

CSC 108H1 F 2016 Midterm Test
Duration — 50 minutes
Aids allowed: none

Student Number: _____

Last Name: _____ First Name: _____

Lecture Section: L5101 (W6-9) Instructor: Jacqueline Smith

*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 is double-sided, and consists of 5 questions and a list of function/method descriptions provided for your reference. *When you receive the signal to start, please make sure that your copy is complete.*

- Comments are not required except where indicated, although they may help us mark your answers. # 1: _____/ 7
 - No error checking is required: assume all user input and all argument values are valid. # 2: _____/ 4
 - If you use any space for rough work, indicate clearly what you want marked. # 3: _____/ 5
 - Do not remove any pages from the exam booklet. # 4: _____/ 4
 - You may use a pencil; however, work written in pencil will not be considered for remarking. # 5: _____/ 4
- TOTAL: _____/24
-

Question 1. [7 MARKS]

Beside each code fragment in the table below, write the printed output when the code fragment is executed. If the code would cause an error, instead write ERROR and give a brief explanation of why an error occurs.

Code	Output or Cause of Error
<pre>message = 'CSC 108' print(message[4])</pre>	
<pre>print(1 + 4 == 5 or 4/0 == 1)</pre>	
<pre>values = [3.1, 2.7] for value in values: value = value - 1 print(values)</pre>	
<pre>print('1.5' + 2.0)</pre>	
<pre>for i in range(0, 5, 2): print(i)</pre>	
<pre>size = 25 if size >= 10: print('medium') elif size >= 20: print('large') else: print('small')</pre>	
<pre>items = [0] + [2] + [4] items = items.append(1) items[1] = 3 print(items)</pre>	

Question 2. [4 MARKS]

Complete this function according to its docstring description.

```
def extract_clothing_type(item):
    """ (str) -> str

    Return the type of clothing from item, which is formatted as
    <name>: <type> or <name>: <type>, <colour> depending on if the colour is
    known.

    >>> extract_clothing_type('Jacqueline Smith: jacket')
    'jacket'
    >>> extract_clothing_type('Paul: pants, purple')
    'pants'
    """
```

Question 3. [5 MARKS]

In the function below, complete (i) a precondition the function requires, (ii) the function description, and (iii) the example function calls by adding arguments that result in the return values shown. Write your preconditions and description in the large box, and the example calls in the smaller boxes. For the example calls, there may be several correct answers, and providing any one of them will earn full marks.

```
def mystery(message):  
    """ (str) -> str
```

```
>>> mystery(  )
```

```
'!!!'
```

```
>>> mystery(  )
```

```
',' ? '
```

```
"""
```

```
result = ''  
i = 0  
while not message[i].isdigit():  
    if not message[i].isalnum():  
        result = result + message[i]  
    i = i + 1  
return result
```

Question 4. [4 MARKS]

Complete this function's body according to its docstring description.

```
def has_element_of_size(items, size):
    """ (list of str, int) -> bool

    Return True iff items contains a string of exactly length size.

    >>> has_element_of_size(['hi', 'cat', 'hello'], 4)
    False
    >>> has_element_of_size(['hi', 'cat', 'hello'], 3)
    True
    """
```

Question 5. [4 MARKS]

Consider the following function definition and additional Python statements. Beside each Python statement in the table below, write what is printed when the statement is executed. Assume all of the code above the table is run first, and then each print statement is executed in the order listed.

```
def some_function(value, num_times):
    """ (int, int) -> int """

    result = 0
    for i in range(num_times):
        result = result + value
        value = value + 1
    return result
```

```
value1 = 3
result1 = some_function(value1, 2)
value2 = 4
result2 = some_function(value2, 3)
```

Code	Output
<code>print(value1)</code>	
<code>print(result1)</code>	
<code>print(value2)</code>	
<code>print(result2)</code>	

[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]

Last Name: _____ First Name: _____

Short Python function/method descriptions:

`__builtins__`:
`int(x) -> int`
Convert `x` to an integer, if possible. A floating point argument will be truncated towards zero.
`len(x) -> int`
Return the length of list, tuple, or string `x`.
`print(value) -> NoneType`
Prints the values.
`range([start], stop, [step]) -> list-like-object of int`
Return the integers starting with `start` and ending with `stop - 1` with `step` specifying the amount to increment (or decrement). If `start` is not specified, the sequence starts at 0. If `step` is not specified, the values are incremented by 1.
`str(x) -> str`
Return an object converted to its string representation, if possible.

`str`:
`x in s -> bool`
Produce True if and only if `x` is in string `s`.
`S.count(sub[, start[, end]]) -> int`
Return the number of non-overlapping occurrences of substring `sub` in string `S[start:end]`. Optional arguments `start` and `end` are interpreted as in slice notation.
`S.find(sub[,i]) -> int`
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.index(sub) -> int`
Like `find` but raises an exception if `sub` does not occur in `S`.
`S.isalpha() -> bool`
Return True if and only if all characters in `S` are alphabetic and there is at least one character in `S`.
`S.isalnum() -> bool`
Return True if and only if all characters in `S` are alphanumeric and there is at least one character in `S`.
`S.isdigit() -> bool`
Return True if and only if all characters in `S` are digits and there is at least one character in `S`.
`S.islower() -> bool`
Return True if and only if all cased characters in `S` are lowercase and there is at least one cased character in `S`.
`S.isupper() -> bool`
Return True if and only if all cased characters in `S` are uppercase and there is at least one cased character in `S`.
`S.lower() -> str`
Return a copy of the string `S` converted to lowercase.
`S.upper() -> str`
Return a copy of the string `S` converted to uppercase.

`list`:
`x in L -> bool`
Produce True if and only if `x` is in list `L`.
`L.append(object) -> NoneType`
Append object to end of list `L`.
`L.extend(iterable) -> NoneType`
Extend list `L` by appending elements from the iterable. Strings and lists are iterables whose elements are characters and list items respectively.