THE BRITISH COMPUTER SOCIETY

THE BCS PROFESSIONAL EXAMINATION
Certificate

SOFTWARE DEVELOPMENT

20th April 2001 – 2.30 p.m. – 4.30 p.m.
Time: 2 hours

SECTION A

Answer TWO questions out of FOUR. All questions carry equal marks.

*The marks given in brackets are **indicative** of the weight given to each part of the question.*

1. ***Code cracking*** - Simple codes, based on one arbitrary symbol (character) for each letter of the alphabet, could be broken easily by means of frequency tables. Thus in the English language 'e' and 't' greatly predominate in any text, followed by 'n', r', 'o', etc. Thus a count of the different symbols in a simple coded message can be made and the one occurring most frequently can be assigned as 'e', the next as 't' and so on. Other letter frequencies in sufficiently long messages are also close to the values given in the frequency table (see **Table 1.1**) at the bottom of the page.

a) Write an algorithm in pseudocode, for a well-structured program, where an input text file 'text' containing a simple coded message is to be read. Counts are to be made of all the symbols read, ignoring all spaces and punctuation (if present). A sorted list of the frequency of each symbol is then to be printed. The algorithm should then use the frequency table (Table 1.1) to assign letters to the symbols in descending order of occurrence. The original coded message is then to be output and beneath it are to be printed the corresponding assigned letters and an '*' for the rest. **(24 marks)**

b) Discuss the data structures which would be needed before the program could be developed further. **(6 marks)**

You may assume that the procedure SORT() with appropriate parameters is available for your use. Code for a sorting technique is not required.

Example with three assigned letters (e,t,n) where the decoded message is

```
the BCS examination is dreadfully difficult this year :-
```

…….…….. coded message ……

```
t*e *** e****n*t**n ** **e******* ********t t*** *e**
```

E	13.0		...	…..		U	2.6
T	9.2		S	6.1		M	2.5
...	…..		D	4.2		...	…..
N	7.9		L	3.6		X	0.5
R	7.6		H	3.4		Q	0.3
O	7.5		C	3.1		K	0.3
A	7.4		F	2.8		J	0.2
I	7.4		P	2.7		Z	0.1

Table 1.1
(Frequency of occurrence of letters in English as a percentage)
(Encyclopaedia Britannia, Vol. 5, page 331)

2. *a)* Dry run the algorithm given below (**Algorithm 2.1**) with suitably chosen test data values. Only one dry run is expected. 'max' should be restricted to 3 or 4. **(16 marks)**

b) Explain the purpose of the algorithm. Highlight any problems you believe the algorithm contains. **(6 marks)**

c) Translate the algorithm into a suitable procedural language. Add appropriate comments and input prompts, and more meaningful output statements. State the language used. **(8 marks)**

```
Line
No.  Algorithm

0.   DECLARE data as ARRAY [1.. max]
1.   total  <-- 0
2.   INPUT limit
3.   FOR ct <-- l TO limit DO
4.   BEGIN
5.        INPUT data[ct]
6.        total <-- total + data[ct]
7.   END
8.   average <-- total / limit
9.   PRINT average
10.  above <-- 0
11.  FOR ct <-- 1 TO limit DO
12.  BEGIN
13.       IF data[ct] GREATER THAN average THEN
14.       BEGIN
15.            above <-- above + 1
16.            PRINT above, data[ct]
17.       END
18.  END
19.  PRINT ABOVE
```

Algorithm 2.1

3. *a)* What features should be incorporated into a good CASE tool and why? **(11 marks)**

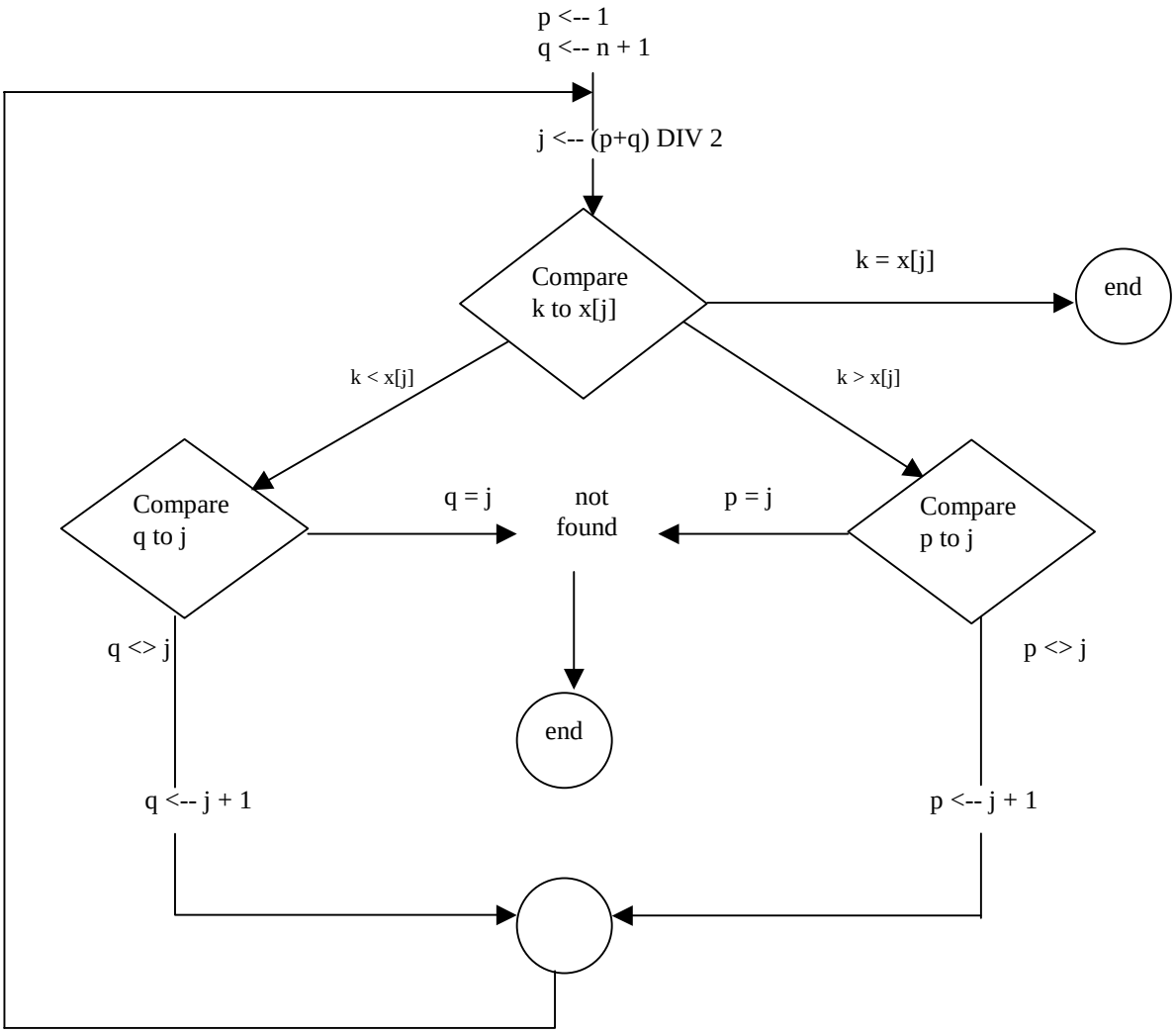
b) Critically comment on ONE CASE tool with which you are familiar. **(11 marks)**

c) What impact do you believe the continuing development of the World Wide Web will have on CASE tools and why? **(8 marks)**

4. The binary search technique is important in locating a single record from a table held in an array (x[j] for j = 1 to n) using a single key (k). A flowchart, taken from an old computing book of methods, is given below:

a) Translate this flowchart into pseudocode. Meaningful names for the items being coded should be used. **(18 marks)**

b) Convert this pseudocode into a structured form which does not use 'GOTO' statements and which would be suitable for straightforward translation into a procedural language. Include comments and prompts for the input of data as required. **(12 marks)**



'p' is the lower bound of the search
'q' is the upper bound of the search
'n' is the size of array 'x' or its upper bound
'DIV' means integer division; e.g. 7 DIV 3 returns 2

'j' is the current position being tested
'k' is the required key value
'x[j]' is the data item currently being compared

NOW PLEASE ANSWER QUESTIONS FROM SECTION B OVERLEAF ➜

SECTION B

Answer FIVE questions out of EIGHT. All questions carry equal marks.

The marks given in brackets are indicative of the weight given to each part of the question.

5. Selection, sequence and iteration are common constructs in programming languages. Show, via examples, how these constructs are incorporated into a design method of your choice. **(12 marks)**

6. The constructs 'function' and 'procedure' are used extensively in programming languages.

a) Describe these constructs using examples. **(6 marks)**
b) Describe a context where either a function or a procedure could be used. **(3 marks)**
c) Give an example where it would be more appropriate to use a function. **(3 marks)**

7. If A is an approximation to the cube root of a real number N, then A + C is a better approximation where C is given by the formula

$$C = \frac{1}{3}\left[\frac{N}{A^2} - A\right]$$

a) Write a function which takes two parameters N and E (where E is a given error value) and which repeatedly evaluates the cube root of N until subsequent values differ by less than the chosen error value, E. The first approximation of the cube root (A) should be the square root of N. **(10 marks)**

b) Why might this function never terminate? **(2 marks)**

8. Compare and contrast, with examples, the following pairs of terms:

a) compilers and interpreters **(4 marks)**
b) stacks and queues **(4 marks)**
c) white box and black box testing **(4 marks)**

9. Define, with reasons, a set of criteria suitable for assessing user interface design. Include in your answer the type of user you are considering. **(12 marks)**

10. Write brief notes on each of the following:

a) library routines **(4 marks)**
b) independent compilation **(4 marks)**
c) parallel processing **(4 marks)**

11. Identify the potential types of users of software documentation and the reasons why they need the documentation. By what methods might the users gain access to the documentation? **(12 marks)**

12. What are the main functions of the system software on a general purpose computer? State, with reasons, whether these functions differ significantly between a Personal Computer (PC) or a Linux/Unix type environment **(12 marks)**