University of Nottingham

SCHOOL OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

A LEVEL 2 MODULE, AUTUMN SEMESTER 1998–1999

ALGORITHMS AND DATA STRUCTURES
(Course G52ADS)

Time allowed TWO hours

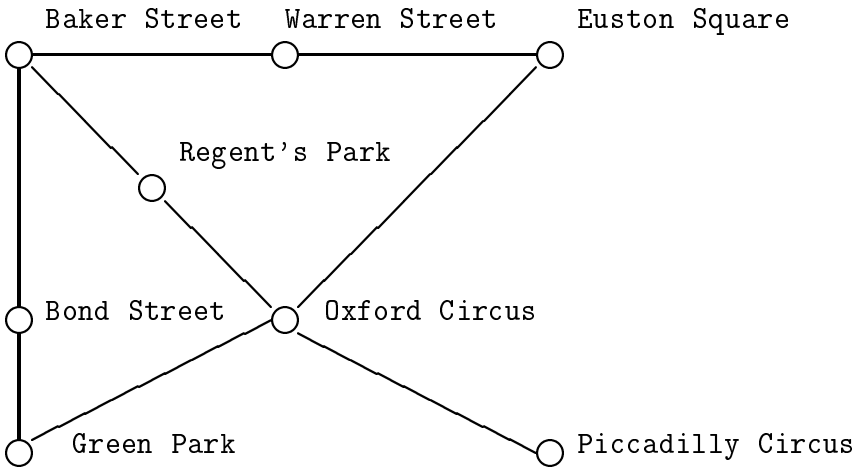
Candidates must NOT start writing their answers until told to do so

Candidates should attempt FOUR out of five questions

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

Only silent, self-contained calculators with a single-line display are permitted in this examination. Dictionaries are not allowed with one exception. Those whose first language is not English may use a dictionary to translate between that language and English provided that neither language is the subject of this examination.

1 (a) Give the adjacency matrix and adjacency list representations for the following graph: (5)



Question continued overleaf

(b) Give the trace of a depth-first search through the graph, assuming that the search starts at `Green Park`. (5)

(c) Give the trace of a breadth-first search through the graph, assuming that the search starts at `Green Park`. (5)

(d) Imagine that you are asked to design a route planner for London Underground. A user presses buttons with the start station name and the destination station name. The route planner should display the list of stations on the shortest route from the start to the destination, where 'shortest' means: with the minimal number of stations. The planner does not have to minimise line changes. Explain how you would represent the data, and motivate your choice. (5)

(e) Suppose that the route planner has to minimise line changes. How would you represent the data in this case? (5)

2 The following program implements a sorting algorithm:

```
void  sort(int arr[], int len){
   int i;
   int j;
   int pos_greatest;
   int *temparray;
   temparray = new int[len];

   for( i = len - 1; i > 0; i--){
      pos_greatest = 0;
      for(j = 0; j <= i; j++){
         if( arr[j] > arr[pos_greatest]){
            pos_greatest =  j;
         }//end if
      }//end inner for loop
   temparray[i] = arr[pos_greatest];
   temparray[pos_greatest] = arr[i];
   arr[i] = temparray[i];
   arr[pos_greatest] = temparray[pos_greatest];
   }//end outer for loop
}//end sort
```

(a) What is the name of this algorithm? Explain how it works. Trace it for the following array: `[4,1,3,5]`. (5)

Question continued overleaf

(b) What is the time complexity of this algorithm? Explain how you arrive at this estimate. (5)

(c) How would you improve the given implementation without changing the underlying algorithm? Write your version of (the relevant part of) the code. (Don't worry about getting the C++ syntax exactly right.) (5)

(d) What is the loop invariant of the inner loop in the program given above? What is the loop invariant of the outer loop in the program given above? (5)

(e) Using the loop invariants above, prove that the program is correct, i.e. that when it is finished, the array `arr[]` is sorted in ascending order. (5)

3 (a) What is a complete binary search tree? Draw a complete binary search tree with 5 nodes storing characters A, B, C, D, E. (Assume the usual ordering $A < B < C < D < E$.) (5)

(b) Describe two possible implementations for complete binary trees. Show how your tree from question 3(a) would be implemented in each of them. (5)

(c) Describe the search algorithm for binary search trees. For one of the implementations of complete binary trees, give pseudocode for the search method. Trace the search for C on your tree from question 3(a). (10)

(d) How many comparisons in the worst case would the search algorithm make searching for an item in a complete binary tree holding 1000 nodes? (5)

4 (a) What does the 'order of' (big O) complexity estimate tell you about an algorithm? Which other considerations are important when choosing an algorithm? (5)

(b) Which considerations are important in designing algorithms and data structures for external storage? (5)

(c) What are the merits and demerits of using hash tables and balanced trees for storing data? (5)

(d) What is the principle behind divide and conquer algorithms? (5)

(e) What is the principle behind greedy graph algorithms? (5)

5 A database of a telephone company contains telephone numbers, addresses and names of customers. It should be possible to search efficiently for the telephone number given a customer's name and address, and for the name and address given a telephone number. Insertions (i.e. adding a new customer record with a new telephone number) and deletions (i.e. deleting a customer record, whereby a telephone number becomes free and should be re-used) are done reasonably often. The company is doing well, and the database is growing.

Assume that the database is kept in internal memory.

(a) Specify ADTs and methods which are involved in the problem. (10)

(b) Describe the data structures and algorithms you would employ to implement them, explaining how they are to be used, and the reasons for your choices. (15)