Student Number | |

The University of Melbourne
Department of Computing and Information Systems

Mid-semester Test, Semester 1, 2016
COMP10001 Foundations of Computing

Writing Time: 45 minutes

This paper has 7 pages including this cover page.

There are 5 questions in the paper, for a total of 45 marks (one mark for each minute of test time).

- All questions should be answered by writing a brief response or explanation in the lined spaces provided on the examination paper.
- It is not a requirement that all the lined spaces be completely filled; answers should be kept concise.
- Only material written in the lined spaces provided will be marked.
- Your writing should be clear; illegible answers will not be marked.
- Extra space is provided at the end of the paper for overflow answers.
  Please indicate in the question you are answering if you use the extra space.
- You should base all of your answers on Python3 (the version of Python that is used in Grok).
- Your answers can use any of the standard Python libraries, but be sure to call/import them correctly.

Authorised Materials: No materials are authorised.

Calculators: Calculators are not permitted.

Examiners' use only						
1	2	3	4	5	Total	

Question 1

[9 marks]

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

(a) `"philology"[3:6]`

(b) `'alphabet' in 'alphabetical order'`

(c) `sorted({'cycling': ['andrew', 'tim'], 'duplo': 'tim', 'dogs': 'andrew'}.keys())[-1]`

Question 2

[9 marks]

What are the final values of each of the variables indicated below, on completion of execution of the following code:

```
def fun(i, j):
    return i > j

text = "she sells sea shells".split()
text_len = len(text)
count = 0
for i in range(text_len - 1):
    if fun(text[i + 1], text[i]):
        count = count + 1
```

(a) `text_len`

(b) `i`

(c) `count`

Question 3

[7 marks]

Rewrite the following function, replacing the `while` loop with a `for` loop:

```
def max_digit_sum(maxnum):
    i = 1
    maxval = maxsum = 0
    while i <= maxnum:
        numsum = sum([int(j) for j in str(i)])
        if numsum > maxsum:
            maxval = i
            maxsum = numsum
        i  = i + 1
    return maxval
```

Question 4

[10 marks]

The following is intended to take a list of (unique) 3-digit student numbers (in the form of integers) and a list of test venues (in the form of strings), and allocate students seat numbers in the respective test venues. It does this by sorting the students in ascending order of student number, and cycling through the venues from one student to the next. It also generates a seat number for each student in a given venue, starting from seat number 1 in each venue.

```
def seat_allocation(student_ids, venues):
    seat_list = []
    ┌──────1──────┐
    next_available_seat = []
    for _i in range(venue_count):
        ┌──────2──────┐
    venue_id = 0
    ┌──────3──────┐
        seat_list.append((student_id, venues[venue_id], next_available_seat[venue_id]))
        next_available_seat[venue_id] = next_available_seat[venue_id] + 1
        ┌──────4──────┐
    ┌──────5──────┐
```

When run as follows, the provided output should be generated:

```
>>> seat_allocation([116, 562, 320, 109, 888], ["Tim Hall", "Andrew Theatre"])
[(109, 'Tim Hall', 1), (116, 'Andrew Theatre', 1), (320, 'Tim Hall', 2),
 (562, 'Andrew Theatre', 2), (888, 'Tim Hall', 3)]
```

Complete the code by providing a code snippet for each of the numbered gaps, noting that the code should be appropriate for the indicated indentation level of each gap.

(1) _____

(2) _____

(3) _____

(4) _____

(5) _____

Question 5

[10 marks]

Write the function `piig_seq(intlist)` which takes the single argument `intlist` (a list of positive integers, at least three elements in length), and returns `True` if `intlist` is a "piig" (positively increasing integer geometric) sequence of numbers, and `False` otherwise. A "piig" sequence is one where the ratio between all adjacent elements n_i and n_{i-1} is a fixed integer $r > 1$. For example, `[5, 15, 45, 135]` is a piig sequence, as the ratio between each adjacent pair of numbers is 3 (i.e. $\frac{15}{5} = \frac{45}{15} = \frac{135}{45} = 3$). On the other hand, `[3, 3, 3, 3]` is not as the ratio $r = 1$ is constant but *not* $r > 1$, `[100, 150, 225]` is not as the ratio $r = 1.5$ is constant but not a whole number, and `[7, 14, 56, 112]` is not as the ratio is not constant for all adjacent pairs in the sequence (e.g. $\frac{14}{7} \neq \frac{56}{14}$).

Example calls to `piig_seq` are:

```
>>> piig_seq([5, 15, 45, 135])
True
>>> piig_seq([3, 3, 3, 3])
False
>>> piig_seq([100, 150, 225])
False
>>> piig_seq([7, 14, 56, 112])
False
>>> piig_seq([8, 4, 2, 1])
False
```

This is blank space for further answers should you need it. Please ensure that you label the answers in this area carefully, and that you indicate on the corresponding question page that your answer can be found here.

End of Test