UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR08019 INFORMATICS 2C - INTRODUCTION TO SOFTWARE ENGINEERING

Tuesday 16 th December 2014

09:30 to 10:30

INSTRUCTIONS TO CANDIDATES

Answer QUESTION 1 and ONE other question.

Question 1 is COMPULSORY.

All questions carry equal weight.

CALCULATORS MAY NOT BE USED IN THIS EXAMINATION

Convener: D. K. Arvind
External Examiner: C. Johnson

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

1. **You MUST answer this question.**

(a) The following statements (that may appear in a requirements specification) are unclear and ambiguous. Please explain why the statements are unclear and ambiguous, and then rewrite the statements so that they can be objectively evaluated (that is, we can test an implementation and determine if the statements have been met). You may make any additional reasonable assumptions about the requirements. You need to reformulate the statements below in a better way.

Stmt 1: The response time should be minimized.

Stmt 2: The alarm should be raised quickly after a high fuel level has been detected.

[*13 marks*]

(b) You are working on a software project that will sell tickets for the University Symphony Orchestra. In particular, the application will have the following features: Administrators will be able to add an event, for example a performance of Beethoven's 9th Symphony on Friday, April 23, 2014, at 7pm. The administrator will be able to specify where the performance will be held, e.g., Reid Concert Hall. Users will be able to view upcoming performances. Users will be able to view details of each performance, such as the pieces to be played and the performers playing them. Users will be able to check for available seats to a performance. Users will be able to purchase tickets to a performance. Users can buy more than one ticket at a time, in which case they will be given adjacent seats. So if a user asks for two tickets, they will not be given one seat in the front and another in the back.

Draw a UML Use-Case diagram showing the main use-cases of the system. Show the system boundary, use-case(s), and actor(s). [*9 marks*]

(c) You are buying a server for your e-business and you cannot afford that it is down for more than Y hours per year. What reliability measure would be of most interest to you?
(more than one answer may be correct, pick all that apply).

a. Availability

b. Probability of failure on demand

c. Rate of fault occurrence

d. Mean time to failure

[*3 marks*]

2. (a) You will be designing a controller for a lift. The lift can be at one of two floors: Ground or First. There are two buttons that control the lift - UpButton and DownButton. Also, there are two lights in the lift that indicate the current floor: Red for Ground, and Green for First. At each time step, the controller checks the current floor and current input, changes floors and lights in the obvious way. Assume the Up button and Down button cannot be pressed at the same time. Draw the state machine for this controller. [*12 marks*]

   (b) Which one of the following coverage criteria always requires strictly more test cases than the others?

   a. Statement Coverage

   b. Branch coverage

   c. Path coverage

   d. None of the above [*2 marks*]

   (c) Briefly describe the difference between throwaway prototyping and evolutionary prototyping. [*5 marks*]

   (d) When we discuss software testing, we refer to Faults and Failures. Briefly describe what a Fault is and what a Failure is. Make sure to point out the difference between a Fault and a Failure. [*6 marks*]

3. (a) Briefly describe evolutionary development process model for software development and the main stages in the process (consider drawing a figure to illustrate your description). [*5 marks*]

(b) Given an example of a system (or portion of a system) that is suitable for evolutionary development? Briefly mention why this example is suitable for evolutionary development. [*4 marks*]

(c) For the following function,

```
 void findMinMax(int array[], int length, int min, int max)
{
   if (NULL!=array) then
   {
      min=max=array[0];
      for (int i=1; i < length; i++)
      {
         if (min > a[i]) then min=a[i];
         if (max < a[i]) then max=a[i];
      }
    }
}
```

a) Develop test-cases that will provide statement coverage. [*3 marks*]

b) Develop test-cases that will provide branch coverage. [*4 marks*]

c) Is it possible to develop test-cases that will provide full-path coverage? Briefly motivate your answer. [*4 marks*]

d) Modify the program to introduce a fault so that you can demonstrate that even if we could achieve full path coverage we would not guarantee to reveal all faults. [*5 marks*]