



**DUBLIN INSTITUTE OF TECHNOLOGY**

---

**BSc. (Honours) Degree in Computer Science**

**Year 2**

---

**SUMMER EXAMINATIONS 2014/2015**

---

**SOFTWARE ENGINEERING 2 [CMPU2020]**

MR. RICHARD LAWLOR

WEDNESDAY 20<sup>TH</sup> MAY

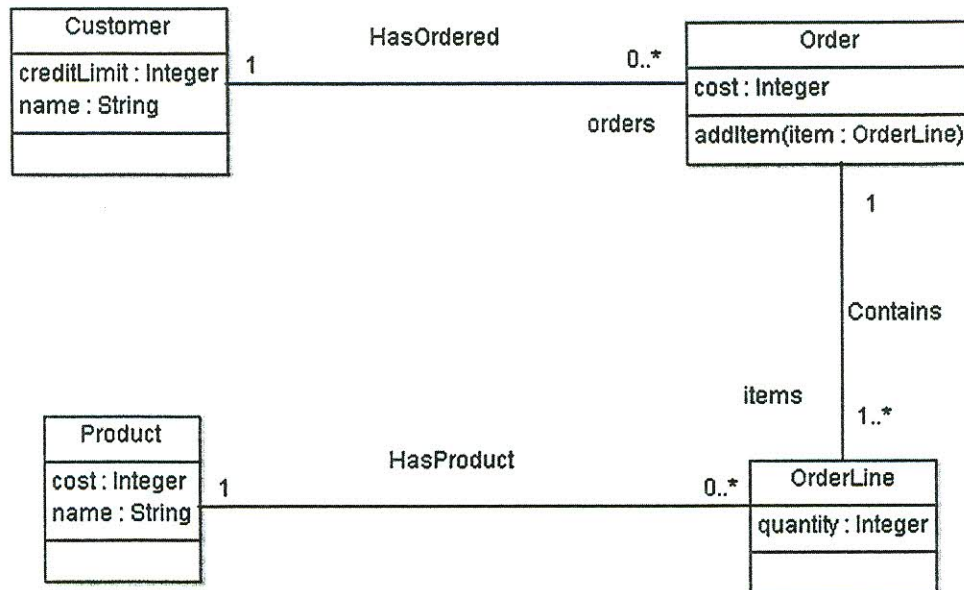
1.00 P.M. – 3.00 P.M.

**TWO HOURS**

ATTEMPT FOUR OUT OF FIVE QUESTIONS  
ALL QUESTIONS CARRY EQUAL MARKS

1. (a) Discuss the pros and cons of class inheritance versus interface inheritance in building flexible and loosely coupled reusable software. (12 marks)
- (b) Describe a design pattern which might be helpful if you wish to design a class which allows only a single instance. In what situation would such a design be useful?
- Provide some code fragments to indicate how such a class could be implemented. (13 marks)
2. You are required to do some object-oriented design for a standalone restaurant software system that mainly manages bookings. The restaurant software should be able to handle advance reservations, walk-in bookings, assigning tables to reservations and so on.
- (a) A layered architecture allows for separation of concerns. Explain what is meant by this. Then describe an appropriate layered architecture for the restaurant system given that it will be implemented as standalone software. (6 marks)
- (b) Describe the structure and purpose of the Observer design pattern and in which generic situations it might be applicable. What role could it play in the design of the restaurant system? (7 marks)
- (c) Show and explain in writing, with the aid of a package/class diagram and a sequence diagram, how the Observer pattern could be incorporated into the design of the restaurant system to couple application code with user interface code. (12 marks)

3. The following class diagram is an incomplete UML specification. OCL will be used to specify some additional business rules and in order to test the specification it has been decided to model it in USE (UML Specification Environment).



- (a) The business rules to be added to the specification are listed below. Show how they can be expressed in OCL.
- Before adding any order line to an order, the quantity of the particular product has a maximum value of 10.
  - When adding a new order line to an order with *addItem()*, the customer credit limit must be greater than or equal to the cost of the order line plus the existing cost.
  - When a new OrderLine object *item* has been added to an order, then:  

$$\text{orderLines\_after\_operation} = \text{orderLines\_before} \cup \{\text{item}\}$$
- (9 marks)
- (b) Briefly explain how the preconditions and postcondition in part (a) could be tested in USE with the help of !openter and !opexit.
- (8 marks)
- (c) Provide a rough Java or C# implementation of the Order and OrderLine classes paying particular attention to the *Contains* association. Outline how preconditions for *addItem()* could be incorporated in the code.
- (8 marks)

4. (a) Explain what is meant by *Design by Contract* (DbC).

(8 marks)

(b) Comment on DbC from the following viewpoints:

- i) Precision versus Detail
- ii) DbC and Quality Assurance

(9 marks)

(c) Write an appropriate contract in Spec# for a method `int ISqrt(int x)` that computes the integer square root of x.

Supposing you then implemented the method in C#, what could be done with the Spec# contract?

(8 marks)

5. (a) Describe six of the key practices of the agile methodology XP.

(12 marks)

(b) Discuss the diagram below from the point of view: Anticipatory Design versus Refactoring.

(13 marks)

