Quiz 3

To get credit for this quiz, use the Quiz tool at eee.uci.edu to enter your answers, within the Sunday-to-Tuesday quiz period. This quiz is a little longer than usual, to give you more midterm-review opportunities.

Problem 1 (4 points) **Topic: Function calling, flow of control, parameter passing**
Below are four function headers (with parameter types and a docstring comment describing the function's behavior):

```
def function1(L: 'list of numbers') -> None:
    ''' Print, one per line, double the value of each item in the parameter '''

def function2(L: 'list of numbers') -> None:
    ''' Print the parameter in bracketed Python list notation '''

def function3(L: 'list of numbers') -> 'list of numbers':
    ''' Return a list containing each item of the parameter, doubled; leave
        parameter unchanged  '''

def function4(L: 'list of numbers') -> 'list of numbers':
    ''' Double each item in parameter, changing corresponding argument (side effect);
        also return changed list  '''
```

Each of the following groups of statements could be the function body that goes with exactly one of the above headers. For each of the statement groups below, indicate which of the above function headers (`function1` through `function4`) could be the header that goes with that statement group as its body.

```
#a.
print(L)
return
```

function2

```
#b.
for n in range(len(L)):
    L[n] = L[n] * 2
return L
```

function4. Here we're mutating the list---changing it in place, by assigning a new value to each individual indexed element of the list.

```
#c.
doubled = [ ]
for i in L:
    doubled = doubled + [i * 2]
return doubled
```

function3. This leaves the parameter L unchanged; the new list (doubled) is created with the specified result.

```
#d.
for i in L:
    print(2 * i)
return
```

function1

Problem 2 (10 points) **Topic: Flow of control with if/else and for**

For each of these sequences of statements, what does Python print?

(a)

```
s = 'Duck'
t = 'Duck'
print('Donald')
if s >= t:
   print('Huey')
   print('Dewey')
print('Louie')
```

Donald
Huey
Dewey
Louie

(b)

```
a = 10
b = 20
if a >= b:
   print('Huey')
else:
   print('Dewey')
print('Louie')
```

Dewey
Louie

(c)

```
p = 'Mickey'
q = 'Minnie'
print('Mouse')
if p == q:
   print('Goofy')
print('Pluto')
```

Mouse
Pluto

(d)

```
L = ['Huey', 'Dewey', 'Louie']
print('Nephews')
for duck in L:
   print(duck)
print('Uncle Donald')
```

Nephews
Huey
Dewey
Louie
Uncle Donald

(e)

```
L = [2004, 2008, 2012, 2016]
print('Election years')
for i in range(4):
   print(i, "--", L[i])
print('Remember to vote!')
```

Election years
0 -- 2004
1 -- 2008
2 -- 2012
3 -- 2016
Remember to vote!

Problem 3 (6 points) **Topic: Function definition, string indexing**

Write a function called `abbr` that takes two non-empty strings (i.e., you can assume each string is at least one character long; you don't have to check). It should return a two-character abbreviation constructed from the first character of each of the input strings. For example, `abbr('University', 'California')` returns `UC` . Your definition should follow the design recipe by including: (a) type specifications for the parameters and return value, (b) a docstring comment, and (c) at least two example calls with the expected answers that can be run as tests; you may use `print` or `assert`.

```
def abbr(s1: str, s2: str) -> str:
    ''' Return abbreviation made from first char of each parameter '''
    return s1[0] + s2[0]
print(abbr('University', 'California'), 'should be UC')
assert abbr('Los', 'Angeles') == 'LA'
```

Problem 4 (8 points) **Topic: sorting lists of namedtuples**

Write a function called `least_expensive_dish` that takes a list of restaurants and returns the name of the least expensive dish on the list.

There are two approaches to this; either one is fine. The first approach, as in Lab 3, uses the `sort` method on lists with the `key=` argument and the name of a function like this one:

```
def Restaurant_price(r: Restaurant) -> float:
  ''' Return the value of the price attribute of the parameter
  '''
```

(You don't have to define this function; assume it already exists.)

The second approach just goes through the list with a loop, keeping track of the lowest-priced item it has found. Choose the first approach or the second and write your code, including type annotations and docstring comments. You do not need to supply tests (assertions).

```
def least_expensive_dish(L: list) -> str:
    ''' Return the name of the least expensive dish in the list
    '''
    L.sort(key=Restaurant_price)
    return L[0].dish

def least_expensive_dish2(L: list) -> str:
    ''' Return the name of the least expensive dish in the list
    '''

    lowest_price_so_far = L[0].price
    lowest_dish_so_far  = L[0].dish
    for r in L[1:]:
        if r.price < lowest_price_so_far:
            lowest_price_so_far = r.price
            lowest_dish_so_far  = r.dish
    return lowest_dish_so_far
```

Problem 5 (12 points) **Topic: Processing lists of namedtuples**
Suppose we have a collection of images (as we might on Flickr or with Facebook photos). We could represent each image as a namedtuple, with numbers for the height and width and an additional field for the content of the image itself (whose form we don't need to worry about now).

```
Image = namedtuple('Image', 'height width content')
image1 = Image(250, 150, "w")
image2 = Image(150, 250, "x")
image3 = Image(100, 100, "y")
image4 = Image(1500, 1000, "z")
image_list = [image1, image2, image3, image4]
```

(a) (3 points) A portrait-style image is taller than it is wide. (A landscape-style image is wider than it is tall.) Complete the following function definition according to the header, docstring, and assertions shown.

```
def is_portrait(i: Image) -> bool:
    ''' Return True if image is portrait-style (taller than wide) and False otherwise
    '''
```

return i.height > i.width

It's bad style, even if we might not take off points points on an exam, to write something like:

if i.height > i.width // return True // else: // return False

```
assert is_portrait(image1)
assert not is_portrait(image2)
assert not is_portrait(image3)
```

(b) (6 points) Complete the definition of the function below according to the header and docstring shown, with one identifier name, constant, or operator in each blank space. Where applicable, use functions from elsewhere in this quiz rather than duplicating code.

```
def keep_portraits(L: 'list of Image') -> 'list of Image':
    ''' Extract portrait-style images from parameter, returning them in a list
    '''
    result = [ ]
    for i in _______________:                         L

        if _______________ (_______________):         is_portrait(i)

            _______________.append(_______________)   result.append(i)

    return _______________                            result
```

(c) (3 points) Write two assert statements to test `keep_portraits`. One should use the definitions above; the other should test it with an empty list.

assert keep_portraits(image_list) == [image1, image4]

assert keep_portraits([ ]) == [ ]

Creating a thorough set of test cases is another skill that good programmers have. In particular, you should always test for the case where the main data is empty or zero; generally you want your function to return gracefully in those cases, perhaps the values empty or zero.

Problem 6 (5 points) **Topic: Policies on collaboration and academic honesty**

Which of the following are accurate statements about doing your lab work in ICS 31? We ask this because there are cases every quarter where students have not followed the policies on appropriate collaboration and pair programming, often leading to unhappy consequences. If the statement is accurate, just answer "Yes." If not, explain what's wrong.

A. The work you submit for a lab assignment should be the joint effort of two students who have officially designated each other as partners using the Partner App [except in specific cases with the explicit permission of the students' TA].
YES

B. Pair programming means that it's okay to work with any student in the class, so long as no more than two students are working together at any one time. **NO; once you have designated a partner for an assignment, that is the only person you may work with on that assignment (besides tutors and TAs).**

C. You may work on the assignment outside of lab hours with your designated partner.
YES

D. If your designated partner is not available, you may work with your roommate or other classmate outside of lab hours. **NO. "Work with" means writing solutions to lab problems. You MAY discuss with anyone (a) what a problem means, (b) the operation of specific Python features, (c) error message meanings.**

E. It is never permissible to copy code from the lab solution of a student who isn't your partner. "Copy" means writing code down by hand, receiving code electronically, or even having someone read the code to you line by line. **YES, it's never permissible to share code outside your partnership. General high-level discussions about an overall approach are okay, but code-level solutions must be kept private.**

F. If you can't complete your lab assignment by the deadline, turn in what you have completed as of the deadline. Then you may ask your TA about taking more time to work longer on the assignment, resubmitting a more complete solution for more credit (it's the TA's decision whether to allow that or not on a given assignment). It's always a good idea to try to complete the assignment eventually, even without more credit, because you learn more if you work more problems.
YES

G. If you don't complete one part of a lab assignment, it might lower your score a little on that assignment (or it might not, depending on how many parts there are). One slightly-lower score on one lab assignment isn't very likely to affect your grade in the course. But if you are found to have violated the collaboration rules, you will get a zero on the assignment, you may have your course grade lowered (possibly all the way to F), and a record of the incident will be kept at Student Affairs.
YES

H. It's helpful to mail your code for a lab problem to another student in the class who isn't your partner, so he or she can get a general idea of how you did it. **NO! The other student may panic (or just mess up) and submit your code as theirs; then you could be in hot water for violating honesty policies.**

I. It's helpful to mail your code for a lab problem to your designated partner after each lab session, incase someone isn't available at the next scheduled lab time.
YES

J. It's okay to share your solution to a lab problem with other students after the lab is due. **YES, but it's risky: If the person you share it with has arranged for a deadline extension, that person still might steal and submit your code. This has happened in the past. It's best to wait a few days, at least, before sharing. It is NOT okay to post your solutions to ICS 31/32/33 lab assignments on public sites like GitHub or CourseHero. Other people will find it and submit it as their work, even in a later quarter. The Student Conduct office regards the posting as a form of dishonesty. And posting your first-year work won't help you professionally.**
—> —> —> For the ICS 31 guidelines on collaboration, see http://www.ics.uci.edu/~kay/courses/31/col-lab.html and http://www.ics.uci.edu/~kay/courses/31/hw/