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

THE UNIVERSITY
of LIVERPOOL

JANUARY 2003 EXAMINATIONS

Master of Science : Year 1

ALGORITHM DESIGN AND IMPLEMENTATION

TIME ALLOWED : Two Hours

INSTRUCTIONS TO CANDIDATES

Answer **four** questions only.

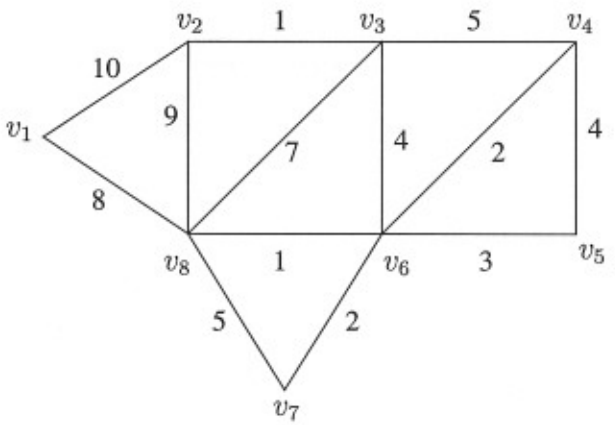
If you attempt to answer more questions than the required number of questions (in any section), the marks awarded for the excess questions will be discarded (starting with your lowest mark).

THE UNIVERSITY
of LIVERPOOL

1. (a) Describe three common techniques for designing algorithms. For each technique briefly explain an algorithm that uses it (this should be different from the one described in (b) below). **12 marks**

   (b) A weighted graph contains n nodes $v_1, \ldots v_n$. Some pairs of nodes are joined by undirected edges of given length. It is required to find a minimal spanning tree for the graph (i.e. there is a route between any two nodes, no circular routes and the total length of the edges is a minimum). Give an algorithm to find the minimal spanning tree. **5 marks**

   (c) Showing all your working, use your algorithm to find the minimal spanning tree for the network below. **5 marks**



   (d) Is the solution unique (i.e. could there be more than one spanning tree that is minimal by applying the algorithm to a graph)? Why? **3 marks**

2. (a) Explain briefly what is meant by structured programming. **2 marks**

   (b) Using both pseudo code and a flow diagram explain the operation of a choice statement (using if, then, else, endif). **6 marks**

   (c) Give a parse tree for the if statement

```
if a > 7 then x := 5; else x:= 2; endif;
```

   which is derived from the production for *if_statement* in the grammar below. **8 marks**

$$
\begin{array}{rcl}
< if\_statement > & ::= & \texttt{if } < boolean\_exp > \texttt{ then} \\
 & & < statement\_sequence > \texttt{ endif ;} | \\
 & & \texttt{if } < boolean\_exp > \texttt{ then} \\
 & & < statement\_sequence > \texttt{ else} \\
 & & < statement\_sequence > \texttt{ endif ;} \\
< statement\_sequence > & ::= & < statement > | \\
 & & < statement >< statement\_sequence > \\
< statement > & ::= & < assignment\_statement > | \\
 & & < if\_statement > \\
< assignment\_statement > & ::= & < identifier >:=< exp >; \\
< boolean\_exp > & ::= & < exp >< comparison\_operator >< exp > \\
< comparison\_operator > & ::= & > | < | \geq | \leq | = | \neq \\
< exp > & ::= & < term > \\
 & & | < exp >< adding\_operator >< term > \\
< adding\_operator > & ::= & + | - \\
< term > & ::= & < identifier > | < constant > |(< exp >) \\
< identifier > & ::= & < letter > | < identifier >< letter > \\
 & & | < identifier >< digit > \\
< constant > & ::= & < digit > | < constant >< digit > \\
< letter > & ::= & A|B|C|D|E|F|G|H|I|J|K|L|M|N|O \\
 & & |P|Q|R|S|T|U|V|W|X|Y|Z|a|b|c|d|e|f \\
 & & |g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z \\
< digit > & ::= & 0|1|2|3|4|5|6|7|8|9
\end{array}
$$

(d) What languages cannot be expressed in BNF (or EBNF)? Explain briefly why. **3 marks**

(e) Consider the following choice statement in Java syntax.

```
if (a == 10)
  {
    if (b < 3)
      screen.println(''X'');
    else
      screen.println(''Y'');
  }
else
  screen.println(''Z'');
```

Give values of a and b that result in:-

i. X being printed;

ii. Y being printed;

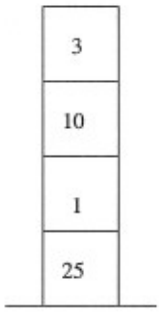
iii. Z being printed;

explaining what happens on each path through the code. **6 marks**

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

   (b) What is a stack data structure? **2 marks**

   (c) Let S be the stack below where 3 is at the top of the stack.



   Having created a stack, the basic operations on stacks are:

   - *push* something onto the stack;
   - *pop* the top item (i.e. remove the top item from the stack);
   - inspect what is on the *top* of the stack;
   - test whether the stack is *empty*;
   - return the *size* of the stack;

   with the specification for each operation as described in lectures. Show what is returned (if anything) and the resulting stack, after each of the following operations is applied to S: push(7), empty(), size(), top(), pop(), pop(). **6 marks**

   (d) One way to implement stacks is by using an array, A, with maximum size N where the variable t (initialised to $t = -1$) is used to give the index of the top element in the array. Assume the first array subscript is 0. Give pseudo code for the five operations above in an array based implementation of stacks. **10 marks**

   (e) How else could stacks be implemented? Give one advantage of this alternative implementation. **3 marks**

4. (a) What is the difference between a procedure and a function? How are procedures and functions implemented in Java? Illustrate your answer with a small example of each from Java. **8 marks**

   (b) In a modern high level language what are the *defining* and *applied* occurrences of identifiers? Illustrate both the above with a small example in Java. In the presence of block structure how is the defining occurrence corresponding to a given applied occurrence found? **7 marks**

   (c) Explain what are meant by virtual data types. What types are commonly provided as virtual types? Give a very brief description of each type you mention. Give one of these that is not directly provided by Java? **10 marks**

5. (a) Clearly explaining all steps taken, show how the following list

$$[15, 7, 21, 5, 1]$$

may be sorted using (the basic, unoptimised) *bubble sort*. **5 marks**

(b) Give the pseudo code for another list sorting algorithm (other than bubble sort). **5 marks**

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

(d) Give the order of complexity of the bubble sort algorithm, showing how this has been obtained. **5 marks**

(e) An optimised version of bubble sort reduces the number of comparisons needed each time round the cycle. Explain briefly why this works. What is the order of complexity of the optimised bubble sort algorithm. Why? **7 marks**