National Examinations December 2019

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 ONE aid sheet 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
Briefly describe agile development methodologies and their advantages and disadvantages. Briefly describe conventional document-driven process models and their advantages and disadvantages.

Question 2
Use one or more examples to explain functional requirements of software systems. Use one or more examples to explain non-functional requirements of software systems.

Question 3
What is software metrics in software development? In software design, the major concern is to design a quantifiable high-quality software system. Explain the relationship between "quantifiable" and "software metrics".

Question 4
Assume that you are asked to develop a personal financial management application which supports a user to add, edit, calculate, save and load personal financial transaction information. Draw a UML use case diagram to show user requirements of the application. Draw a UML class diagram to show your design of the application. Briefly explain how your design is based on or implement the use cases.

Question 5
In designing a software system, design decisions are usually made based on quality concerns. Give 2 example quality concerns and explain how design decisions can be made differently based on either of these 2 quality concerns.

Question 6
In terms of software testing, test cases can be created in different development phases including requirements, design, and implementation phases. For the following 3 types of software testing: white-box testing, black-box testing, and grey-box testing, in which development phase which type of testing cases can be created? Explain your answer.

Question 7
What is software reuse? Why software reuse is important? Briefly describe major activities for software reuse during software design and implementation phases.

There are two aspects in software reuse in software development. (i) A software development project can use existing reusable software modules or components. (ii) A software development project can produce reusable software modules/components for others to use. Give an example for (i) and an example for (ii).

PART II—Design by Contract

Question 8
What is design-by-contract in software design? Give a simple example to illustrate your answer. What are pros and cons of using design-by-contract methodology in software design.

Question 9
In design-by-contract based programming, we usually use assertions to guarantee correctness of programs. What is assertion in programming? How assertion is used in design-by-contract based programming? Use an example to illustrate it.

Question 10
In design-by-contract based software design, in designing a software module, we usually specify preconditions and postconditions of the module. Briefly describe how such specified preconditions and postconditions realize design-by-contract in designing a module. Use an example to illustrate your answer. What is the relationship between preconditions/postconditions and assertions?

Question 11
What is class invariant in object-oriented software design? Give an example class design which has class invariant(s). Explain the class invariant(s) of the example class. How class invariant is used in design-by-contract?

Question 12
What is Liskpv substitution principle in software design? Give a simple design example that follows the principle. Give a simple design example that violates the principle.

PART III—Patterns

Question 13
Use an example design pattern to explain what a design pattern is, and why design patterns are important in software design.

Question 14
The "Gang of four" design patterns are classified as creational, structural, and behavioural. What do these three terms mean? Give an example design pattern of each kind and justify its classification.

Question 15
Use examples to describe the differences between the Factory Method design pattern and the Abstract factory design pattern.

Question 16
Use examples to describe the differences between the Decorator design pattern and the Strategy design pattern.

Question 17
Assume that you are asked to develop a personal financial management application which allows a user to add, edit, calculate, save, and load personal financial transaction information. You are also asked to use the Façade design pattern to design the application. Draw a diagram to show your design of the application and explain where and how the Façade design pattern is implemented in the design.

PART IV—GUI Design

Question 18
Assume that you are asked to develop a personal financial management application which allows a user to add, edit, calculate, save, and load personal financial transaction information. 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 in your design may help or hurt. Justify your answer in each case.

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

Question 19
In object-oriented software design, what are boundary classes? What are control classes? What are entity classes? How is the "separations of concerns" software engineering principle applied in such classification of design classes? Among the three types of classes, which type of classes involves GUI design? Using your design for the personal financial management application to illustrate these three types of classes.

Question 20
Use examples to explain how using the Command design pattern and Memento 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 personal financial management application which allows a user to add, edit, calculate, save, and load personal financial transaction information. Describe your design strategy for achieving each of the above usability goals for the system.

PART V—C++/Java and Modular Programming

Question 22
One of the important object-oriented programming principles is "Favor object composition, over class inheritance." Use C++ or Java code examples to explain this principle.

Question 23
In designing a class in C++ and Java, we can visibility of a class, variable, or method? What is visibility? Use Java or C++ code to explan. Why visibility is important to support encapsulation and modularity software engineering principles?

Question 24
Give sample code in C++ or Java to illustrate the following terms in object-oriented software design and programming:

- polymorphism
- dynamic binding
- overloading
- overriding.

Question 25
Using example C++ code to show how a C++ class can allow other classes access its private variables and methods, which is not supported or allowed in Java. What are advantages and disadvantages of the C++ approach? In your opinion, why Java does not allow it.

Question 26
Another important object-oriented programming principle is "Programming to interface, not to implementation". Use simple example code in C++ or Java to explain this principle.

Question 27
Java supports "interface" in addition to "class", but C++ has only "class". What is the difference between interface and class? How interfaces in Java can be simulated or implemented in C++ using classes?

Question 28
Briefly describe how modular programming can be implemented in programming using C++ and Java. Use examples to illustrate your answer.