W228/203

DUBLIN INSTITUTE OF TECHNOLOGY

BSc (Honours) Degree in Computer Science

Year 2

WINTER EXAMINATIONS 2014/2015

Software Engineering 1 [CMPU2019]

Internal Examiners
Mr R Lawlor
Dr D Lillis

External Examiner
Mr P Collins

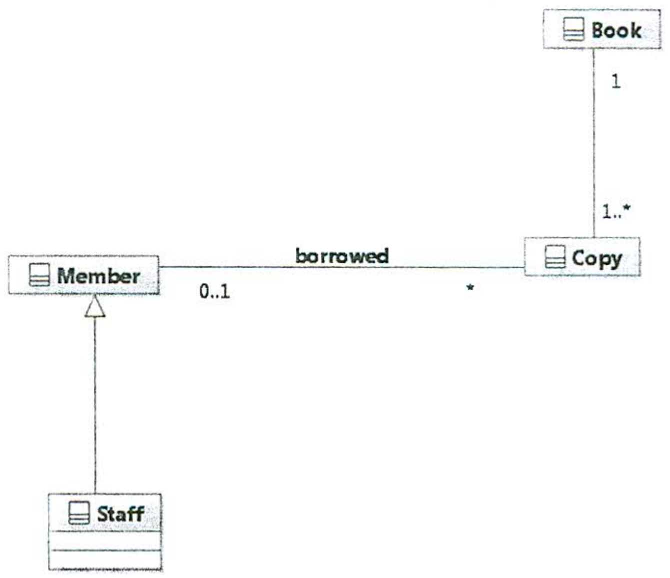
Thursday 8th January 4.00 p.m. – 6.00 p.m.

Exam duration: 2 hours

Answer FOUR questions out of FIVE

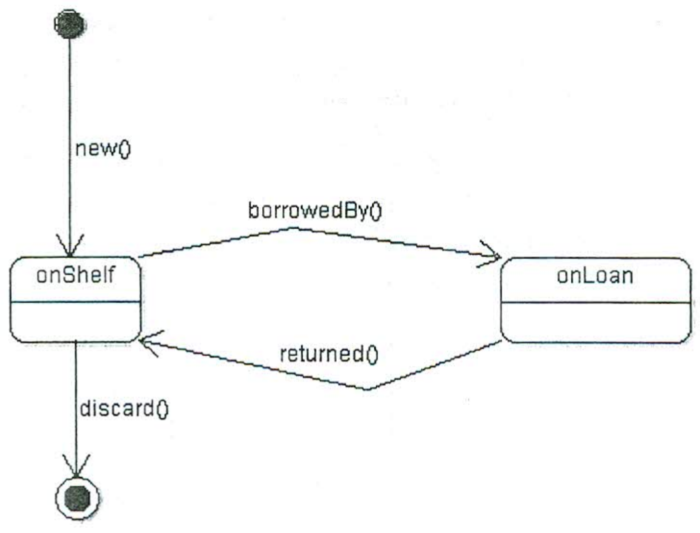
All questions carry equal marks

1. (a) Given the following class diagram for a Library software system, modify it so that library members can also reserve books. Comment on your modification.



(10 marks)

(b) Given the following state chart for copy of a book in a library system, refine it so that a reservations can be taken into account. Make sure to include any appropriate guard conditions.



(7 marks)

(c) Draw sequence diagrams which show the object interactions when a library member reserves a book.

(8 marks)

2. (a) What is a use-case?
List three significant advantages and a potential disadvantage in using use-cases.
(10 marks)

(b) Mention two other roles use-cases may have besides requirements description.
(5 marks)

(c) Provide an use case description for the following 2 related use cases:

- borrow book
- borrow book and pay fine

and draw a corresponding use case diagram.
(10 marks)

3. (a) Outline the stages of the *waterfall* process model and then discuss the major problems associated with it.

Is the waterfall process model suitable for any type of software development?
(15 marks)

(b) Comment on four aspects in which *Iterative and Incremental* processes can help overcome some of the issues connected with the waterfall process.
(10 marks)

4. (a) Briefly explain what is meant by the terms *modularity, cohesion* and *coupling* within the context of software design and programming and then discuss their relevance. How do they relate to Object-oriented concepts like: classes/object, data hiding and encapsulation?
(16 marks)

(b) Describe three types of coupling.
(9 marks)

5. (a) Provide a conceptual level class diagram for the C# code fragments below.

(7 marks)

(b) Provide a class diagram which represents the actual structure of the code fragments below.

(8 marks)

(c) A UML class diagram can be interpreted at the conceptual level or at the implementation level. Describe what this means with reference to the diagrams for parts (a) and (b) of this question.

(10 marks)

```
abstract class Customer {
    string name;
    string address;
    ISet<Order> orders;
    public void addOrder(Order o) {
        orders.Add(o);
    }
    public abstract string getCreditRating();
    public string getName() { return name;}
}

class PersonalCustomer : Customer {
    string creditCardNumber;
    public PersonalCustomer(string n, string a, string ccn) {…}
    public override string getCreditRating() {return "poor";  }
}

class Order {
    Customer customer;
    List<OrderLine> lineItems;
    string number;
    Date dateReceived;
    bool isPrepaid;
    Money price;

    public Order (Customer c, string n, Date d) {
        customer = c;
        c.addOrder(this);
        number = n;
        dateReceived = d;
        price = new Money(0,0);
        lineItems = new List<OrderLine>();
    }

    public void addItem( Product p, int quantity) {
        if (p.available(quantity) == true) {
            OrderLine item = new OrderLine(p, quantity);
            lineItems.Add(item);
            price.add(item.getPrice());
        }
    }
    public void show() {
        …
    }
}

class OrderLine {
    int quantity;
    Money price;
    Product product;

    public OrderLine( Product p, int q) {
        product = p;
        quantity = q;
        price = p.take(q);
    }

    public Money getPrice() { return price;}

    public void show() {… }
}

class Product {
    string name;
    int stockLevel;
    Money unitPrice;

    public Product(string n, int q, Money up) {
        name = n; stockLevel = q; unitPrice = up;
    }

    public bool available( int q) {
        return q <= stockLevel;
    }

    public Money take(int q) {
        stockLevel = stockLevel - q;
        Money price = new Money(q, unitPrice);
        return price;
    }
    public void show() {… }
}
```