THE BRITISH COMPUTER SOCIETY

THE BCS PROFESSIONAL EXAMINATION
Diploma

OBJECT ORIENTED PROGRAMMING (Version 2 Syllabus)
20th April 2005, 2.30 p.m.-4.30 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. The Unified Modelling Language (UML) class diagram in **Figure 1** below models a software house that employs programmers and project leaders.

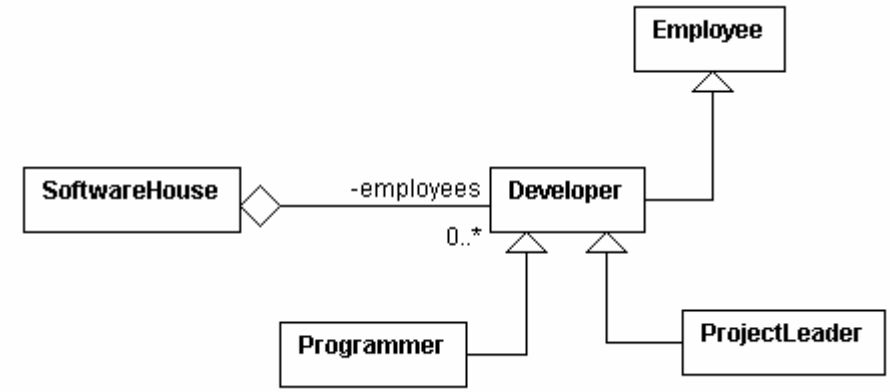

Figure 1

a) Construct a UML object diagram showing one ProjectLeader and two Programmers working for the same SoftwareHouse. Give a short explanation of the diagram. **(5 marks)**

b) If the two Programmers and the one ProjectLeader are present in the employees collection, then explain if the SoftwareHouse needs to distinguish between these differing types of developer. **(5 marks)**

c) Revise the above class diagram to introduce a Secretary class, representing an employee not involved in any software development activities. Explain the principal revisions that have been made to the diagram. **(5 marks)**

d) Revise the above class diagram so that a ProjectLeader is given managerial responsibilities for a team of developers. In your scheme explain how a management hierarchy of ProjectLeaders would be possible. **(5 marks)**

e) Revise the object diagram from part 1*a)* showing both Programmers being managed by the ProjectLeader. **(5 marks)**

2. *a)* Carefully distinguish between the terms *subclass* and *superclass*. When combined they give rise to a *class hierarchy*. Why is a class hierarchy important when modelling object oriented systems? **(5 marks)**

b) What do you understand by the term *abstract class*? **(2 marks)**

c) Draw a revision to the class diagram given in question 1, clearly distinguishing those classes that have been changed into abstract classes and explaining why they have changed. **(6 marks)**

d) What do you understand by the term *interface class*? **(2 marks)**

e) Offer a strong argument why object oriented software should be developed in terms of interfaces **(4 marks)**

f) Revise the class diagram developed in part 2*c)* to show where interfaces would be introduced into the scheme. **(6 marks)**

3. *a)* A guiding principle for object oriented development processes is that they should be:

i) Use-case driven
ii) Architecture centric
iii) Iterative and incremental

Explain what is meant by these terms. **(12 marks)**

b) Discuss the impact that an iterative and incremental development process has on the testing activity. **(5 marks)**

c) In the context of testing an object oriented system, explain what is meant by the terms:

i) Unit testing
ii) Regression testing **(8 marks)**

4. *a)* Give definitions of the following:

i) abstract data type
ii) modular programming
iii) structured programming
iv) typed languages
v) untyped languages **(15 marks)**

b) Choose THREE of the above concepts and discuss how each has contributed to the development of object oriented languages. **(10 marks)**

5. *a)* Briefly explain what is meant by the term, *Design Pattern.* **(3 marks)**

b) Explain how an understanding of *Design Patterns* helps the following people:

i) computing students
ii) inexperienced software developers
iii) experienced software developers
iv) software maintainers **(12 marks)**

c) Describe a *Design Pattern* with which you are familiar. Your answer should include the motivation for the existence of the *Design Pattern*, its structure, participants and consequences of its use. **(10 marks)**

6. *a)* A requirement of object oriented systems is to manage a collection of objects.

i) Describe how collection classes are used to realise this requirement.
ii) Give examples of two collection classes with which you are familiar.
iii) Explain how a particular collection class might maintain its objects as an ordered collection. **(12 marks)**

b) Using a suitable example, explain the essential differences between specialisation and delegation. You should use a programming language of your choice to illustrate your answer. **(5 marks)**

c) Consider the following scenario:

> A class Vector is a collection class that holds its elements in the order in which each element is added. Each element has a unique index associated with it. Indices start at 1 and increase by 1 each time a new element is added. For example, the first element added has an index of 1, the second 2 and so on. The Vector class also has a method get that is used to retrieve a particular element from a Vector. When supplied with an integer representing an index, the method get returns the element with that index. For example, get(1) would return the first element in the Vector.
>
> You are required to use the Vector class in the construction of a Queue class. This new class should mimic a queue. As with the Vector, it holds its elements in the order in which each are added. However it has a method front to return the first element added i.e. the element at the front of the queue. It also has a similar method back to return the last element added i.e. the element at the back of the queue. Crucially, it should not be possible to access elements at intermediate positions in the queue.

Discuss whether you should you use delegation or specialisation to develop the Queue class. Give a detailed explanation of the reasons for your decision. As before, you should use a programming language of your choice to illustrate your answer. **(8 marks)**