COMP10001 Foundations of Computing
Final Exam Preparation

Semester 1, 2019

Tim Baldwin, Nic Geard, Farah Khan, and Marion Zalk

— version: 1593, date: May 27, 2019 —

Announcements

- Once again (SIGH!) there is code on the internet that can be used as part of a Project 3 solution, and which I can't readily take down ... we know about the code, and will pursue academic honesty charges for any student who makes use of that code in their submission ... JUST DON'T!!!
- Don't forget to provide subject feedback via the SES:
  `https://ses.unimelb.edu.au`

Announcements

- For Project 3, make sure to:
  1. submit to the tournament for Q3/Q4 (combined into a single diamond), subsequent to submitting for marking as per usual
  2. if you want an earlier submission to be used for the official tournament, make sure to resubmit this as your last submission to the tournament
  3. note that we cannot rerun the tournament, so will not accept any requests for remarking on the basis of not having submitted to the tournament — **you have been warned!**

Lecture Agenda

- Last lecture:
  - Text representation
  - Ethics and fairness in computing
- This lecture:
  - Final exam preparation

Lecture Outline

Final Details

- We will rerun the tournament after the submission deadline based on each student's final submission, and tournament results will be based on the tournament rerun
- Make sure to submit to the tournament if you want to be considered for tournament marks ... repeat, make sure to submit to the tournament
- Watch those PEP8 warnings, as they will be factored into the stylistic mark calculation

Lecture Outline

1. Project 3
2. Preparing for the Final Exam
3. Practice Exam

Tim's Tips for Exam Nirvana

- Get to the exam venue in plenty of time (go to the right place at the right time!)
- Note that the first 15 minutes of the exam are reading time; **use it wisely**
- Take your student card
- Look up your seat number ahead of time
- Take writing instruments and not much else to the exam venue
- Rug up in REB!

Tim's Tips for Exam Nirvana

- Apply for special consideration *within 48 hours of the exam* *via the Student Centre* (**not** the lecturers) if you were impeded by illness or other documentable causes
- Use the past exams as part of your preparation, but bear in mind that the pre-2015 exams are based on **Python2** (and pre-2018 exams are not Python 3.6)
- The exam focus varies from semester to semester, based on the focus of that particular subject instance, usually determined by the content of the projects (and to a lesser extent the particular guest lecturers)

What's Examinable?

- Everything from all the lectures/tutorials, with the exception of the Advanced Lecture
- *Code/maths/physics* from guest lectures are not examinable, but general computing-related concepts **are**
- We won't ask questions that require you to **write** code (from scratch) that uses recursion
- We won't ask you to **write** code to generate anything other than the most basic HTML document (e.g. simple list or table)

Lecture Outline

1. Project 3
2. Preparing for the Final Exam
3. Practice Exam

Question 1a

Evaluate the following expressions, and provide the output in each case.

```
"2,4".split(",")
```

Question 1a

Evaluate the following expressions, and provide the output in each case.

```
"2,4".split(",")
```

A: *['2', '4']*

Question 1b

```
bool(2 or "")
```

Question 1b

```
bool(2 or "")
```

A: *True*

Question 1c

```
[i%2 for i in range(3)]
```

Question 1c

```
[i%2 for i in range(3)]
```

A: *[0, 1, 0]*

Question 1d

```
sorted({(1,1): "racehorse", (2,2):
  "one too"}.items())[0][1][1:4]
```

Question 1d

```
sorted({(1,1): "racehorse", (2,2):
  "one too"}.items())[0][1][1:4]
```

A: *'ace'*

Question 2

Consider the following code fragment:

```
x = [("nic", True), ("marion", False),
     ("mariam", True), ("tim", True)]
limit = 2
i = 0
j = len(x) - 1
y = []
while i < limit:
    if x[j][1]:
        y.append(x[j][0])
        i += 1
    j -= 1
```

List the values of the variables `i` and `j` after each iteration of the loop.

Question 2 (cont.)

iteration	i	j
1	1	2
2	2	1

Question 2 (cont.)

Consider the following code fragment:

```
x = [("nic", True), ("marion", False),
     ("mariam", True), ("tim", True)]
limit = 2
i = 0
j = len(x) - 1
y = []
while i < limit:
    if x[j][1]:
        y.append(x[j][0])
        i += 1
    j -= 1
```

What is the final value of `y`?

Question 2 (cont.)

Consider the following code fragment:

```
x = [("nic", True), ("marion", False),
     ("mariam", True), ("tim", True)]
limit = 2
i = 0
j = len(x) - 1
y = []
while i < limit:
    if x[j][1]:
        y.append(x[j][0])
        i += 1
    j -= 1
```

What is the final value of `y`?

A: *['tim', 'mariam']*

Question 3

The following program is meant to determine whether a string contains only alphabetic characters, but the lines are out of order. Put the line numbers in the correct order and introduce appropriate indentation (indent the line numbers to show how the corresponding lines would be indented in your code).

```
1 assert(all_letters("abCD"))
2 if not 'A' <= letter <= 'Z':
3 return False
4 for letter in string:
5 return True
6 def all_letters(string):
7 if not 'a' <= letter <= 'z':
8 assert(not all_letters("abCD123"))
```

Question 3 (cont.)

```
6
    4
        2
            7
                3
    5
1
8
```

Question 3 (cont.)

- A couple of notes on the style of question on the actual exam
  - *Note that the provided code makes use of a semi-colon ("*`;`*") at two different locations, to combine two statements (to initialise variables) into a single line of code.*
  - You will be provided with a grid to fill:

Question 3 (cont.)

A:

6						
	4					
		2				
			7			
				3		
	5					
1						
8						