Lab assessment for Software Design and Modelling

Week 6, Semester 1, 2016

Introduction

This assessment is to be done individually on your DICE machines during the lab session, making use of the tools you have used in the lab sessions so far.

In order to be able to give students with schedules of adjustments extra time, the exercise has been designed to be done in 75 minutes. Those given extra time in exams may use the remaining time, according to their usual arrangements.

The assessment will be marked out of 20. It is in three parts.

- Part A is worth 10 marks. You can check it yourself using the methods described below. These checks are not definitive – they don't check everything – but if what you submit passes the checks below, you can expect to get at least 8 marks. Part A is intended to take no more than 30 minutes, provided you understand the work and the tools well. You will submit your work, together with the transcript of your checking.
- Part B is worth 6 marks. You are advised not to work on it until you are confident that you have done Part A well. It is intended to take no more than 30 minutes for those who understand the work and the tools very well, but might take up to 45 minutes with a few false steps.
- Part C is worth 4 marks. It is intended to challenge students who find parts A and B easy and do them fast: as you see, a first-class mark can be obtained without attempting it. You are advised not to attempt it *unless and until* you feel you have done Parts A and B well.

To submit

Some questions ask you to submit specific files. To do this, use a terminal, navigate to the directory containing the file, and use examsubmit to submit it, e.g.

```
examsubmit Foo.java
```

Others ask you to submit your entire Eclipse project. To do this, in Eclipse, use menu File → Export → General → Archive file. (NB do *not* use a Papyrus-specific export menu!) Use the Select All button to select all your files. (It does not matter if some of them are irrelevant.) Use the Browse button to choose where to save your archive; be sure to give it the required name. Then use examsubmit to submit it, as above, e.g.

```
examsubmit A1.zip
```

Part A

1. Using Papyrus, and calling your project A1, draw a UML class diagram showing:

   (a) A class Customer, with private attributes name of type String and balance of type Integer (use the UML primitive types in this assessment, not the ecore ones);

   (b) A class Order;

   (c) An association between Customer and Order, named places, navigable from Customer to Order only;

   (d) Multiplicities indicating that each Order is linked to exactly one Customer, while each Customer is linked to one or more Orders.

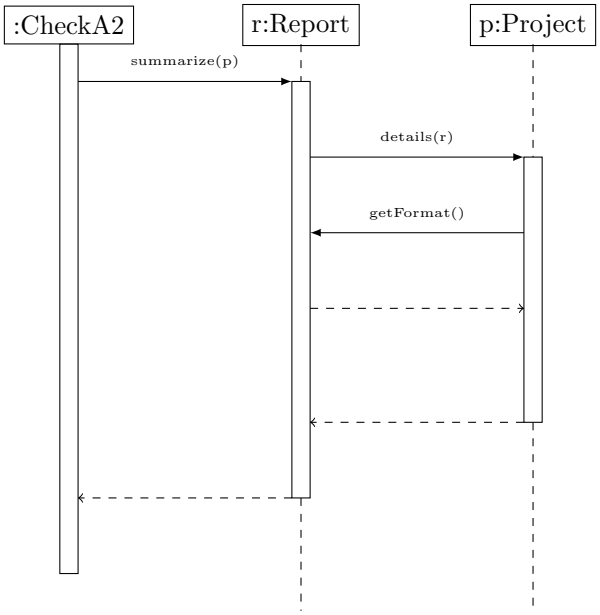
   You can check your answer by following the instructions at `file:///group/examreadonly/sdm/`. **Submit file checkA1.txt**.

   Export your project as A1.zip as described above. **Submit A1.zip**.

   [6 marks]

2. Consider the following sequence diagram. Write classes Report and Project that are consistent with it. (Do not write the class CheckA2!)

   - Your code must compile correctly and must be in the default package (no `package whatever;` statements).
   - Assume that each of the methods shown is public and has return type String.
   - Keep it simple! You will not need any attributes or constructors, or any methods other than those mentioned in the diagram.



You can check your answer by following the instructions at `file:///group/examreadonly/sdm/`. **Submit file checkA2.txt**.

Submit Report.java and Project.java.

[4 marks]

Part B

1. Consider the following situation.

   Each undergraduate student has a personal tutor; a personal tutor may have up to 25 tutees. Personal tutors meet with their students several times a year. Some meetings are individual; others are in groups. A group meeting, which involves up to 10 students, needs to be scheduled in a meeting room, while an individual meeting takes place in the tutor's office.

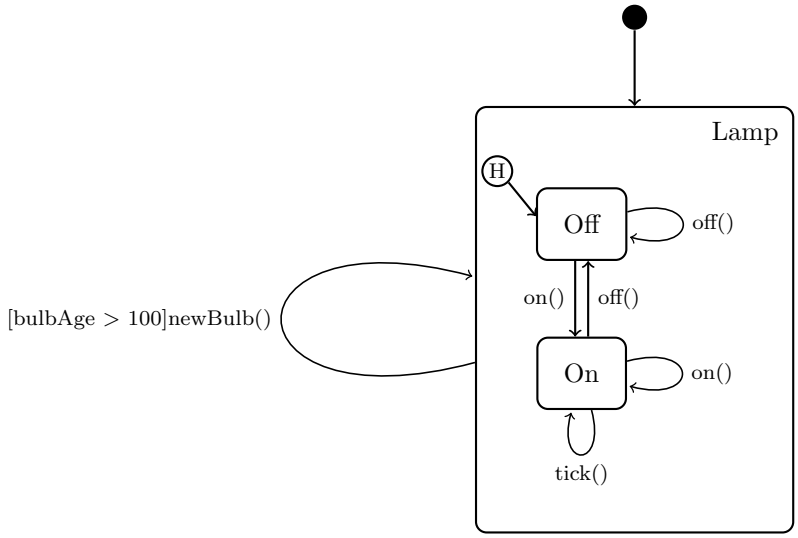
   In Papyrus, develop a conceptual class model for this situation, in a project called B1. Include multiplicities and attributes, where you can deduce them from the above, but do not concern yourself with navigabilities or operations.

   Export your project as B1.zip as described above. **Submit B1.zip**.

   [4 marks]

2. Consider the Java code in file Lamp.java, in conjunction with the protocol state diagram shown below. The Java and the UML are in-

consistent in several ways. Assuming that the UML is correct, modify the Java code to be consistent with it. Include comments to explain, briefly, what you have changed and why.



Submit your modified version of Lamp.java.

[2 marks]

Part C

1. Consult the UML2.5 standard, provided at `file:///group/examreadonly/sdm/`. According to UML, a class is not allowed to inherit from itself, either directly or transitively, e.g., it's not allowed that class A inherits from class B, and class B inherits from class A. But where in the standard is this rule specified?

   (a) Find the relevant place in the standard. In the text file C1.txt, state the fully-precise section number (e.g. 1.2.3.4).

   (b) Still in file C1.txt, explain where you found the rule in terms of the structure of the standard. That is, if you found the rule in section 1.2.3.4, give the titles of Section 1, Section 1.2, Section 1.2.3, and Section 1.2.3.4; briefly explain the purpose of each of these sections; and say what the words in these titles have to do with classes and inheritance, to explain why the rule is found where it is.

   Submit file C1.txt. [2 marks]

2. Consider again the situation described in part B1. Make a copy of your project (use the Copy and Paste from the menu you get by right-clicking the project name) and name it C2. Using your choice of

one or more appropriate UML diagrams (including, at least, a modified version of the class diagram), design functionality for scheduling meetings. Briefly explain your work in text file C2.txt, and submit this along with any relevant diagrams.

Export your project as C2.zip as described above. **Submit C2.zip**.

Submit file C2.txt.

[2 marks]

Summary

If you did every part of this assessment, the files you should have submitted are:

- checkA1.txt
- A1.zip
- checkA2.txt
- Report.java
- Project.java
- B1.zip
- Lamp.java
- C1.txt
- C2.zip
- C2.txt

Later submissions override earlier ones, so if in doubt, resubmit.