Quiz 5

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.

Problem 1 (8 points) **Topic: Processing lists of namedtuples containing lists.**

Fill in the blanks in the function definition below to make it consistent with its header and docstring. Each blank should contain just one identifier, constant, or operator. [Recall that an identifier is just a name: a variable name, a parameter name, a field/attribute name, a function name, a method name.]

```
Dish = namedtuple('Dish', 'name price calories')
Restaurant = namedtuple('Restaurant', 'name cuisine phone menu')

def Restaurant_average_calories (R: Restaurant) -> float:
    ''' Return the average number of calories on the restaurant's menu.
        The menu is a list of Dish structures.
    '''
    return Menu_average_calories(_______________._______________)
```

return Menu_average_calories(R.menu)

```
def Menu_average_calories(M: 'list of Dish') -> float:
    ''' Return the average number of calories on the menu (a list of Dishes)
    '''
    if len(M) == 0:
        return 0
    else:
        sum = _______________                          sum = 0
        for d in _______________ :                     for d in M:
            _______________ += _______________.calories      sum += d.calories
        return _______________ / _______________(M)    return sum / len(M)
```

— If this was hard for you, it may be that you're not yet familiar enough with Python statements and how they're constructed.

— Take the first two blanks: We know that the Restaurant_average_calories function takes a Restaurant (which has four fields). We also know that Menu_average_calories takes a list, so that those first two blanks, the arguments in the call to Menu_average_calories, must specify a list. Where can we get a list, given the data we have inside Restaurant_average_calories? The only data there is R, the parameter, which is a Restaurant. The only list available is the menu field/attribute of R. So how do we get one field out of a namedtuple? With dot notation. R.menu (that's the answer) gives us the menu field of R, a list of dishes, which is the appropriate type of data to call Menu_average_calories with.

— If any terms or concepts in the preceding paragraph are unclear to you, now's the time, before the next midterm, to learn or review what they mean.

— It may also mean that you're not comfortable enough with the parts of Lab 5 that deal with dishes and collections. Processing lists, and processing lists of objects that themselves include lists, is an important aspect of the course.

Problem 2 (16 points) **Topic: Lists of namedtuples, namedtuples containing lists**

Suppose we have a list of Student objects similar to those we've seen before:

```
Student = namedtuple('Student', 'ID name level major studylist')
# All are strings except studylist, which is a list of Courses.
# An example showing the form of the data:
s1 = Student('11223344', 'Anteater, Peter', 'FR', 'PSB', [ics31, wr39a, bio97, mgt1])
```

Each Student object contains a list of Course objects defined as follows:

```
Course = namedtuple('Course', 'dept num title instr units')
# All are strings except number of units
# An example showing the form of the data:
ics31 = Course('ICS', '31', 'Intro to Programming', 'Kay', 4)
```

(a) (6 points) Complete the definition of the function below according to the header and docstring shown.

```
# Note:  The annotation [Student] below means the same thing as 'list of Student'
def class_level_count(SB: [Student], class_level: str) -> int:
    '''  Return the number of students in the list SB whose class level matches the
         specified value.    '''
  result = 0
  for s in SB:
    if s.level == class_level:
      result += 1
  return result
```

(b) (10 points) Complete the definition of the function below according to the header and docstring shown. You may define a second function if it helps you organize your solution.

```
def enrollments_for_instructor(SB: [Student], instructor_name: str) -> int:
    '''  Return the total number of students enrolled in courses taught by named
         instructor.  (If a student is enrolled in two courses taught by the same
         instructor, that student counts twice.)
    '''
def enrollments_for_instructor(SB: [Student], instructor_name: str) -> int:
  ''' Return the number of students enrolled in courses taught by named instructor
  '''
  result = 0
  for s in SB:
    result = result + enrollments_on_studylist(s.studylist, instructor_name)
  return result

def enrollments_on_studylist(courses: [Course], instructor_name: str) -> int:
  ''' Return the number of enrollments by this student in courses taught by named instructor
  '''
  result = 0  # Of course this is a separate (local) variable from the one in the other function.
  for c in courses:
    if c.instr == instructor_name:
      result += 1
  return result
[ A solution using nested loops is also possible. ]
```

Problem 3 (6 points) **Topic: Distinguishing data types; interpreting error messages**

Below are two code segments; each one generates an execution error whose message is shown. Fix the code as simply as possible to remove the error and produce the intended result.

(a) (3 points)

```
L = ['Huey', 'Dewey', 'Louie', 'Donald', 'Daisy']
for i in range(10):
    print(L[i])

Traceback (most recent call last):
  File "/ICS/31/Quizzes/Quiz Code/quiz5.py", line 3, in <module>
    print(L[i])
IndexError: list index out of range
```

The index (or subscript or position) number, i, goes from 0 to 9 according to the for-loop. But when it hits 5, it runs off the end of the five-element list. The best correction is

```
for i in range(len(L)):
```

so the loop just ranges through the actual size of L. This approach would also work:

```
for duck in L:
  print(duck)
```

(b) (3 points)

```
Restaurant = namedtuple('Restaurant', 'name cuisine phone dish price')
RESTAURANTS = list of Restaurant objects
def Restaurants_serving_cuisines (RL: [Restaurant], cuisines: [str]) -> [Restaurant]:
    ''' Return a list of Restaurants serving any of the cuisines specified.
    '''
    result = [ ]
    for r in RL:
        if r.cuisine in cuisines:
            result.append(r)
    return r

print("Names of Southeast Asian Restaurants:")
SEAsian_restaurants = Restaurants_serving_cuisines(RESTAURANTS,
                      ['Thai', 'Vietnamese', 'Laotian', 'Cambodian'])
for each_rest in SEAsian_restaurants:
    print(each_rest.name)

Traceback (most recent call last):
  File "/ICS/31/Quizzes/Quiz Code/quiz5.py", line 16, in <module>
    print(each_rest.name)
AttributeError: 'str' object has no attribute 'name'
```

The short answer here is that Restaurants_serving_cuisines is supposed to return a list (of Restaurants) but instead it returns just a single Restaurant. When we try to use that value as if it were a list of Restaurants, we get the error.
The correction is to have the function return result instead of r.

Here's the complete chain of reasoning:

A message like "'str' objet has no attribute 'name' " says that we're trying to apply name (which should give us the restaurant's name from a Restaurant) to something for which name doesn't make sense---that is, something that isn't a Restaurant.
So where in the code do we apply name? To each_rest in the print statement. We EXPECT each_rest to be a Restaurant, but it must not be. Where did we go wrong?
In the bottom for-loop, each_rest takes on each item in SEAsian_restaurants, which is the value that was returned by Restaurants_serving_cuisines. We EXPECT that to be a list of Restaurants, but it must not be---at least one thing in SEAsian_restaurants (and thus in what was returned by Restaurants_serving_cuisines) must NOT be a Restaurant.

So we'd better look at what Restaurants_serving_cuisines actually returns.
The Restaurants_serving_cuisines function is supposed to return a list of Restaurants.
But instead it returns r, a single Restaurant. The bottom for-loop in the calling program iterates through the fields of that one Restaurant; the fields in question are strings; it makes no sense to take the name field of a single string, which is what gave us the error.

The function returns r, but it should return result. In fact, just looking at the function definition might tell us that, if we're familiar with the pattern of building up a result by scanning through a list:
We set the result to empty, we go through the list appending what we want to append, and then at the end we return the result.
But here, we got that wrong and returned the control variable of the for loop instead.
There's the mistake.