December 2016 98-Comp-A6

Software Engineering

3 Hours Duration

Notes:

1. If doubt exists as to the interpretation of a question, the candidate is urged to submit with the answer paper a clear statement of any assumptions made.

2. No calculators permitted. This is a closed book exam.

3. Answer any five of the eight questions.

4. Any five questions constitute a complete paper. Only the first five questions as they appear in your answer book will be marked.

5. All questions have equal weight.

Marking Scheme:
1. (a) 5 marks; (b) 5 marks; (c) 10 marks
2. 20 marks
3. 20 marks
4. (a) 5 marks; (b) 5 marks; (c) 10 marks
5. (a) 5 marks; (b) 5 marks; (c) 10 marks
6. (a) 5 marks; (b) 5 marks; (c) 10 marks
7. (a) 5 marks; (b) 5 marks; (c) 10 marks
8. (a) 6 marks; (b) 6 marks; (c) 8 marks

Total mark out of 100

Question 1. *The Software Development Process.*

(a) List the stages of the software development life cycle and briefly describe each stage.

(b) In percentage of total effort, how much effort does each stage require on average in industry? Explain your answer.

(c) Contrast and compare these stages to the stages of building and owing a house. Comment on how good the analogy is between the software development process and the processes of building and owning the house.

Question 2. *Software Design.*

A software system is to be developed for a microprocessor-based *Insulin Delivery System* (IDS) in a hospital. The system works by using a micro-sensor embedded in the patient to measure blood parameters that are proportional the sugar level. These parameters are then sent to a pump controller. This controller computes the sugar level, judges how much insulin is required and sends signals to a miniaturized pump to deliver the insulin via a permanently attached needle.

Using a *function-oriented* approach, derive a design for the HSS described above. Make reasonable assumption and clearly state them.

Question 3. *Object-Oriented Software Design.*

Identify possible objects in the following system and develop an object-oriented design for it. Make and state reasonable assumptions about the system when deriving the design.

> A group diary and time management system is intended to support the timetabling of meetings and appointments across a group of co-workers. When an appointment is to be made that involves a number of people, the system finds a common slot in each of their diaries and arranges the appointment for that time. If no common slots are available, it interacts with the user to rearrange his or her personal diary to make room for the appointment.

Question 4. *Software Testing.*

(a) Explain why testing can only detect the presence of errors, not their absence.

(b) What is regression testing? Explain how the use of automated tests and a testing framework simplifies regression testing.

(c) Give a set of test cases for the following components:

- a sorting routine which sorts arrays of integers.
- a routine which takes a line of text as input and counts the number of non-blank characters in the line.
- a module designed to read in a date expressed using the format YYYY/MM/DD, where YYYY is the year (exactly 4 digits), MM is the month (1 or 2 digits allowed), and DD is the day (1 or 2 digits allowed). Spaces are to be ignored. Thus, valid entries might be 2011/ 1/ 1, 1990 /12/25, or 1250/03/18.

Question 5. *Component-Based Software.*

(a) What is component-based software engineering?

(b) Why is it important that all component interactions be defined in terms of "requires" and "provides" interfaces?

(c) Design possible interfaces of components that might be used in a system for an emergency control room. The interfaces are for a call-logging component that records calls made, and a vehicle discovery component that, given a postal code and an incident type, finds the nearest suitable vehicle to be dispatched to the incident.

Question 6. *Software Verification and Validation.*

(a) Discuss the difference between *verification* and *validation*. Explain why validation is a particularly difficult process.

(b) Explain why it is not necessary for a program to be completely free of defects before it is delivered to its customers. To what extent can testing be used to validate that the program is fit for its purpose.

(c) Using your knowledge of Java, C++, C, or some other programming language, derive a checklist of common errors (not syntax errors) that could not be detected by a compiler but that might be detected by program inspection.

Question 7. *Distributed Software Systems.*

(a) Explain why distributed systems are inherently more scalable than centralized systems. What are the likely limits on the scalability of distributed systems?

(b) What is the difference between a fat-client and a thin-client approach to client-server systems development?

(c) Using a distributed object approach, propose a distributed architecture for a national store e-commerce system, where users can check product availability and buy products online. State any assumptions you make clearly.

Question 8. *Software Maintenance.*

(a) Explain why a software system that is used in a real-world environment must change or become progressively less useful.

(b) Describe three types of software maintenance, substantiating each type with an example.

(c) Do software engineers have a professional responsibility to produce code that can be readily evolved, even if their employer does not explicitly request this? Justify your answer.