Work with your neighbor. (This will be graded for participation only.)

1. Below is a portion of the `Point` class definition:

```
import math
class Point:
    def __init__(self, x, y):
        self._x = x
        self._y = y

    def distance_from_origin(self):
        return math.sqrt(self._x**2 + self._y**2)
```

Write an `__eq__()` method for the `Point` class.

Usage:

```
>>> p1 = Point(8,12)
>>> p2 = Point(3,5)
>>> p1 == p2
False
```

```
    def __eq__(self, other):
```

2. Write a method `as_tuple(self)` that returns the x and y coordinates of a `Point` as a tuple.

Usage:

```
>>> p = Point(7,12)
>>> p.as_tuple()
(7, 12)
```

3. Write a `__str__()` method for `Point`.

   Usage:

```
>>> p = Point(7,12)
>>> str(p)
'Point(7,12)'
>>> print(p)
Point(7,12)
```

4. In this problem, you will define a `Bookdata` class and then perform a simple computation on `Bookdata` objects after they have been created. You can use the starter code on the class website in the link next to the ICA: **`books_start.py`**

   Define a class called `BookData` that has the following attributes: a title, an author, and a rating, which is of type `int`. In addition, implement the following methods:

`get_title()`	returns the title of a `book`
`get_author()`	returns the author of a `book`
`get_rating()`	returns the rating of a `book`
`__str__()`	returns a string representation of a `BookData` object in the format `BookData -- title: author : rating`

```
class BookData:
    # your code goes here
```

The following program continuously prompts the user for a book title, author, and rating; it then creates a `BookData` object with that data and saves it in a list. When the user types "no" the program stops prompting for information.

Finish the program by writing the code to perform the actions specified in the comments below:

```
def main():
    book_list = []
    answer = 'yes'
    while prompt != 'no':
        title = input("Book: " )
        author = input("Author: ")
        rating = int(input("Rating: "))
        b = BookData(author, title, rating)
        book_list.append(b)
        answer = input('Enter another book? Answer yes or no: ')

    # Write the code to print out the BookData objects, one per line,
    # and compute the average rating of all the books that were added
    # to the list. Print the average rating.

    # Your code goes here:




main()
```

5. **(Challenge.)** Define a class called `ClockTime` that keeps track of information about time as represented in a clock. Times are measure on the 12 hour clock scale where 11:59 PM is following by 12:00AM. The class should have the following methods: **(Fixed the typos here.)**

   a) `__init__()`

      Takes three arguments: `hours`, `minutes`, and `isAM` and sets three instance variables accordingly. Note that `isAM` is a Boolean value.

   b) `__str__()`

      for AM, returns the time in the format `hours:minutes AM`
      for PM, returns the time in the format `hours:minutes PM`

   c) `total_minutes()`

      returns the total number of minutes. If 4:12 PM were the time, it would return the following: 60 * 4 + 12 = 252

   d) `tick()`

      advances the time by one minute

An example of creating and using a `ClockTime` object follows:

```
>>> t = ClockTime(11, 58, False)  <-fixed the time here
>>> print(t)
11:58 PM
>>> t.tick()
>>> print(t)
11:59 PM
>>>
>>> t.tick()
>>> print(t)
12:00 AM
>>>
```