CSCI 150 HW: class design practice

Due: Wednesday, November 7

To receive full credit, for each exercise you should do the following:

1. **Design**: First, design a Python class as requested in the exercise. Type in your class definition.
2. **Check**: Run the provided test code. Does your actual output agree with the given correct output?
3. **Evaluate**: If the actual output does not match the expected output, keep experimenting, consult the textbook or Python documentation, ask a friend or TA or professor, *etc.* until you can fix your class definition and explain what your misunderstanding(s) were. (You do not need to do anything for step 3 if the outputs already agree exactly.)

You should consider the code in each exercise separately from the other exercises.

1. Write a Python class `BouncyBall`, which represents a bouncy ball containing a certain amount of air.

   - When a `BouncyBall` object is first created, it should have 10 units of air.
   - There should be a method `bounce()` which normally prints the word `Bounce!` and decreasese the amount of air in the ball by two units. However, if the amount of air is less than or equal to three, then `bounce()` does not decrease the amount of air and prints `Thupp.` instead of `Bounce!`.
   - There should be a method `inflate()` which increases the amount of air by three units. If the amount of air ever becomes greater than 12, then the ball explodes by printing `BANG!!!`.
   - You cannot bounce or inflate an exploded ball. After a ball explodes, calling `bounce()` or `inflate()` should just cause a message to be printed such as `Sorry, you cannot bounce this ball! It has exploded.`

   To test your class, you can type in and run the following code:

```
def main():
    b = BouncyBall()

    for i in range(6):
        b.bounce()

    b.inflate()
    b.bounce()
    b.bounce()
```

```
    for i in range(5):
        b.inflate()

    b.bounce()

main()
```

If your definition of `BouncyBall` is correct, `main()` should produce the following output:

```
Bounce!
Bounce!
Bounce!
Bounce!
Thupp.
Thupp.
Bounce!
Thupp.
BANG!!!
Sorry, you cannot inflate this ball!  It has exploded.
Sorry, you cannot bounce this ball!  It has exploded.
```

2. Write a Python class `Gradebook` which works as follows:

   - When a new `Gradebook` object is first created, it should start out with an empty list of grades, and zero points of extra credit.
   - There should be a method `add_grade(g: int)` which adds the grade `g` to the end of the list.
   - There should be a method `add_ec(ec: int)` which adds `ec` points of extra credit to the current amount of extra credit.
   - There should be a method `average()` which computes and returns the average of all the grades so far (the sum of all the grades, plus the extra credit score, divided by the number of grades).

   You can test your implementation of `Gradebook` by running the code below:

```
def main():
gb = Gradebook()
gb.add_grade(90)
gb.add_grade(83)
gb.add_grade(97)
gb.add_ec(10)

print(gb.average())
```

   If your definition of `Gradebook` is correct, this should print `93.33333333333333`.