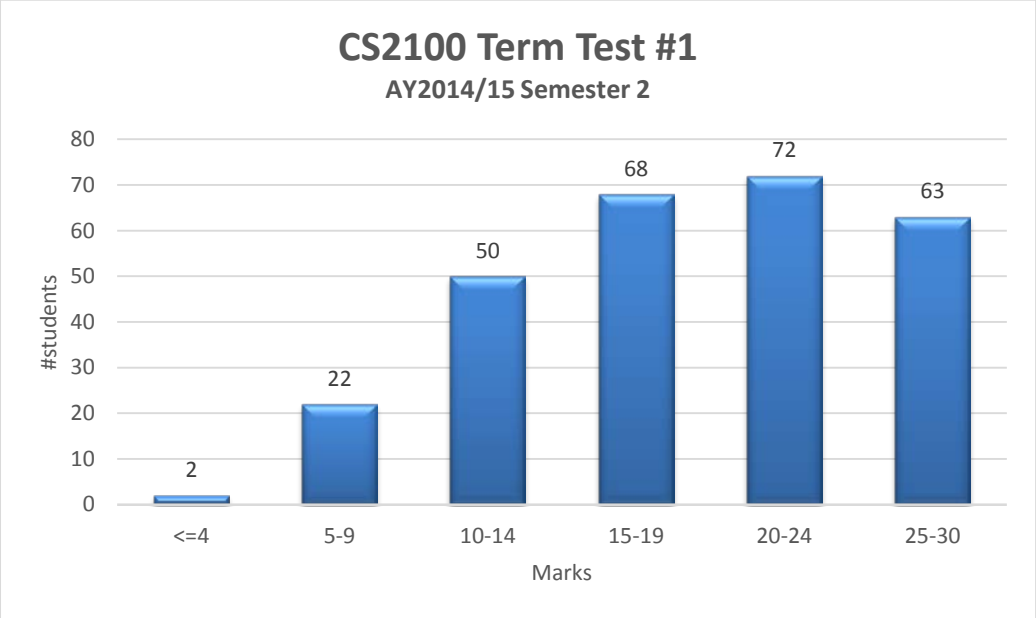
Comments on CS2100 Term Test #1 (AY2014/15 Semester 2)

1. Summary of results

277 students sat for the test yesterday. The mean is 18.85 out of 30 marks, or 62.8%, which is higher than the mean of last year semester 2's test (56.4%), but comparable to the means of AY2012/13 and AY2011/12 semester 2' tests, which was around 64%.

I set this test paper with the aim of getting a mean between 63% and 65%, so it missed the lower end by just a little.

The distribution of the marks, in brackets of 5 marks, is shown in the chart below.



I don't think it is a difficult paper. I would urge the 74 students who scored below 15 marks, and even more so for those who scored below 10 marks, to work really hard and revise the materials for the final exam, which has about 30% on part 1 (digital logic design) and 70% on part 2 (computer organisation).

Yuen Jien and I hope to see better results from you for Term Test #2.

(And to those who wrote interesting stuffs in the speech bubble, thank you! They have provided me with comic relief during my marking and I enjoyed reading them. I've shared a few of them on the wall of my facebook as well as CS2100 facebook group. ☺)

The answers are on **EduTech "Materials" → "Term tests"** folder.

You may check your marks on the IVLE gradebook. We will return your answer sheets to you at your next tutorial.

2. Discussion on MCQs

The table below shows the percentage of students who chose the correct answers, and of those who chose the most popular wrong answers:

	Q1	Q2	Q3	Q4	Q5
%students who chose the correct answer	D (50.2%)	B (58.1%)	E (39.4%) Hardest	D (66.8%)	A (82.3%) Easiest
%students who chose the most popular wrong answer	C (15.9%)	C (22.0%)	B (25.3%)	A (13.4%)	E (10.1%)

I intended Q1-Q3 as give-away questions – they are all questions on definitions – especially Q3 which I have "shocked" the class with a class quiz (Boolean Algebra lecture, slide 23). I thought that quiz had helped. Hence, I was very surprised, with a tint of disappointment, seeing Q3 end up as the killer MCQ. Certainly, for a function on 3 variables, $A' + B + C'$ is a maxterm, and hence it is also a product-of-maxterms expression!

Q5 surprised me for being the easiest MCQ, as I thought it might catch many students.

3. Discussion on Questions 6 – 10

The table below shows the average marks for each question, normalised as percentages.

Q6	Q7	Q8	Q9	Q10
73.8% Easiest	56.2%	68.4%	38.7% Hardest	62.9%

The results here don't come much as a surprise as they match my expectation. I would like to comment on a few questions.

Q6c: Simplify $(A \oplus B \oplus C) \oplus (A \oplus B \oplus C)'$.

I thought this is a dead give-away since it is of the form $X \oplus X'$, which must be 1. An example of a 3-second question, I would say.

Q7: IEEE 754 floating point format

A number of students made the careless mistake of neglecting that the value is negative!

Q8. $Z = A'{\cdot}B{\cdot}C'{\cdot}D{\cdot}E + (D'+B)' + B{\cdot}D{\cdot}E$

The simplified SOP expression is $D{\cdot}E + B'{\cdot}D$. Unfortunately I made a mistake in this question. The expression cannot be implemented on the given decoder without any logic gate. Hence, I have allocated all the 4 marks to the simplified SOP expression.

I have also modified the question and answer posted on the **EduTech "Materials" → "Term tests"** folder so that students will get a corrected version in future.

Q9. Using 4-bit adder to implement 6*x* + 3, where *x* is *ab*, a 2-bit unsigned number

A student wrote on the speech bubble that it took him half an hour to solve this question. Actually, if you make the correct observation, it would take you only half a minute to answer it. ☺ As I've always mentioned in class, you need to make the right observation and use the correct technique.

When you need to multiply a 2-bit unsigned number *ab* by 4 or by 2, you don't need any logic gate. Simply append 00 to the value, and that's multiplying it by 4: *ab* × 4 → ab00; or append 0 to the value, and that's multiplying it by 2: *ab* × 4 → ab0.

Hence, multiplying *ab* by 6 means *ab*00 + 0*ab*0. (Verify this yourself.) And this fits into the given 4-bit adder nicely.

How about adding 3? Just put 11 into *ab*00, turning it into *ab*11. Hence *ab*11 + 0*ab*0 is what you need, and you can connect 0 to C_{in} of the adder. This is the most intuitive solution. There are several other alternative solutions, mostly variants of this idea.

However, a number of students gave some weird answers, by dismantling the outputs *cdefg* and considering them as separate functions. For example, observing that *g* is always 1 and *e* is always the same as *a*, they connect 1 to *g* and *a* to *e*. This is very piecemeal-like and it loses the essence of the adder. Also, I think some students did it by trial-and-error. In retrospect, perhaps I should have fixed the outputs of the adder to *cdefg* to avoid receiving such answers. ☺

Q10. Sequential circuit

This is a very standard question. Just follow the steps you will definitely get the answers. Might be a bit tedious, but it is straight-forwards. Make sure you know the steps. This is the kind of questions where you will get the marks easily provided you don't make careless mistakes.

All the best for term test #2 and your final exam!

Aaron Tan
8 March 2015