CSC 242 Section 504 Winter 2017
Homework Assignment 2

Due: As specified on D2L

The purpose of this homework assignment is to gain experience in writing your own Python module. A Python module is written in its own .py file. The module may then be loaded from the Idle prompt, or as part of the code in a different module. The module you will be completing is called hw2, because it is in a file called hw2.py.

We will use the "import" statement to load this module into its own namespace. Notice that in the accompanying hw2_test.py file, there is an import statement at the top. This means that if you press F5 with the Python editor open to hw2_test.py, the code you write in hw2.py will automatically load at the same time.. Therefore, there is no need to press F5 while the editor is open to hw2.py (unless you want to run tests of your own in addition to those in hw2_test.py).

Notice that the hw2 module imports the built-in Python math module. You can explore the contents of the math module by using the dir function, as in

```
>>> dir(math)
```

The homework assignment is worth 4% of your overall grade, and will therefore be graded on a scale of 0 to 4. Note that there are 5 functions for you to complete. Problems 1 and 2 are worth .5 points each, and problems 3-5 are worth 1 point each. On problems 3-5, you may receive partial credit of .5 points if you function has no syntax errors and works correctly on at least 1/2 of the test calls in hw2_test.py.

I have placed a file named `hw2.py` in the D2L Homework 2 dropbox folder. You may use this is a starting point for your program. **You must write the rest of your code by yourself. Do not copy a solution, or any part of it from any source, including (but not limited to) other students or the Internet... doing so would be plagiarism.**

Please upload your completed hw2.py file containing your solution to this assignment to D2L. Include a comment at the top with your name. Your submission must be uploaded by the date and time as specified on D2L to be considered on time. Please see the course syllabus for my late submission policy.

1. (.5 points) Write a function called circle_area, which is passed the radius r of a circle and returns the circle's area. Remember that the area of a circle of radius r is πr^2. The value of π is in the math module.

   Here are some examples:

   ```
   >>> circle_area(1)
   3.141592653589793      # π
   >>> circle_area(2.5)
   ```

```
19.634954084936208      # π * 2.5²
```

2. (.5 points) Write a function called circle_circumference. It is passed the radius *r* of a circle and returns the circle's circumference. Recall that the circumference of a circle with radius *r* is $2\pi r$. For example:

```
>>> circle_circumference(1)
6.283185307179586                    # 2 * π
>>> circle_circumference(2.5)
15.707963267948966                   # 5 * π
```

3. (1 point) Write a function called line_length. It is passed 2 parameters, which are both tuples of length 2. A Python tuple is an immutable list. The members of a tuple are surrounded by parentheses rather than [ ]. In this case, the tuples represent points in 2-dimensional space. For example:

```
>>> line_length((1, 1), (4, 5))
5.0
```

because 5 is the length of a line segment connecting the points (1, 1) and (4, 5). Recall that in general, the length of a line segment connecting the points (x_1 , y_1) and (x_2 , y_2) is $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.

Here are some other examples:

```
>>> line_length((0,1), (1,0))
1.4142135623730951               # square root of 2
>>> line_length((3,-1), (4,2))
3.1622776601683795               # square root of 10
```

4. (1 point) Write a function called triangle_perimeter.. It is passed 3 parameters, all of which are tuples of length 2. It returns the perimeter of the triangle defined by the 3 points represented by the 2-tuples. For example:

```
>>> triangle_perimeter((1, 1), (1, 2), (2, 1))
# 1 + √2 + 1
3.414213562373095
>>> triangle_perimeter((0,0), (3, 4), (3,0))
# 5 + 4 + 3
12.0
```

5. (1 point) Write a function called truncate. It is passed 2 parameters, a float *f* and an integer *d*. The function truncates *f* so that it has *d* digits to the right of the decimal point. For example:

```
>>> truncate(line_length((0,1), (1,0)), 3)
1.414
>>> truncate(circle_circumference(1), 1)
6.2
>>> truncate(circle_area(1), 4)
3.1415
```

You may assume that the second parameter *d* will be a positive integer.