Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Software Engineering II

Date: Tuesday 26th May 2009
Time: 09:45 – 11:45

Please answer any THREE Questions from the FOUR Questions provided
Use a SEPARATE answer book for each Section.
This is a CLOSED book examination

The use of electronic calculators is NOT permitted.
Section A

1. The following program (written in a Java-like language) searches for the greatest common divisor of two given integers. For example, the common divisors of 12 and 18 are 1, 2, 3 and 6. Therefore, their greatest common divisor is 6.

```java
int GreatestCommonDivisor (int m, int n) {
    int r ;
    if (n > m) {
        r = m ;
        m = n ;
        n = r ;
    }
    r = m % n ;
    while (r = 0)  {
        m = n ;
        n = r ;
        r = m % n ;
    }
    return n ;
}
```

m % n denotes m modulo n, that is the remainder when m is divided by n.

a) Draw the control flow graph of the program. (5 marks)

b) What is the equation which computes the McCabe cyclomatic complexity metric? Using this equation, specify the number of independent paths in the basis set for the above program. (2 marks)

c) Specify a set of independent paths for the above program that would constitute a basis set according to your answer to (b). An independent path can be specified using a sequence of numbers referring to the lines in the program, for example, (1,2,3,...). (3 marks)

d) For any two of the independent paths you identified in your answer to (c), specify a test case that forces its execution using a table of the form shown below:

<table>
<thead>
<tr>
<th>Test case</th>
<th>n</th>
<th>m</th>
<th>Expected result</th>
<th>Actual result</th>
<th>Covered path</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(6 marks)

e) What do the contents of the table you produced for (d) tell you about the above program? (1 mark)

f) In your opinion, what part of the behaviour of the above program is not tested using the independent paths identified in your answer to (c)? Justify your answer. (3 marks)
Imagine you are leading a programming team to develop software for an internet auction company. Your team has decided to develop a library of basic modules, as a basis for a range of applications in internet auctioning.

a) In developing the library, why would you use Design by Contract? (5 marks)

b) What programming language would you choose, and why? (5 marks)

c) The auction company provides online services for its registered customers. Using the language of your choice, outline a module for an online customer account, together with its contracts. You can assume that each customer can offer a new item for auction, accept a bid (for an item offered by him), or bid for an item offered by other customers. You should give a clear explanation of the outline code and the contracts. (10 marks)
3. Recall the revisited badminton club example from the course book (including its exercises) and the lectures. Here are the given sets, axiomatic description, and the state schema, presented as though the revised specification was written from scratch.

\[\text{[STUDENT]}\]

\[
\begin{align*}
\text{maxPlayers: } & \mathbb{N} \\
\text{maxPlayers} &= 20
\end{align*}
\]

\[
\text{ClubState}
\]

\[
\begin{align*}
\text{Badminton} & : \mathbb{P} \text{ STUDENT} \\
\text{Hall} & : \mathbb{P} \text{ STUDENT} \\
\text{OnCourt} & : \exists \text{ STUDENT} \\
\text{Waiting} & : \text{iseqSTUDENT}
\end{align*}
\]

\[
\begin{align*}
\text{Hall} & \subseteq \text{badminton} \\
\#\text{hall} & \leq \text{maxPlayers} \\
\langle \text{onCourt}, \text{ran waiting} \rangle & \text{ partition hall}
\end{align*}
\]

The variable \textit{badminton} records which students are members of the club.
The variable \textit{hall} records which students are in the hall.
The variable \textit{onCourt} records which students are playing on the court.
The variable \textit{waiting} records which students are waiting to play, with head (waiting) being the student at the front of the queue.

In the parts below, you should not schema-include any corresponding schemas from the first visit to the badminton club example, rather you should appropriately define the schemas from scratch.

a) Present a schema which defines the normal case of the operation \textit{EnterHall} in which a person \(p\) enters the hall. The person must be a member of the club and not already in the hall, which must not be full. He or she joins the back of the queue. You should explicitly specify the preconditions and postconditions. \(4 \text{ marks}\)

b) Present a schema which defines the normal case of the operation \textit{LeaveHall} in which a person \(p\), who must be in the waiting queue, leaves the hall. You should explicitly specify the preconditions and postconditions. \(4 \text{ marks}\)

(Question 3 continues on the following page)
c) Present a schema which defines the normal case of the operation \textit{NewGame} in which a new game is started on the court. There must be no persons on the court beforehand. At least two people must be waiting to play. If there are four or more people waiting, then the number who play must be four. Otherwise it can be two or three. The players must comprise the person at the front of the queue, plus the right number of people selected from the next five positions in the queue. (6 marks)

d) Present a schema which defines the normal case of the operation \textit{FinishGame} in which the players leave the court. There must be some players on the court beforehand. These players must join the back of the queue, but in some non-determined order. (6 marks)

4. Every now and then I hear a new piece of music that I love, so I rush to my favourite music shop and buy the manuscript to add to my collection. I can never play it at first, but after a few painful sessions (especially painful for my family), I finally declare that I have learnt it. Occasionally I get really fed up with a piece of music, and in a fit of temperamental rage I tear the manuscript into a shower of confetti, and forget all about it forever!

Here is the given set and state schema for this specification.

\[\text{MUSIC}\]

\[
\begin{array}{c}
\text{MyMusic} \\
\text{collection : } \mathbb{P} \text{ MUSIC} \\
\text{learnt : } \mathbb{P} \text{ MUSIC} \\
\text{notLearnt : } \mathbb{P} \text{ MUSIC} \\
\text{learnt} \cup \text{notLearnt} = \text{collection} \\
\text{learnt} \cap \text{notLearnt} = \emptyset
\end{array}
\]
a) Present a schema to define the initial state of the system after which all three state variables are empty. (2 marks)

b) Present a schema to define the normal case of the operation Collect, in which a piece of music \( m \) is collected by me. It must not already be in my collection, and it is always not yet learnt. (3 marks)

c) Present a schema to define the normal case of the operation TearUp, in which a piece of music \( m \) is destroyed by me. It must be currently in my collection, and it may or may not have been learnt. (3 marks)

d) Present a schema to define the normal case of the operation Learn, in which a piece of music \( m \), becomes learnt by me. The piece of music must already be part of my collection, and must not have already been learnt. (3 marks)

Totalize your operations, using the style in which they will return an outcome value of type \( \text{RESULT} \). For this, you will need the following free type definition:

\[
\text{RESULT} ::= \text{success} | \text{alreadyCollected} | \text{notCollected} | \text{alreadyLearnt}
\]

You will also use the schema, SuccessMessage, defined as:

\[
\text{SuccessMessage} \ \\
\text{Outcome! : RESULT} \\
\text{Outcome! = success}
\]

e) Present a schema, AlreadyCollected, with an input \( m \) which must be part of my collection, and an output \( \text{outcome!} \) which has the value alreadyCollected. (2 marks)

f) Present a schema NotCollected defined appropriately in a similar way. (2 marks)

h) Present total definitions for Collect, TearUp and Learn. (3 marks)

END OF EXAMINATION