Mod 1 Homework: Basic Python

In this assignment, we will make our own versions of some of the attributes (functions and variables) provided by the `math` module, then compare the two.

Part 1 - `CustomMath.py`

Let's make our own math module to import in `hw1.py`. We wil name this file `CustomMath.py` - you need to create it yourself.

Do not use the built-in `math` module here - create everything from scratch.

Attributes

Create a file named `CustomMath.py`. Then add the following attributes:

Variables

- `pi` - a variable containing the value 3.14
- `e` - a variable containing the value 2.72

Functions

- `sqrt(num)`
  - If `num` is an `int` or `float`, returns the square root of `num`
  - Otherwise, returns the string "*{} is not a number*". The curly braces denote that the value of `num` should be inserted in the string, e.g. with the `str.format()` command.
- `floor(num)`
  - If `num` is a `float`, returns the highest integer less than `num`
  - If `num` is an `int`, returns the string "*{} is the floor*"
  - Otherwise, returns the string "*{} is not a number*". The curly braces denote that the value of `num` should be inserted in the string, e.g. with the `str.format()` command.
- `ceil(num)`
  - If `num` is a `float`, returns the lowest integer greater than `num`
  - If `num` is an `int`, returns the string "*num {} is the ceil*"
  - Otherwise, returns the string "*{} is not a number*". The curly braces denote that the value of `num` should be inserted in the string, e.g. with the `str.format()` command.

Special Behavior

- Use an `if __name__ == '__main__'` block to differentiate between direct execution and import
  - when executed directly, print the string *CustomMath is being executed directly*

```
$ python3 CustomMath.py
CustomMath is being executed directly
```

- when imported, print the string *CustomMath has been imported*

```
>>> import CustomMath
CustomMath has been imported
>>>
```

Examples

Any examples below are intended to be illustrative, not exhaustive. Your code may have bugs even if it behaves as below. Write your own tests, and think carefully about edge cases.

Import and variables

```
>>> from CustomMath import *
CustomMath has been imported
>>> pi
3.14
>>> e
2.72
```

sqrt

```
>>> sqrt(4)
2.0
>>> sqrt("hello")
'hello is not a number'
```

floor

```
>>> floor(1.2)
1.0
>>> floor(-1.2)
-2.0
>>> floor(1)
'1 is the floor'
>>> floor([1,2,3])
'[1, 2, 3] is not a number'
```

ceil

```
>>> ceil(1.2)
2.0
```

```
>>> ceil(-1.2)
-1.0
>>> ceil(1)
'1 is the ceil'
>>> ceil("hello")
'hello is not a number'
```

Part 2 - `hw1.py`

We will compare the output of `CustomMath.py` to python's built in math module in `hw1.py` (you need to create this file yourself)

Import `math` and `CustomMath.py` aliased as `mth` and `cmth`, respectively, in `hw1.py`.

Write a function `f_dif` that returns the difference of two other functions. `f_dif` should take two functions as (required) parameters and an optional third parameter which specifies the argument to call those functions with. If no third parameter is specified, use a value of 0.0.

- `f_dif()`
  - two required parameters - functions
  - one optional parameter - an argument to be passed to the functions. If this parameter is not specified, it should default to a value of 0.0.
  - returns the difference of the two input functions called with the argument specified

Examples

Any examples below are intended to be illustrative, not exhaustive. Your code may have bugs even if it behaves as below. Write your own tests, and think carefully about edge cases.

```
>>> from hw1 import *
CustomMath has been imported
>>> mth.pi
3.141592653589793
>>> cmth.pi
3.14
>>> mth.floor(mth.pi)
3
>>> cmth.floor(cmth.pi)
3.0
>>> mth.floor(cmth.pi)
3
>>> f_dif(mth.floor, cmth.floor)
0.0
>>> f_dif(mth.ceil, mth.floor, 3.14)
1
```

Submitting

At a minimum, submit the following files:

- `CustomMath.py`
- `hw1.py`

Students must submit to Mimir individually by the due date (typically, the Wednesday 10 days after this module opens at 11:59 pm EST) to receive credit.

Grading

70 - `CustomMath.py`

- 10 - variables
- 10 - sqrt
- 10 - floor
- 10 - ceil
- 15 - print statement when executed
- 15 - print statement when imported

30 - `hw1.py`

- 30 - all behavior

Feedback

If you have any feedback on this assignment, please leave it here.

We check this feedback regularly. It has resulted in:

- A simplified, clear **Submitting** section on all assignments
- A simplified, clear **Grading** section on all assignments
- Clearer instructions on several assignments (particularly in the recursion module)