PAPER CODE NO.	EXAMINER : Dr. C. Dixon
COMP505	DEPARTMENT : Computer Science Tel. No. 43674

THE UNIVERSITY
of LIVERPOOL

JANUARY 2005 EXAMINATIONS

Master of Science : Year 1

ALGORITHM DESIGN AND IMPLEMENTATION

TIME ALLOWED : Two and a half hours

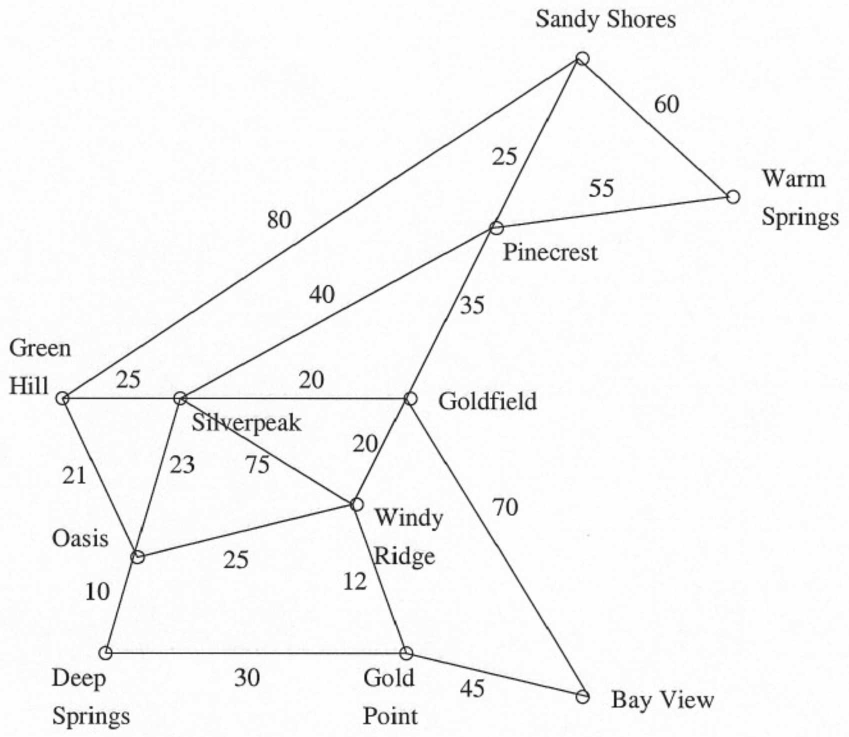
INSTRUCTIONS TO CANDIDATES

Answer **four** questions only.

If you attempt to answer more questions than the required number of questions (in any section), the marks awarded for the excess questions will be discarded (starting with your lowest mark).

THE UNIVERSITY
of LIVERPOOL

1. (a) The graph represents the distance by road between several villages on an island. Currently all roads on the island are unpaved (unsurfaced) although a plan has been suggested to pave (surface or tarmac) some of the roads. Give an algorithm to work out which sections of road should be paved such that the length of paved road required is the minimum possible and that there is a paved route between any two villages (possibly via other villages). **4 marks**



(b) Apply your algorithm to the graph shown, showing all your working, and the order in which each edge is added. State the minimum amount of paved road required. **6 marks**

(c) When applying the algorithm in part (a) to any graph is the solution obtained (in terms of which graph edges are selected) unique? Justify your answer. Does the algorithm always find the optimum solution (i.e. the minimum length of paved road)? **4 marks**

(d) What main algorithm design method have you used in this algorithm? Describe and evaluate the main features of this method of algorithm design. **5 marks**

(e) A traveller starting from Sandy Shores wishes to find a path of minimum length (travelling on either paved or unpaved roads) to allow him to visit each village only once and to return to his starting point. Suggest an algorithm using the same design technique as part (d) and comment on its effectiveness. Note you do not need to actually apply the suggested algorithm to the above graph. **6 marks**

THE UNIVERSITY
of LIVERPOOL

2. (a) Describe what is meant by the syntax and semantics of a programming language. **2 marks**

   (b) Using both pseudo code and flow diagrams explain the operation of a "for loop" (definite loop) and one other loop construct. **6 marks**

   (c) Consider a situation where a user enters a sequence of positive integers (whole numbers greater than 0) one by one, stopping when zero or a negative integer is entered. The mean (the total of the numbers divided by the number of values entered) should be calculated and returned from the numbers input. The last (negative or zero number) should not be included in the calculation. Write pseudo code for the above algorithm. Comment on the choice of loop construct you have used. **6 marks**

   (d) What is BNF and why is it useful? In BNF or EBNF give a production for $< for\_loop >$ to define a "for loop" in Java. You can assume that the productions for $< declaration\_statement >$, $< boolean\_condition >$, $< assignment\_statement >$ and $< statement\_sequence >$ are defined elsewhere. **5 marks**

   (e) Comment on any differences between the standard form of a "for loop" and how it is are represented in Java. **2 marks**

   (f) What are the limitations of BNF? **4 marks**

THE UNIVERSITY
of LIVERPOOL

3. (a) What is meant by the *order of complexity* of an algorithm? **2 marks**

   (b) Consider the pseudo code for the following two algorithms. The first algorithm, `findminimum`, takes a list of unsorted numbers l as input.

```
findminimum(l)
  min = first value;
  for i from 1 to length of l
     if ith value less than min
       then min = ith value;
  end for;
  return min;
```

   The second algorithm, `find`, takes a sorted list of numbers l and a number e as input.

```
find(l,e)
  if l is empty
  then return false
  else
    compare e with middle element of l
    if middle element == e
       return true
    else if e < middle element
       return find(first half of l,e)
    else return find(second half of l,e)
```

   Explain briefly the operation of each algorithm. For each of the above algorithms give the order of complexity for that algorithm explaining clearly why the algorithm has that order of complexity. **12 marks**

   (c) An interesting complexity class is that of NP complete problems. Explain what is meant by this class of problems. Give a problem that is in this class. **6 marks**

   (d) Algorithmic problems can be classified as feasible, infeasible and undecidable. What is meant by each of these and why are they of interest to a computer scientist? **5 marks**

THE UNIVERSITY
of LIVERPOOL

4. (a) What is an abstract data type? How are abstract data types implemented in Java? **4 marks**

(b) Let Q be the queue below where 3 is at the front of the queue.

3	7	1	10	4

Having created a queue, the basic operations on queues are:

- *enqueue* something onto the end of the queue;
- *dequeue* the front item (i.e. remove the front item from the queue);
- inspect what is on the *front* of the queue;
- test whether the queue is *empty*;
- return the *size* of the queue;

with the specification for each operation as described in lectures. Show what is returned (if anything) and the resulting queue, after each of the following operations is applied to Q: enqueue(21), empty(), size(), front(), dequeue(). **5 marks**

(c) One way to implement queues is by using an array, A, with maximum size N where (assuming the first array subscript is 0) $A[0]$ is *always* used to store the front of the queue and a variable r is used to store the index of the array element *after* the end of the queue. What are the advantages and disadvantages of using this implementation? **4 marks**

(d) A second implementation of the queue also uses an array A and two variables f and r (initialised to $f = r = 0$) where f is used to give the index of the front element in the array and r is to give the index after the last element. Below is some pseudo code for the enqueue and dequeue operations.

```
enqueue(val):
  Q[r] = val
  r = r+1

dequeue():
  Q[f] = null
  f = f+1
```

Explain any disadvantages of the overall approach of this implementation and any particular problems with enqueue and dequeue as outlined. **6 marks**

(e) Describe how the above implementation in part (d) could be improved to overcome the disadvantages outlined and give pseudo code for enqueue and dequeue **6 marks**

THE UNIVERSITY
of LIVERPOOL

5. (a) What is the difference between a function and a procedure? How are functions and procedures implemented in Java? Illustrate your answer with a small example of each from Java. **7 marks**

   (b) There are five main ways by which parameters can be passed to functions or procedures in modern high level languages. Describe three of these. **9 marks**

   (c) What are the two main ways of parameter passing in Java and on which data types are they used? Assume a call is made to the function you gave in part (a) from the main program using actual parameters. Explain clearly how the parameters are passed to the function with reference to your example. **4 marks**

   (d) Explain what is meant by the scope of an identifier. Discuss the scope of an identifier in one of your examples in part (a). **5 marks**