CS115 Fall 2014 Test 3 —SOLUTION GUIDE—

Closed book: no textbook, no electronic devices, one sheet of paper with notes. Hand in your note sheet, with your name on it please.

Read carefully before answering! Write your answers on the test paper.

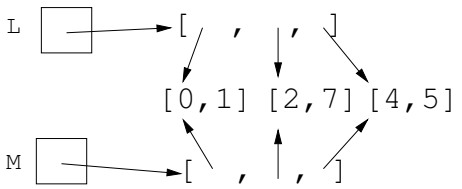
Question 1 *(15 points)* **Assess:** [state] *Show the output* that gets printed by the following code.

```
L = [[0,1],[2,3],[4,5]]
M = list(L)
M[1][1] = 7
print L
```

Also, draw a boxes-and-arrows diagram to show the data at the time `L` is printed. For integers, you can write them in place rather than as references.

SOLUTION The output is `[[0,1],[2,7],[4,5]]`.

The diagram should show that L and M reference different three-element lists, but the elements are the same references. It looks like this:



The reason is that `list(L)` makes a shallow copy of M.

Rubric: 1 (print a list of lists) + 2 (list contents correct except possibly the 7) + 5 (print 7 in the right place) + correct result printed + 1 draw some data with pointers + 3 L,M reference different lists + 3 sharing of inner lists

Question 2 *(20 points)* **Assess:** [execution]

(a) Show what gets printed by this code.

```
xs = [0,3,1,9,23,42]
ys = [0,7,3,9,0,42]
j = 0
s = 0
t = 0
print "j s t\n-------------"
print j, s, t
while j < len(xs):
    if xs[j] == ys[j]:
        t += (2*s) + 1
        s += 1
    j += 1
    print j, s, t
```

SOLUTION

```
j  s  t
-------------
0  0  0
1  1  1
2  1  1
3  1  1
4  2  4
5  2  4
6  3  9
```

Rubric: 3 some rows correct + 3 right number of rows + 2 all rows correct

(b) Yes or no: is this condition an invariant of the loop?: `s <= t`

SOLUTION Yes

Here's why (but you weren't asked to say why): Initially s and t are the same. And each time s is increased, t is increased by at least the same amount.

Rubric: 3 correct answer

(c) Yes or no: is this condition an invariant of the loop?: `j <= s`

SOLUTION No

Again, you weren't asked to say why but the answer is evident in the loop trace in part (a).

Rubric: 3 correct answer

(d) Yes or no: is this condition an invariant of the loop?: `t == s*s`

SOLUTION Yes

Notice that $(s+1)^2$ is $s^2 + 2 * s + 1$. Given that $t = s^2$, when s gets increased to $s+1$, what gets added to t is $2 * s + 1$.

Rubric: 3 correct answer

Question 3 *(25 points)* **Assess:** [design]

Implement the following function, using a loop and not recursion.

```
def takeWhile(f, L):
    '''
    Assume L is a list and f is a function that returns True or False.
    Return a new list that contains the elements of f that make f true,
    up to but not including the first element that makes f false.
    '''
```

For example, `takeWhile(odd, [1,3,2,5,3])` returns `[1,3]`.

SOLUTION

Here is one solution.

```
def takeWhile(f, L):
    '''
    Assume L is a list and f is a function that returns True or False.
    Return a new list that contains the elements of f that make f true,
    up to but not including the first element that makes f false.
    '''
    M = []
    i = 0
    while i < len(L) and f(L[i]):
        M = M + [L[i]]
        i += 1
    return M
```

Here's another.

```
M = []
for x in L:
   if f(x):
       M += [x]
   else:
       break
return M
```

And another:

```
M = []
for i in range(len(L)):
   if f(L[i]):
       M += [L[i]]
   else:
       return M
```

And another:

```
 j = 0
 while j < len(L) and f(L[j]): j += 1
 return L[:j]
```

Another:

```
j = 0
M = []
while j < len(L):
  if f(L[j]):
     M.append(L[j])
     j += 1
  else:
     j = len(L)
 return M
```

And another, by Daniel Heyman:

```
for i in range(len(L)):
    if not f(L[i]):
       return L[:i]
return list(L)
```

Rubric: 5 design at least vaguely plausible + 4 correct use of f + 4 brackets for singleton list and no index error + 4 stops after first false val + 4 uses a for-loop or while-loop + 4 does not modify contents of L

Question 4 *(15 points)* **Assess:** [execution]

This question is about the following function.

```
def merge(L,M):
    '''Assume L and M are non-decreasing lists of numbers.
    Return a new list of length containing all the elements
    of L and M, in non-decreasing order.
    '''
    R = [0] * (len(L)+len(M))  # new list to be returned,
    i = 0
    j = 0
    while i < len(L) and j < len(M):
        # invariant: R[0:i+j] contains L[:i] and M[:j] and is ascending
        #            and 0 <= i <= len(L) and 0 <= j <= m <= len(M)
        if L[i] <= M[j]:
            R[i+j] = L[i]
            i += 1
        else:
            R[i+j] = M[j]
            j += 1
    # now i==len(L) or j==len(M) so just need what's left of other list
    while i < len(L):
        R[i+j] = L[i]
        i += 1
    while j < len(M):
        R[i+j] = M[j]
        j += 1
    return R
```

Your job: make a loop trace for the first loop, in the call `merge([2,4,6],[1,3,7,9])`. That is, fill in the table below, tracing the values of the variables `i`, `j`, and `R`, assuming that `L` is [2,4,6] and `M` is [1,3,7,9]. The first row already shows the initial values of `i`, `j`, and `R`. You just need to add a row to show their values after each iteration of the first loop. (You're not required to trace the other loops.)

R	i	j
[0, 0, 0, 0, 0, 0, 0]	0	0

SOLUTION

R	i	j
[0, 0, 0, 0, 0, 0, 0]	0	0
[1, 0, 0, 0, 0, 0, 0]	0	1
[1, 2, 0, 0, 0, 0, 0]	1	1
[1, 2, 3, 0, 0, 0, 0]	1	2
[1, 2, 3, 4, 0, 0, 0]	2	2
[1, 2, 3, 4, 6, 0, 0]	3	2

Note that after the first loop, the last loop executes a couple of iterations:

```
[1, 2, 3, 4, 6, 7, 0]  3  3
[1, 2, 3, 4, 6, 7, 9]  3  4
```

But you weren't asked to write these.

Rubric: 5 there's at least one correct row + 2 each for correct row (of the five possible)

Question 5 *(25 points)* **Assess:** [class]

Notes during test:

- some of the example data has artist/track switched — but it doesn't matter, the given asserts are consistent
- what if there's more than one max? — should say *get a longest track* (we won't test for case where there are more than one)

```
# test3_Q56 - questions 5 and 6 on third exam, 2014

##############################################################################
# RULES: you can use Moodle to download this file and upload your solution.
# You can use IDLE to edit and run your program.  You should NOT look at
# other programs in IDLE, you should NOT use any other programs, and you
# should NOT use any notes or books.
# According to the Honor Code, you should report any student who appears
# to be violating these rules.
##############################################################################

'''This section tests your knowlegde of OOP with classes in Python.
We have a Track class that stores data about audio tracks (songs) and a
PlayList class that manages a list of Tracks.'''

########################
# Question 5 (15 points)
# Implement the missing sections in the Track class.
########################
class Track(object):
    def __init__(self, title, artist, minutes, seconds):
        '''Initializes PRIVATE fields with the same name as those
           passed in as arguments.
           title and artist will be passed in as strings.
           minutes and seconds will be passed in as integers.
           seconds will be between 0 and 59.
           '''
        # TODO
        self.__title = title
        self.__artist = artist
        self.__minutes = minutes
        self.__seconds = seconds

    def length_as_str(self):
        '''Returns the length of the track as a string.
           Example:
              A track of 4 minutes and 22 seconds should be "4:22"
```

```
            A track of 3 minutes and 9 seconds should be "3:09"
            (note the leading 0 for the seconds portion)
        '''
        # TODO
        result = str(self.__minutes) + ':'
        if self.__seconds < 10:
            result += '0'
        result += str(self.__seconds)
        return result

    def length_in_seconds(self):
        '''Returns the length of the track in seconds.'''
        # TODO
        return self.__minutes * 60 + self.__seconds

    def __str__(self):
        '''Returns a string representation of the track.'''
        return self.__title + ': ' + self.__artist + ' (' + \
               self.length_as_str() + ')'

#########################
# Question 6 (15 points)
# Implement the missing sections in the PlayList class.
#########################
class PlayList(object):
    def __init__(self):
        '''Initializes PRIVATE field track_list to an empty list.'''
        # TODO
        self.__track_list = []

    def add_track(self, track):
        '''Adds a track to the playlist's list of tracks.'''
        # TODO
        self.__track_list += [track]

    def get_longest_track(self):
        '''Returns the longest track in the playlist.
           You may assume that the playlist contains at least 1 track.
           Use method length_in_seconds() in the Track class in your
           algorithm below.'''
        # TODO
        longest_track = self.__track_list[0]
        for track in self.__track_list:
            if track.length_in_seconds() > longest_track.length_in_seconds():
                longest_track = track
        return longest_track

    def __str__(self):
        '''Returns a string representation of the playlist.'''
        s = ''
        for track in self.__track_list:
            s += str(track) + '\n'
        return s
```

```
t0 = Track('The Heart Wants What It Wants', 'Selena Gomez', 4, 35)
t1 = Track('Shake It Off', 'Taylor Swift', 4, 1)
t2 = Track('All About That Bass', 'Meghan Trainor', 3, 10)
t3 = Track('Animals', 'Maroon 5', 3, 49)
t4 = Track('Am I Wrong', 'Nico & Vinz', 5, 4)
t5 = Track('Stay With Me', 'Sam Smith', 3, 29)
t6 = Track('Roar', 'Katy Perry', 4, 29)
t7 = Track('Love Me Harder', 'Ariana Grande & The Weeknd', 4, 11)
t8 = Track('Thinking Out Loud', 'Ed Sheeran', 4, 56)
t9 = Track('We Can\'t Stop', 'Miley Cyrus', 3, 33)

def simple_test():
    playlist = PlayList()
    playlist.add_track(t1)
    playlist.add_track(t5)
    playlist.add_track(t7)
    assert str(playlist) == 'Shake It Off: Taylor Swift (4:01)\n' + \
                            'Stay With Me: Sam Smith (3:29)\n' + \
                            'Love Me Harder: Ariana Grande & The Weeknd (4:11)\n'
    print 'Playlist matches.'
    assert str(playlist.get_longest_track()) == \
                            'Love Me Harder: Ariana Grande & The Weeknd (4:11)'
    print 'Longest track matches.'

if __name__ == '__main__':
    simple_test()
    # Feel free to write more of your own tests here.
```

Rubric: Q5: TODO **Rubric:** Q6: TODO