PRIFYSGOL CYMRU; UNIVERSITY OF WALES

M.Sc. AND DIPLOMA EXAMINATIONS

May/June 2003

SWANSEA

Computer Science

CS M22 Algorithms and their Applications

Attempt 2 questions out of 3

Time allowed: 2 hours

Students are permitted to use the dictionaries provided by the University

Students are NOT permitted to use calculators

CS_M22 2002/03
Algorithms and their Applications

(Attempt 2 questions out of 3)

Question 1 The Euclidian algorithm and its analysis

(a) The divisor relation and the greatest common divisor

- (i) Define the divisor relation $\boldsymbol{a \mid b}$. (State what kind of objects a and b are). **[1]**
- (ii) Define $\mathbf{gcd}(\boldsymbol{a, b})$ for $a, b \in \mathbb{N}_0$. **[2]**
- (iii) Compute $\gcd(a, b)$ for all cases where either $a \leq 1$ or $b \leq 1$ holds (or both). **[2]**

[5 marks]

(b) The Euclidian algorithm

- (i) State the Euclidian algorithm (using, e.g., Pascal syntax, C syntax or pseudo code). **[2]**
- (ii) Compute $\gcd(136, 150)$, showing the sequence of recursive calls. **[1]**
- (iii) What is the next step of the Euclidian algorithm for computing $\gcd(a, b)$ for inputs with $a \leq b$? (Cover all possibilities.) **[2]**
- (iv) Why does the Euclidian algorithm always terminate? (Try to prove it). **[3]**

[8 marks]

(c) Fibonacci numbers

- (i) Define the Fibonacci numbers $\boldsymbol{F_n}$. **[1]**
- (ii) Compute the sequence $F_0, F_1, \ldots, F_{10}$. **[1]**
- (iii) State the explicit formula for F_n (without proof). **[2]**

[4 marks]

(d) Analysing the Euclidian algorithm

- (i) What do you know about the complexity of the Euclidian algorithm? (No proofs, but try to be as explicit and complete as possible.) **[2]**
- (ii) Assume that the computation of $\gcd(a, b)$ via the Euclidian algorithm needs $k \in \mathbb{N}_0$ recursive calls. How large must the inputs a, b be? (Only state the relation.) **[1]**
- (iii) Give a proof for your answer from (ii). **[5]**

[8 marks]

Question 2 Searching and Sorting

(a) The meaning of order

(i) What does it mean for a collection of objects to be sorted? **[2]**

(ii) Consider the task of searching for an object in a collection of n objects. Under what circumstances and how can we do better than using n comparisons? **[3]**

(iii) How many comparisons are needed for doing a binary search on n objects? (Only the statement is required; you do not need to justify your answer.) **[1]**

[6 marks]

(b) `SelectionSort`, `InsertionSort`, `MergeSort`, `HeapSort`

(i) Show how to order the sequence $5, 8, 1, 3, 9, 6, 4, 2, 7$ (into ascending order) via each of the above four sorting algorithms. For `SelectionSort` and `InsertionSort` show the eight sequences which correspond to the states of the array after single iterations of the main loop. For `MergeSort` show the binary tree built up, and label each node with the original sequence (to be sorted) and the sorted sequence. Finally, for `HeapSort`, show a sequence of binary trees, with nodes labelled by elements of the sequence, which correspond to the main steps in sorting the sequence: Start with the original sequence, transform it into a heap, and then show stepwise how to extract the currently largest element, put it into the right position, and "re-heapify". Mark each of the binary trees which fulfils the heap condition. **[7]**

(ii) State the worst-case complexity of the four sorting algorithms above, in terms of the growth rate of the maximum number of comparisons needed on input sequences of length n. **[1]**

(iii) State specific advantages and disadvantages for each of the four sorting algorithms above. **[4]**

[12 marks]

(c) Heaps

(i) Given an index i, how can one compute the two child indices and the parent index in a heap? **[1]**

(ii) How is it possible to find out whether an index i corresponds to the root of the tree, or to some leaf node? **[2]**

(iii) State the "heap condition", specifying when an array $a[1], \ldots, a[n]$ is to be considered a "max heap". **[1]**

(iv) Describe in words the main steps of `HeapSort`. **[3]**

[7 marks]

Question 3 Graphs and graph search

(a) Graphs and presentations

(i) Define a directed graph. **[2]**

(ii) Consider the graph $G = (V, E)$ on $V = \{a, b, c, d, e, f, g, h\}$ as depicted below. Give its adjacency list representation, using the alphabetical order of vertices. **[2]**

(iii) Characterise the size of the adjacency list representation of a graph in terms of its number of vertices and number of edges. **[1]**

(iv) How many edges can a directed graph with n vertices have at most? Give reasons for your answer. **[2]**

[7 marks]

(b) Bipartite graphs

(i) What is a bipartite undirected graph? **[2]**

(ii) Consider the "chessboard graph" for an $n \times n$ chessboard, where the nodes are the squares, and two squares are connected by an edge if they are horizontal or vertical neighbours. Draw the graph for $n = 4$. Are these graphs bipartite? (Give reasons for your answer.) **[4]**

[6 marks]

(c) Breadth-first search

(i) Show how breadth-first search works on the graph G below with source vertex a, using the alphabetical vertex order. Indicate schematically the succession of colourings, give the final values of the auxiliary functions d and π, and display the resulting BFS tree. **[5]**

(ii) What is the complexity of breadth-first search? **[1]**

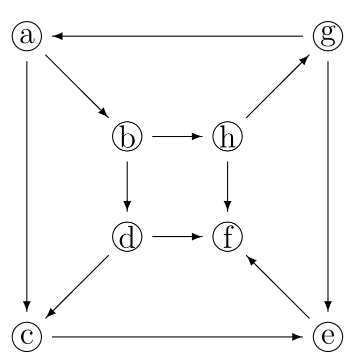
[6 marks]

(d) Depth-first search

(i) Show how depth-first search works on the graph G below, using the alphabetical vertex order. Indicate the succession in which vertices are discovered, the parenthesis structure and the resulting DFS forest. **[5]**

(ii) What is the complexity of depth-first search? **[1]**

[6 marks]



G