BCS THE CHARTERED INSTITUTE FOR IT

BCS HIGHER EDUCATION QUALIFICATIONS
BCS Level 5 Diploma in IT

OBJECT ORIENTED PROGRAMMING

Friday 30th September 2011 - Afternoon
Answer **any** FOUR questions out of SIX. All questions carry equal marks
Time: TWO hours

Answer any Section A questions you attempt in Answer Book A
Answer any Section B questions you attempt in Answer Book B

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

Calculators are **NOT** allowed in this examination.

Section A

Answer Section A questions in Answer Book A

A1. a) Describe **FIVE** features of the UML and explain how they are used in the design of object-oriented software.

(10 marks)

b) Describe the feature of the UML that allows a designer to describe rules which apply to the models developed as part of a design exercise.

(5 marks)

c) Give a simple example of a UML class diagram and a rule which clarifies the semantics of the situation being modelled.

(10 marks)

A2. a) There are two techniques which are commonly used to test a class, one of which uses prior knowledge of the structure of the code and the other which is based on an understanding of the specification of the tasks the class is expected to accomplish. Explain how each of these testing techniques is carried out.

(10 marks)

b) After individual classes have been tested using one or both of the techniques referred to in part a) they are often combined and tested as a group. Describe two approaches to this form of testing.

(10 marks)

c) Discuss the limitations of testing.

(5 marks)

Turn over]

A3. a) Explain the following terms:

i) Structured programming;
ii) Modular programming;
iii) Abstract data types;
iv) Typed languages;
v) Untyped languages.

(10 marks)

b) Explain how the following aspects of software design can contribute the overall quality of an object-oriented program:

i) The degree to which each class relies on another class;
ii) The degree to which each part of a class is associated with each other part;
iii) The gathering together of related aspects of a design, grouping them and hiding the implementation.

(9 marks)

c) Explain the statement “Objects belong to one *class* but may conform to more than one *type*”.

(6 marks)

Section B

Answer Section B questions in Answer Book B

B4. The class diagram in **Appendix A** represents a *Student Enrolment System* which records what courses a student takes, who their current tutor is and a record of any fees paid.

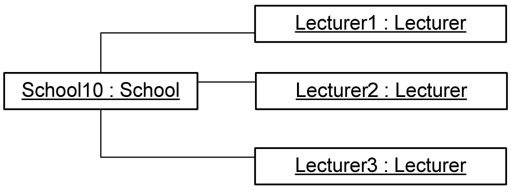
a) Describe what the diagram represents. Include all structural constraints.

(15 marks)

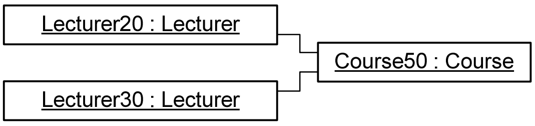
b) Given the object diagrams below (i-vi), state which are legitimate instances. If an object diagram is not legitimate explain why not.

(10 marks)

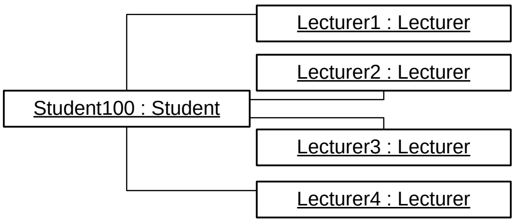
i



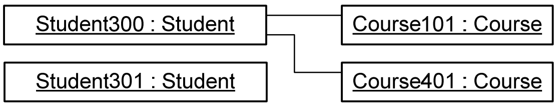
ii



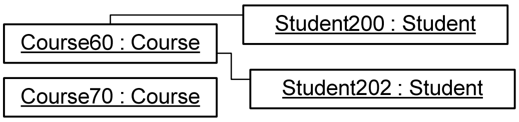
iii



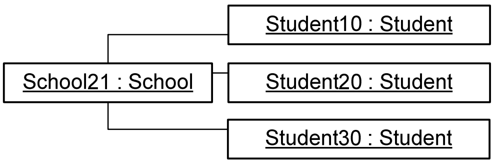
iv



v

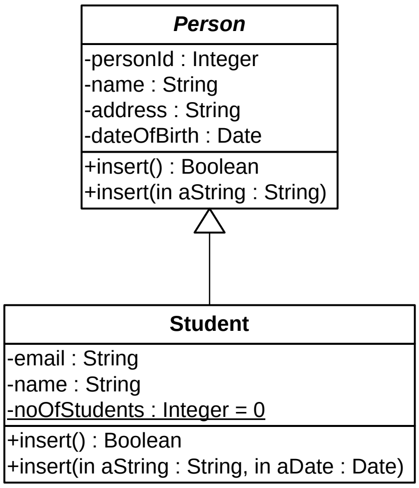


vi



Turn over]

B5. The following diagram is a development of part the design in **Appendix A**:



a) Both classes now include a definition for some methods called **insert** and have an attribute called **name**. Identify the scope of each method and any issues or problems arising from this design.

(10 marks)

b) Describe in detail **THREE** design patterns with which you are familiar. Your answer should include the circumstances in which they are applicable, when they can be applied and the trade-offs when using them within a larger design.

(15 marks)

B6. *Universal Parcels* is a company that specialises in the delivery and collection of parcels for business customers. To use their services, a customer must first register for an account. The customer needs to provide their company name, address, telephone number and the name of a main contact for any queries.

As part of the registration process, the customer will have to decide if they wish to pay monthly on receipt of an invoice, or via credit card for each delivery made. If paying by credit card, then the card details are also required. Once these details have been accepted, the customer will be issued with an account number that they must quote when contacting the company.

When a customer requires a parcel to be delivered, they will contact *Universal Parcels* to arrange collection. The customer needs to provide details of where the parcel will be collected from; where it will be delivered to; how many parcels are to be collected and which type of service they want, for example, next day delivery.

Once the collection has been arranged, an *Airway Bill* will be generated. The details on this will be used by a *Dispatcher* to schedule the vehicle needed for the collection. Each parcel will be given a priority number by the *Dispatcher* and those with the highest priority will be collected first.

By 12 noon each day, the *Dispatcher* also needs to generate a delivery schedule to ensure all the parcels are delivered according to the service required.
Each *Driver* has a mobile device with a copy of the *Airway Bill*; a person at the delivery address must sign this to say that the parcel has been delivered. This will flag that the delivery has been completed.

Once the parcel has been delivered, if the customer pays via credit card, their card will be debited by the amount required, or if they pay monthly, then the invoice account will be debited. Once a month, the *Finance Department* will generate the invoices for payment.

If the parcel cannot be delivered for any reason, it will be returned to the *Depot* and a card will be left with at the delivery address with details of how to arrange re-delivery.

a) Draw a Use Case diagram for this system.

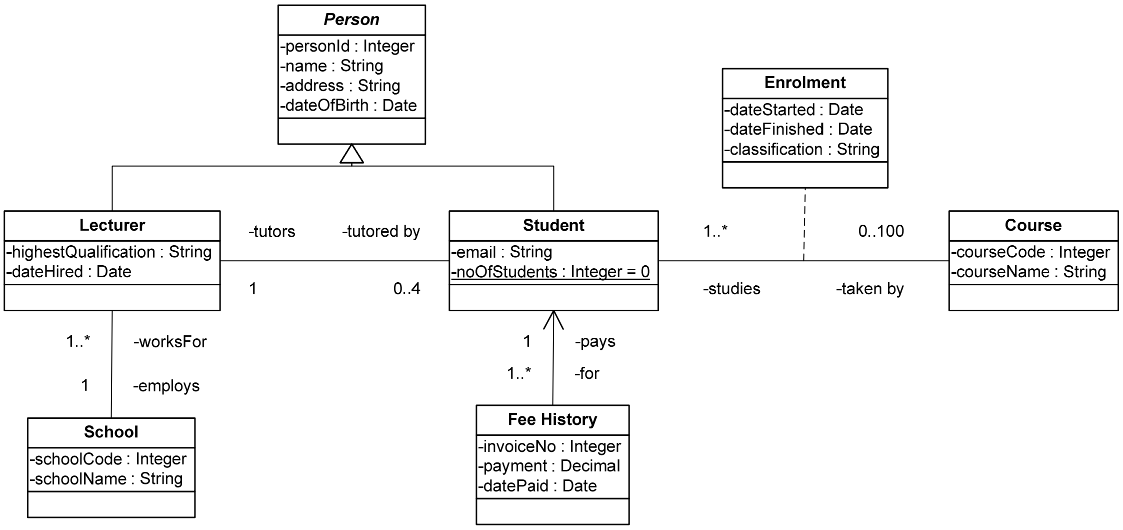
(15 marks)

b) Discuss the role of Use Cases (diagrams and descriptions) in the development of an object-oriented system.

(10 marks)

Appendix A

Class diagram used in Questions B4 and B5


Person
-personId : Integer
-name : String
-address : String
-dateOfBirth : Date
Enrolment
-dateStarted : Date
-dateFinished : Date
-classification : String
Lecturer
-highestQualification : String
-dateHired : Date
-tutors
-tutored by
1
0..4
Student
-email : String
-noOfStudents : Integer = 0
1..*
0..100
-studies
-taken by
Course
-courseCode : Integer
-courseName : String
1..*
-worksFor
1
-employs
School
-schoolCode : Integer
-schoolName : String
1
-pays
1..*
-for
Fee History
-invoiceNo : Integer
-payment : Decimal
-datePaid : Date