PGT Exam Performance Feedback 2016/2017 Semester 1

COMP61511 Software Engineering Concepts in Practice Bijan Parsia

Comments Please see the attached report.

COMP 61511: Software Engineering Concepts *In Practice*
Fall 2015 Exam General Feedback

Bijan Parsia, Course Leader and Marker

Exam Overview

The exam consisted of

- 23 objective questions (7 true/false (TF), 22 multiple choice questions (MCQs))
  - 1 point per question for a total of 29 point
- 2 essay questions
  - 1 worth 5 points and 1 worth 6 points for a total of 11 points

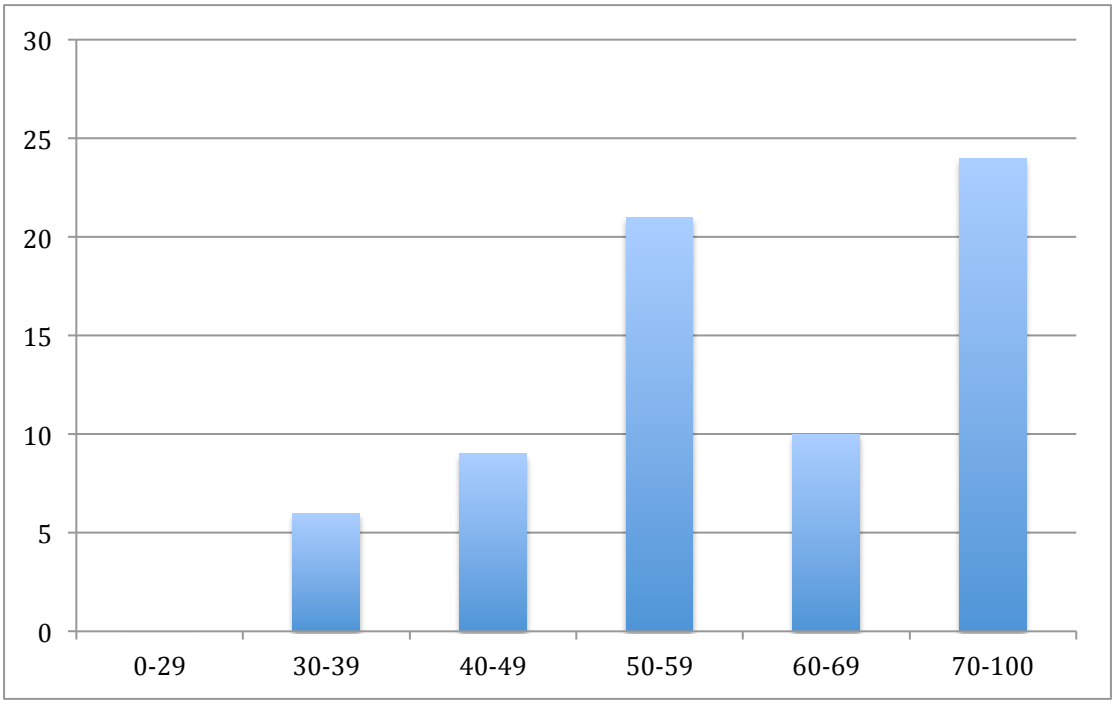
Total possible points = 40

Overall Performance

Performance was pretty standard for a class this size (60% average is the school "norm") and sort and was essentially unchanged from last year:

	2015-2016		**2016-2017**	
	Raw Score	Percentage	**Raw Score**	**Percentage**
Minimum Value	15	38%	13	33%
Maximum Value	34.5	88%	36	90%
Range	19.5	-	23	-
Average	24.34	62%	24.3	61%
Median	24.75	63%	23.5	59%
Std Deviation	4.39	11%	5.7	14%

The distribution is interesting with a bit of bimodality:



As you can see, the largest group was the distinction level with almost a third of the class scoring that. This is a bit different from last year where the distribution was more normal (though skewed right). We had fewer essays this year and that may have some role in this shift. But in general, the cohort taking this class has a wide range of ability and experience which can lead to these sorts of patterns.

Blackboard provides an “item response theoretic” analysis of the questions:

Test Summary

40.0	31	0	70	24.32	01 hr 51 min
Possible Points	Possible Questions	In Progress Attempts	Completed Attempts	Average Score	Average Time

Discrimination

19	Good Questions
9	Fair Questions
3	Poor Questions
0	Cannot Calculate

Difficulty

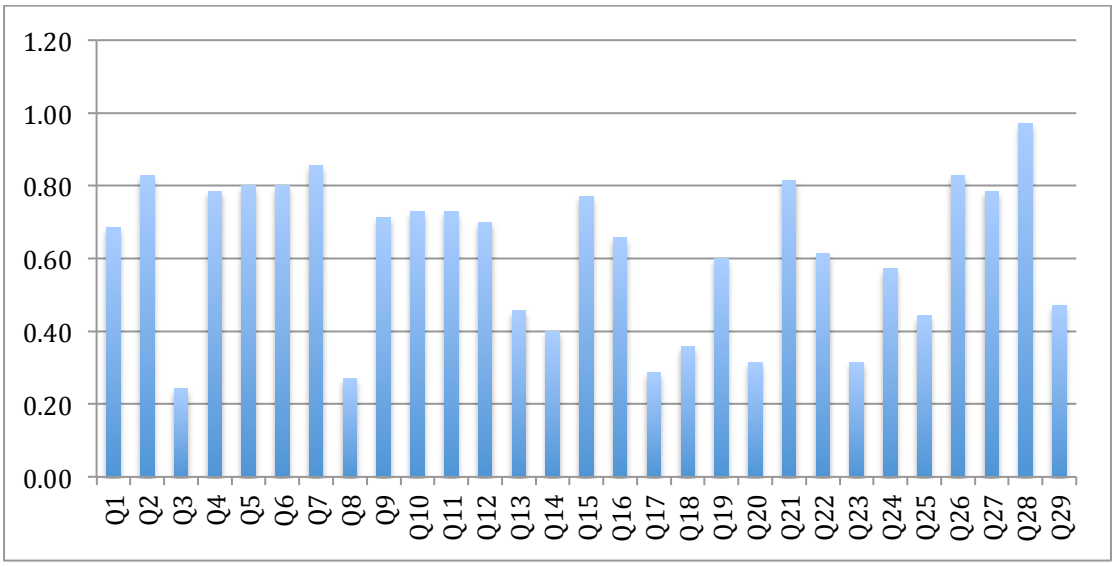
5	Easy Questions
23	Medium Questions
3	Hard Questions

This is perfectly normal and almost the same as last year. People finished a bit quicker (1hr 51 min vs. 1hr 55 mins).

In terms of difficulty breakdown, the strong majority of the questions are in the (desired) "medium" level, which means that between 30-80 of exam takers got it correct. Generally, instead of having the entire exam at the "medium" level, I prefer to have a bit of progression from some easy, to mostly medium, to some hard, which this exam reflects.

Discrimination (i.e., whether student performance on a particular question "tracks" their performance on the rest of the exam -- a question where students who did overall poorly on the exam did better than students who did overall well on the exam exhibits poor discrimination) was also quite reasonable, though there're a bit too many "fair" quality questions.

MCQs



Q3 was about the visibility of internal qualities to end-users. They are *sometimes* visible to end users…consider how end users can request a system be written in Java.

Q8's most attractive distractor is that the bug is in the system under test. But recall that a failing test might itself be buggy (or the test harness might be buggy).

For Q14, usability is plausible as the "key" non-functional quality, but efficiency dominates still.

Q17 was about walking skeleton. The key aspect of a walking skeleton is that it's "end to end", i.e., covers the whole system.

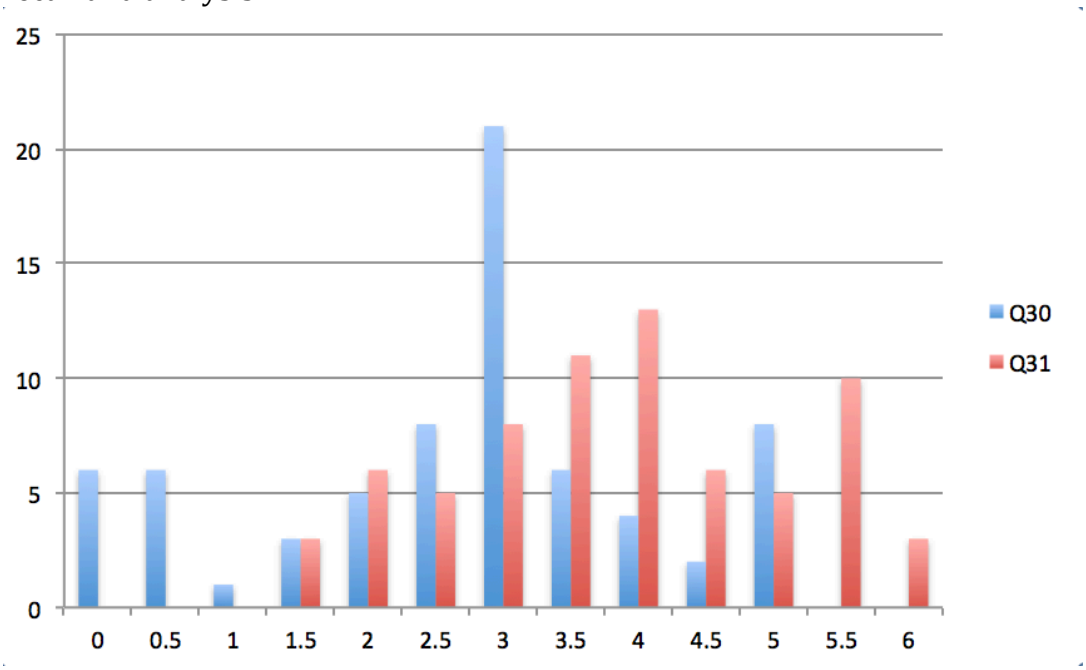
Q18 is about intentional debt. While it's a good idea for intentional debt to have a plan to pay it off, that isn't required for it to be intentional.

Q20: Refactoring (in it's purest for) precludes any other construction activity (debugging, adding functionality, etc.)

Q23 was intended as a fun, recall question about Grace Hopper.

Essays

As the distribution of marks show, people found Q30 significantly more difficult than Q31. In both cases, the rubric followed the basic pattern established in the coursework with two of the marks being for Mechanics and for Answering the Question with the rest for the question specific content. Both questions involved recall and analysis.



Q30: The biggest issue here was that quite a few people did not recall what McConnell's General Principle of Software Quality was (i.e., increasing software quality decreases cost). This was pervasive enough that I shifted the rubric to allow more than 3 marks if, in spite of getting the principle wrong, the rest of the question was reasonable.

Q31: The biggest issue with this question was that people didn't *relate* the wicked problem nature of software development to the *order* of the stages. People would just discuss what the stages *are* which isn't really related to the question.