ROYAL HOLLOWAY, UNIVERSITY OF LONDON
BSc EXAMINATION 2021

CS2800: Software Engineering
CS2800R: Software Engineering – FOR FIRST SITS/RESIT CANDIDATES

Time allowed: **TWO hours**

Please answer **ALL** questions.

- Handwrite your answers on paper, and write your candidate number and the module number at the top of each page. Photograph/scan the pages and keep the original paper versions, as they may be required by the examiners.
- For each question you attempt, please clearly state the question number.
- Please DO NOT include your name or Student ID anywhere on your work.
- **Academic Misconduct:** We will check all assignments for academic misconduct. Suspected offences will be dealt with under the College's formal Academic Misconduct procedures. Please remember:
  - The work submitted is expected to be your own work and only your work. You may not ask for help from any source, or copy anyone else's work.
  - You must not give help to anyone else, including sending them any parts of the questions or copies of your solutions.
  - You must not discuss the questions or solutions with anyone else.
- **Submitting your work:**
  - Your document must be submitted through Moodle using the submission link in the module Moodle page. If possible please convert your document into a PDF document to make the submission process quicker and easier.
  - Emailed submissions will not be accepted.
  - **You must complete your exam upload within 1 hour of the exam finish time.**

1. For each of the following pairs of related software engineering concepts you must:

   - Briefly *describe* each of the two concepts and why they are important in software engineering. This should be enough to introduce the concept to a new student on CS2800.

     Each description could be about six lines of text.

   - Show that you understand how the two concepts are *connected*.

     For example, they have the same or contrasting goals, or they may be techniques that rely on each other to work. This needs careful thought.

     A good answer could be about four lines of text.

   (a) *Primitive Obsession Smell* and *Low Cohesion*. [12 marks]

   (b) *Regression Testing* and *Smoke Testing*. [8 marks]

   (c) *Cherry Pick Merge* and *Back-porting*. [12 marks]

   (d) *Waterfall Development* and *Agile Development*. [8 marks]

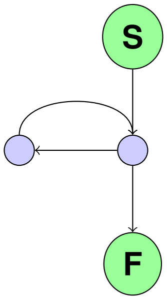
2. (a) Why does TDD tend to introduce the Once and Only Once Smell, and what mechanisms does JUnit provide for dealing with this problem? [2 marks]

   (b) Consider each of the following *suggested* programming standards for Java. In each case, explain carefully whether or not it would be a useful standard that could be enforced by Checkstyle. [6 marks]

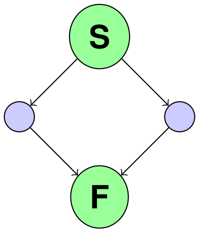
   - Always use braces around the body of a `while` statement.
   - Always use *meaningful* names for variables.
   - Always surround operators (like $+$ or $*$) with spaces.
   - Never use a starred import statement like `import javafx.scene.*`.

   (c) Briefly describe *variable watching*, *breakpoints* and *stepping over* in the symbolic debugger in Eclipse. [3 marks]

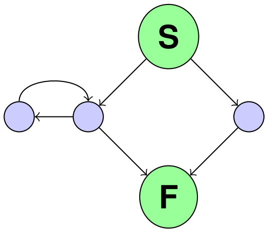
   (d) Write simple Java code snippets that have the following flowgraphs.



(i) [2 marks]



(ii) [2 marks]



(iii) [3 marks]

3. (a) For each of the following statements, state whether it is true or false and give one or two sentences justifying your answer.

i. It is normal to check out a folder from an SVN repository rather than the whole repository. [2 marks]

ii. Every time a file is saved in your working copy the updated version is automatically stored on the repository. [2 marks]

iii. A minor release is made by copying the trunk to a new candidate release branch. [2 marks]

iv. It is necessary to perform `svn update` before committing. [2 marks]

v. The SVN log does not show who made a particular `commit`. [2 marks]

vi. It is possible for a user to remove an old commit to an SVN repository so that a mistake made does not show in the history. [2 marks]

vii. The SVN `merge` command updates the working copy and does not change the repository. [2 marks]

viii. It is not sensible to store all the files you have in an Eclipse project folder in your SVN repository. [2 marks]

(b) You have branched `https://svn.CS2800/branches/my_feature` from the trunk at `https://svn.CS2800/trunk`, and have not merged your work back to the trunk.

You have committed changes to `Fred.java` and `Bert.java` in this branch.

Only `Fred.java` and `Greta.java` have been changed in the trunk.

Your changes to `Fred.java` do not overlap those made in the trunk.

The archive is at revision 73 and no-one else is working using SVN today.

Carefully describe what happens when the following commands are executed. Be sure to mention the working revision of files in your working copy. [5 marks]

```
svn co https://svn.CS2800/trunk
svn merge https://svn.CS2800/branches/my_feature
svn status
svn commit
```

(c) Explain (with a different reason in each case) why each of the following SVN log messages is a sign of poor Software Engineering.

i. “Fixed the JavaDoc in some files so that they pass CheckStyle.” [2 marks]

ii. “Merged in Andy’s work.” [2 marks]

iii. “The code now works.” [2 marks]

4. This question is about improving the design of the following HR system.

Each employee has a name (String), salary (float), and employee number (int).
They also have an employment history which is a list of jobs.
A job has a start date (LocalDateTime) and a position.
A position has a title (String) and a minimum and maximum salary (float).
A position also has a list of zero or more immediately junior positions and sometimes has a promotion position.
Some positions are management positions.
Each department has a name (string), a boss (manager) and a staff list.
An employee can get promoted, a pay increase or fired.

There are five classes: Employee, Manager, Department, Position and Job.

(a) Draw a UML class diagram for this model. Include appropriate multiplicities. Associations may be from a class to itself.

Include relevant attributes and responsibilities for each class. Generic responsibilities like toString will not get any credit and should be omitted. [4 marks]

(b) Sometimes users mis-type the name of the new position when promoting an employee, causing confusion.

Identify the primitive obsession which allows this user error and describe a simple change to the design to fix the smell. [2 marks]

(c) A system enhancement gives each position a history (dated list) of minimum and maximum salaries. When a new maximum or minimum is entered every employee who has an instance of a position with that title in their employment history may need that object to be updated.

Explain how a `PositionFactory`, that returns references to `static Position` objects, removes this design flaw. Your answer must describe changes to existing classes and any new classes that are required. [4 marks]

(d) Explain carefully how the `Visitor` pattern could be used to provide a total salary cost by visiting each department. Your answer should describe the `visit` and `accept` methods for classes `Department` and `Employee`. [5 marks]

END