National Exams - December 2018

17-Comp-B11, Advanced Software Design

3 hours duration

NOTES:

1. If doubt exists as to the interpretation of any question, the candidate is urged to submit, with the answer paper, a clear statement of any assumptions made.
2. No Calculator permitted. This is a closed book exam with up to 2 aid sheets allowed written on both sides.
3. You are requested to answer:
   a. Any five (5) questions in PART I
   (only the first five questions of PART I as they appear in your answer book will be marked)
   b. Any three (3) questions in PART II
   (only the first three questions of PART II as they appear in your answer book will be marked)
   c. Any four (4) questions in PART III
   (only the first four questions of PART III as they appear in your answer book will be marked)
   d. Any two (2) questions in PART IV
   (only the first two questions of PART IV as they appear in your answer book will be marked)
   e. Any five (5) questions in PART V
   (only the first five questions of PART V as they appear in your answer book will be marked)
4. All questions have equal weight.

PART I—General principles

Question 1

Differentiate between the "Waterfall lifecycle model", the "Iterative and Incremental lifecycle model" and the "Rational Unified Process (RUP) lifecycle model." How are they related (differences and similarities)?

Question 2

Explain the following terms related to software requirements, i.e., provide definitions and examples:

a. functional requirement
b. quality requirement
c. process requirement
d. implementation requirement
e. business requirement

Question 3

Why quality requirements should be quantifiable? Illustrate your answer with examples.

Question 4

A use case diagram in requirement analysis consists of actors, use cases and their relationships. Describe how classes in object-oriented software design can be developed based on the use case diagram of an application.

Question 5

In software design, what is the relationship between "design decisions" and "design alternatives?" What is the main basis or major concern in selecting design alternatives to make design decisions? Why documented rationales of design decisions in terms of selection of design alternatives are important?

Question 6

In software testing, what test cases can be developed before software is implemented or coded, and what test cases can only be developed after software is implemented or coded?

Question 7

What are two major tasks in software development for software reuse?
List three or more benefits of using reusable components?
What are the challenges of producing reusable software?

PART II—Design by Contract

Question 8

In software development, what are pros and cons of using design by contract approach? What are pros and cons of using defensive programming approach? Explain and provide examples.

Question 9

In design-by-contract based programming, what are common programming technique used to verify if a contract is fulfilled? What is difference between using such technique in programs and defensive programming?

Question 10

In designing a method or function using design by contract, what is a precondition, and what is a post-condition of the method or function?

In designing a class, what is a class invariant? Why is class invariant important?

Question 11

The Liskpv substitution principle states that behaviours or class invariants of a superclass should not be altered by its subclasses. In designing a superclass in object-oriented programming without knowing its subclasses, how can the principle be implemented in the design?

Question 12

Give an example class design which has class invariant(s). Explain the class invariant(s) of the example class.

PART III—Patterns

Question 13

Design patterns are described in pattern languages with standard structures and contents. Describe one of the "Gang of four" design patterns using a pattern language with standard structure. Explain the meaning of each section in the structured description.

Question 14

The "Gang of four" design patterns are classified as creational, structural or behavioural. What do these terms mean? Give one example design pattern of each kind and justify the classification of each pattern.

Question 15

Use examples to describe the differences between the Decorator design pattern and the Proxy design pattern.

Question 16

Use examples to describe the differences between the Observer design pattern and the Mediator design pattern.

Question 17

Suppose you are asked to design an application that checks the validity of credit cards. For simplicity, let us consider only two types of credit cards—Visa, MasterCard. The application carries out a series of validations on the input credit card information as a series of four steps, which are always all carried out in the same order:

Step 1—verify the expiration date;
Step 2—verify the length of the credit card number;
Step 3—verify that the credit card number has valid characters;
Step 4—check whether the account is in good standing.

These steps are always performed in the same order but are obviously performed differently depending on the credit card.

Describe how you can use the Template Method design pattern to implement the above application. Draw the basic class structure of your pattern-based solution.

PART IV—GUI Design

Question 18

You are asked to design a stock monitoring application. The application takes a sock name from a user and display the current value of the stock graphical and textual forms. Draw a UML class diagram to show how you can use the Model-View-Controller (MVC) architectural pattern to implement the application.

For each design goal below, indicate whether the MVC architecture may help or hurt. Justify your answer in each case.

- extensibility of the system
- response time
- modifiability of the design

Question 19

During Analysis and Design, engineers separate class responsibilities into Entity, Control and Boundary classes, the latter being responsible for interfaces between the system and the actors (e.g., human actors). In other words, a Boundary class represents a GUI (or part of it) when the actor is a human. (The other two kinds of classes are responsible for storing data and executing use cases, respectively.)

Explain why using the above design strategy can help improve quality and efficiency of other implementation, testing and maintenance development/lifecycle phases.

Question 20

Use examples to explain how using the Command design pattern and Decorator design pattern can help design high quality GUI for software applications.

Question 21

Software usability has three goals for software applications: easy to learn, easy to use, and help prevent and/or fix user errors. Assume that you are asked to develop a hospital patient registration system. Describe your design strategy for achieving each of the usability goals for the system.

PART V—C++/Java and Modular Programming

Question 22

One of the important object-oriented programming principles is "programming to interface, not to implementation." Use C++ or Java code examples to explain this principle.

Question 23

Explain the following terms: Encapsulation, Information hiding, and Message passing.
How Information hiding can be implemented by Encapsulation in class design, and by Message passing in class coupling design?

Question 24

Explain the following terms and how they relate to one another:

- Polymorphism
- dynamic binding
- overloading
- overriding.

Question 25

What are pros and cons of using friend functions and friend classes in C++? For each pro or con case, use an example to explain your answer.

Question 26

Another important object-oriented programming principle is "Favor object composition, over class inheritance.". Draw UML class diagrams to show the object composition based Adapter design pattern and the class inheritance based Adapter design pattern. Explain the above principle based on the diagrams.

Question 27

C++ supports multiple class inheritance, and Java only supports single class inheritance but allows multiple inheritance of interfaces. Describe how multiple class inheritance can be implemented by using single class inheritance and multiple interface inheritance in Java.

Question 28

An input variable defines the price for a product. The specification indicates that the price must be greater than or equal to $10.00 and cannot be greater than $1,200.00.
Explain that among the following 5 groups of input values for unit testing the function, why group C is the best choice for the testing.

A 6.00 9.00 55.00 900.00
B 8.50 150.00 1200.00
C 5.12 10.00 500.00 1,200.00 1,500.30
D 5.00 600.00 1300.00 1500.00
E 10.00 150.00 1300.00