THE BRITISH COMPUTER SOCIETY

THE BCS PROFESSIONAL EXAMINATION
Diploma

SOFTWARE DEVELOPMENT ENVIRONMENTS

29th April 2004, 10.00 a.m.-12.00 p.m.
Answer FOUR questions out of SIX. All questions carry equal marks.
Time: TWO hours.

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

1. *a)* Briefly describe the requirements, specification and design stages of the software development lifecycle. For each of the three stages give an example of a common error that can be made. **(9 marks)**

b) At what stage of development would a prototype be constructed and what would it be expected to achieve? In particular, why might a non-functional prototype interface be built rather than a more complete prototype? **(5 marks)**

c) Various diagramming notations can be used at the specification stage. Describe **ONE** such system giving a simple example of its use. **(6 marks)**

d) Of the various ways of creating a program or system specification, one is known as a "Formal Specification". Describe what is usually meant by a formal specification and give an example of a situation where a formal specification would be used, or preferred, over a less formal specification. **(5 marks)**

2. *a)* Describe **FIVE** functions of modern operating systems. **(10 marks)**

b) Windows and Unix are usually quoted as examples of operating systems with contrasting kinds of user interface. For these two operating systems (*or any other two with contrasting user interfaces with which you are familiar*)

i) Outline their differences by briefly describing their user interfaces.

ii) Describe **ONE** difference between the two operating systems that is not concerned with the user interface. **(7 marks)**

c) The central activity initiated by a user of an operating system is to cause it to execute a particular program with a particular file as data. Describe how this is achieved under the two types of operating system, using the two types of user interface in *b)* above, and give **ONE** advantage for each approach. **(8 marks)**

3. *a)* Novice programmers often need guidance to enable them to recognise when to use a procedure (*or function or subroutine*) in a program. How would you advise a novice in that situation? **(6 marks)**

b) Explain how it is possible to have a programmer write a procedure for inclusion in a larger program without being aware of the details of the larger program. **(5 marks)**

c) A procedural design for a computer program has split the problem into a selection between tasks A1 and A2 on the basis of test Q. Task A1 has been further broken down into a sequence of task B1 followed by B2. Task B1 is an iteration of task C1 with a pre-condition (*test at the beginning*) Q1. Task A2 is an iteration of task C2 with a post-condition (*test at the end*) Q2.

i) Present this design as a diagram (flowchart).
ii) Present the same design in procedural code (*in a language of your choice*). **(14 marks)**

4. *a)* Discuss how each of the following features, often found in program development environments, are of benefit to the programmer:

i) Colour coding of keywords
ii) Automated indentation
iii) Automated completion of code constructs **(9 marks)**

b) Describe how the following features of a debugger can be used to assist in the identification of errors within a computer program:

i) Break points
ii) Watches (also known as variable inspectors)
iii) "Step over" and "Step into" operations **(10 marks)**

c) Outline the purpose of **THREE** compiler options with which you are familiar and indicate circumstances under which each of these would be used. **(6 marks)**

5. *a)* Discuss the following statement: "The purpose of testing software is to prove it works as intended." **(6 marks)**

b) Identify **TWO** software tools that assist the testing process and for **EACH** indicate:

i) How each tool assists the testing process
ii) An example where it would be appropriate to use each tool
iii) Any disadvantages associated with using each tool **(12 marks)**

c) Describe **ONE** strategy that assists in the identification of actual test cases. **(7 marks)**

Turn over]

6. *a)* Discuss how maintainable software can be achieved. **(7 marks)**

b) A particular software development company has a small development team and concentrates on the production of a single, evolving product. Each developer works largely in isolation on different aspects of the software, writing code in their own individual style. All files are stored on a central server so that everyone has access to all code. When a developer has an idea for new functionality, they take a copy of the code from the central server, update it, test it themselves and then copy the modified files back to the server. If a customer requests a change to the software, the developer with responsibility for the appropriate aspect of the software makes the change. Again the resulting software is tested by the same developer and when complete, the new version is stored on the central server. Each developer schedules their own work, balancing new development tasks with bug fixes and customer change requests.

When customer installation disks are required, the code stored on the central server is copied onto a blank disk (along with an appropriate install routine) and dispatched. The only documentation is the code itself as the developers are told to use comments throughout their code and this is deemed sufficient within the company.

Discuss **SIX** problems associated with the situation described above and indicate how they may be resolved. **(18 marks)**