

CSC 108H1 F 2011 Test 2
Duration — 45 minutes
Aids allowed: none

Student Number: _____

Last Name: _____ First Name: _____

Lecture Section: L0101 & L0102

Instructors: (circle one) Craig & Horton

*Do **not** turn this page until you have received the signal to start.*

(Please fill out the identification section above, **write your name on the back of the test**, and read the instructions below.)

Good Luck!

This midterm consists of 3 questions on 6 pages (including this one). *When you receive the signal to start, please make sure that your copy is complete.* Comments and docstrings are not required except where indicated, although they may help us mark your answers. They may also get you part marks if you can't figure out how to write the code. No error checking is required: assume all user input and all argument values are valid. If you use any space for rough work, indicate clearly what you want marked.

1: _____ / 7

2: _____ / 7

3: _____ / 5

TOTAL: _____ / 19

Question 1. [7 MARKS]

Each subquestion on this page has a piece of code that is supposed to work as described in the comment statement but has a small part missing. For each one, add the missing part inside the box. Your solution must follow the instructions in the comment statement. Each subquestion is independent.

Part (a) [1 MARK]

```
d = {'k1': 'no', 'k3': 'nope'}
```

```
# Add one key-value pair to d so that the code below the box prints 'YES'.
```

```
print d['k2']
```

Part (b) [1 MARK]

```
d = {}
```

```
# Add one key-value pair to d so that the code below the box prints 'Computer Science'.
```

```
for item in d:  
    print d[item], item
```

Part (c) [1 MARK]

```
d = {(1, 2): [3, ['Anonymous', 5]]}
```

```
# Write a print statement that outputs only 'Anonymous' and does it by accessing  
# the correct piece of d.
```

In the box beside each piece of code below, write its output. If it would generate an error, say so, and give the reason for the error.

Part (d) [2 MARKS]

```
L1 = ['CSC', [108, 148, 165]]
L2 = L1[:]
L1[1].append(209)
print L1
print L2
```

Part (e) [2 MARKS]

```
L1 = ['CSC', [108, 148, 165]]
L2 = L1[:]
L2[1] = ['209', '207', '263']
print L1
print L2
```

Question 2. [7 MARKS]

Suppose we are keeping track of who is working with whom on a course assignment. We could represent the groups using a nested list of student numbers like this: `[[2, 9], [4], [3, 1]]`. (Here we use one-digit student numbers to make the example easier to read.) In this example, we have two groups of two students, and one group of one student.

Part (a) [1 MARK]

Consider the following function.

```
def no_group(group_list, class_list):  
    '''Return a list containing the student number of everyone in list class_list  
    who is not in any group according to group_list.'''
```

Write a call to the function that should return a list of length 2, and involves a class list of 6 students.

Part (b) [6 MARKS]

Now write the function. You do not need to repeat the `def` line or the docstring.

Question 3. [5 MARKS]

Write the following function according to its docstring.

Hint: Use `str.find`

```
def nth(small, big, n):
    '''Return the index of the nth non-overlapping occurrence of string small within string big.
    For example, nth('oo', 'A Cool pool look', 1) returns 3 and
    nth('oo', 'A Cool pool look', 2) returns 9. Return -1 if there are fewer than n occurrences.
    int n is > 0.'''
```

Last Name: _____ First Name: _____

Short Python function/method descriptions:

`len(x)` -> integer
Return the length of the list or string `x`.

`sum(x)` -> integer
Return the sum of the elements in the list `x`.

`open(name[, mode])` -> file object
Open a file.

`range([start], stop, [step])` -> list of integers
Return a list containing the integers starting with `start` and ending with `stop - 1` with `step` specifying the amount to increment (or decrement).

dict:

`D[k]` -> value
Return the value associated with the key `k` in `D`.

`k in d` -> boolean
Return True if `k` is a key in `D` and False otherwise.

`D.keys()` -> list of keys
Return the keys of `D`.

`D.values()` -> list of values
Return the values associated with the keys of `D`.

`D.items()` -> list of 2-tuples.
Return a list of `D`'s (key, value) pairs.

file (also called a "reader"):

`F.close()`
Close the file.

`F.read([size])` -> string
Read at most `size` bytes; with no `size`, read until EOF.

`F.readline([size])` -> string
Read next line, retaining newline; return empty string at EOF.

str:

`S.find(sub[,i])` -> integer
Return the lowest index in `S` (starting at `S[i]`, if `i` is given) where the string `sub` is found or -1 if `sub` does not occur in `S`.

`S.replace(old, new)` -> string
Return a copy of string `S` with all occurrences of the string `old` replaced with the string `new`.

`S.split([sep])` -> list of strings
Return a list of the words in `S`, using string `sep` as the separator and any whitespace string if `sep` is not specified.

`S.startswith(prefix)` -> boolean
Return True if `S` starts with the specified prefix and False otherwise.

`S.strip()` --> string
Return a copy of `S` with leading and trailing whitespace removed.

list:

`L.append(x)`
Append `x` to the end of the list `L`.

`L.index(value)` -> integer
Return the lowest index of `value` in `L`.

`L.insert(index, x)`
Insert `x` at position `index`.