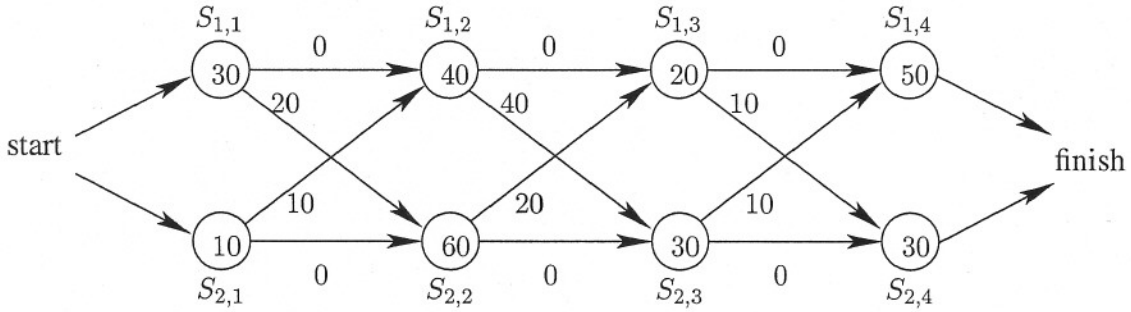
5B. Consider the following recursive algorithm.

**Algorithm** $f(n)$
  **if** $0 \leq n \leq 2$ **then**
    result = 1
  **else**
    **if** $n > 2$ **then**
      result = $f(n-1) + f(n-2) + f(n-3)$
  return result

The above algorithm is not efficient. Design and write the pseudo code of a faster (non-recursive) algorithm using the concept of dynamic programming. What is the time complexity of the faster algorithm (in big-O notation)?

**[6 marks]**

5C. Suppose there are two assembly lines each with 4 stations, $S_{i,j}$. The assembly time is given in the circle representing the station and the transfer time is given next to the arrow from one station to another.




I. Using dynamic programming, fill in the table of the minimum time $f_i[j]$ needed to get through station $S_{i,j}$. Show all the intermediate steps in computing these values.

| $j$ | $f_1[j]$ | $f_2[j]$ |
|---|---|---|
| 1 | ... | ... |
| 2 | ... | ... |
| 3 | ... | ... |
| 4 | ... | ... |

II. What is the minimum time $f^*$ needed to get through the assembly line?

III. Which stations should be chosen?

**[14 marks]**