Full Name: _____

Email Address (UW Net ID): _____@uw.edu

Section: _____

CSE 160 Summer 2024 - Final Exam

Instructions:

- You have **55 minutes** to complete this exam.
- The exam is **closed book**, including no calculators, computers, phones, watches or other electronics.
- You are allowed a single sheet of notes for yourself.
- We also provide a syntax reference sheet.
- Turn in *all sheets* of this exam, together and in the same order when you are finished.
- A couple of extra pages have been provided to you at the end of the exam. You may use these as scratch paper.
- When time has been called, you must put down your pencil and stop writing.
    - **Points will be deducted if you are still writing after time has been called.**
- You may only use parts and features of Python that have been covered in class up to this point.
- You may ask questions by raising your hand, and a TA will come over to you.

Good luck!

Question	**Topic**
Question 1	Style and Errors
Question 2	Testing
Question 3	Errors
Question 4	Classes and Sorting
Question 5	File I/O

Question 1) The following excerpt of code contains numerous issues. For each line number, indicate whether there is a *runtime error* (the program cannot execute) or a *code quality mistake* (style errors). Additionally, please name or describe the error/mistake, and also describe one additional code quality mistake besides the line numbers mentioned.

```
1 def temperature_averaging(temperatures)
2    """
3    Takes in a list of tuples in the format (city, temperature)
4    in which city is a string and temperature is a list of
5    temperatures associated with that city. Then, returns a
6    dictionary mapping a city to its average temperature.
6
7    Example: temperatures = [(“Paris”, [78, 92, 24]), (“Tokyo”, [34,
9    12])]
10
11   Should return: {“Paris” : 64.6666667, “Tokyo”: 23}
12   """
13   D = {}
14   for tup in range(len(temperatures)):
15       sum = 0
16       count = 0
17       for temp in tup[1]:
18           sum += temp
19           count += 1
20       avg = sum / count
21       sum = 0
22       D[city] = avg
23   return D
24   pass
```

A. Line 13

Runtime error or code quality mistake? (circle one)

Explanation:

Code quality mistake. Variable names should be all lowercase and descriptive.

B. Line 17

Runtime error or code quality mistake? (circle one)

Explanation:
Runtime error. Tup is an integer, so it cannot be indexed into.

C. Line 22

Runtime error or code quality mistake? (circle one)

Explanation:
Runtime error. `city` is not defined.

D. What is one other code quality mistake not mentioned above?
setting sum = 0 twice in the for loop, manually counting length rather than calling len(), did not remove pass at the end of the function. Also ok: using range(len()) rather than directly looping over the tuples in the list.

Question 2) Suppose you need to implement the following function:

```
def mean_of_points(list_of_points):
    """
    Calculate the mean of a given group of data points. You should NOT
    hard-code the dimensionality of the data points).

    Arguments:
        list_of_points: a list of lists representing a group of data points

    Returns: a list of floats as the mean of the given data points.

    Example:
        Code:
            list_of_points = [[1.1, 1, 1, 0.5], [4, 3.14, 2, 1], [0, 0, 0, 0]]
            print(mean_of_points(list_of_points))
        Output:
            [1.7, 1.38, 1.0, 0.5]
    """
```

1. How would you write a test in order to test that the given example in the docstring gives the correct output? Note that you should not check the equality of lists using ==.

```
received = mean_of_points(list_of_points)
expected = [1.7, 1.38, 1.0, 0.5]

for i in range(0, len(expected)):
        assert (math.isclose(received[i], expected[i]))
```

2. What output would you get in the terminal if the test failed?

AssertionError as well as a traceback of the lines of code that lead to the error

3. What output would you get in the terminal if the test passed?

No output

Question 3) For each of the following errors, write code that would cause this error in two lines or less.

Example: AssertionError

```
assert 1 == 0
```

A. KeyError

Write code here:

```
{1: 1}[2]
```

B. TypeError

Write code here:

```
1 + “1”
```

C. SyntaxError

Write code here:

```
“a”a
```

D. NameError

Write code here:

```
var += 1
```

Question 4)

In lecture, we created the class below to show the Olympic countries:

```
class Country:

        def __init__(self, country):

         self.country = country

          self.athletes = dict()

          self.medals = dict()


       def add_athlete(self, athlete, sport):

              self.athletes[athlete] = sport


       def count_medals(self, level, count):

              if level not in self.medals:

                     self.medals[level] = 0

              self.medals[level] += count
```

We need to add a couple of new methods to analyze the data. For each method include the def line and any necessary parameters.

1. Before we write the methods, write one line of code to create an instance of this class using whatever country of your choosing.

```
usa = Country('United States')
```

2. Create a method `average_medal_count()` that will average the number of medals that are already stored at each of the levels (Gold, Silver, and Bronze). This method should return a float.

```
def average_medal_count(self):

       total = 0

       for i in self.medals:

              total += self.medals[i]

       return total / len(self.medals)
```

3. We need to announce all of our athletes! At the opening ceremony, they are going to be announced within their sport (ie. all the marathoners are going to come together, all the swimmers will come together, etc).These sports are going to come in alphabetical order. The athletes in each sport also need to be sorted alphabetically.

   Write a method `our_athletes()` that returns a list of the athletes sorted within the sport. If an instance of your class has the athletes dictionary: `{"Megan Keith": "Athletics", "Carl Hester": "Equestrian", "Tom Daley": "Diving", "Jade O'Dowda": "Athletics"}` your method should return `['Jade O'Dowda', 'Megan Keith', 'Tom Daley', 'Carl Hester']`. After writing your method, include what a call to the instance would look like.

```
def our_athletes(self):

    both_items = list(self.athletes.items())

    sorted_by_athlete = sorted(both_items, key=itemgetter(0))

    sorted_by_sport = sorted(sorted_by_athlete, key=itemgetter(1))

    final_lst = [couple[0] for couple in sorted_by_sport]

    return final_lst
```

call: usa.our_atheletes()

4. The exact number for athletes on an Olympics team is constantly changing with athletes falling ill, getting injured before the Olympics begin, and other athletes being added. Write a short method `print_numbers()` that will print the number of athletes and the number of different sports that are being participated in by your instance to allow us to get the numbers quickly. This print statement should be in the format: `"[country] has ___ athletes participating in ____ different sports"` Include what a call to this method would look like.

```
def print_numbers(self):

    len_athletes = len(self.atheletes)

    sports = set()

    for competitor, sport in self.athletes.items():

        sports.add(sport)

    print(self.country, "has", len_athletes, 'participating in',

          len(sports), 'different sports')
```

Call: usa.print_numbers()

Question 5:

Given a `.csv` file that has the following columns headers:

```
name,homework,exams,participation
```

Define a function called `grade_calculation` which takes in a filename and returns a list of tuples containing the name and the overall score out of 1.0, which is calculated by the following weights: `homework * 0.6 + exams * 0.3 + participation * 0.1`. You may assume that the column headers in the file will always be the same, although the number of rows may vary. For example:

```
name,homework,exams,participation
Dubs,0.8,0.8,0.8
Butch,0.65,0.8,0.5
Duck,0.5,0.1,0.9
```

Would return:

```
[("Dubs", 0.8), ("Butch", 0.68), ("Duck", 0.42)]
```

```
import csv

def grade_calculation(filename):
    scores = []
    with open(filename) as f:
        reader = csv.DictReader(f)
        for row in reader:
            overall = float(row["homework"]) * 0.6 + float(row["exams"]) * \
                0.3 + float(row["participation"]) * 0.1
            scores.append((row["name"], overall))
    return scores
```

Extra Credit)
Draw what you will do with Python after this class