Flights and Passengers

In this problem, you will be modeling passengers flying on an airplane.

Every time an airplane flies, that airplane has a particular flight number, and some number of passengers. Each passenger has a particular type of ticket - for example, 'coach', 'economy', or 'business'. The passengers also might carry aboard some suitcases, and each suitcase will have its weight recorded.

I've provided a class called `Passenger`, **so you don't need to write that class.** The documentation for `Passenger` is as follows:

```
class Passenger
 |  A simple Passenger class
 |
 |  Methods defined here:
 |
 |  __init__(self, name, suitcase_weights, ticket_type)
 |      name - name of the passenger (string)
 |      suitcase_weights - a list of floats representing the weights of the
 |                         Passenger's suitcases (list)
 |      ticket_type - a string representing the ticket type for this
Passenger
 |                     (string)
 |
 |  __str__(self)
 |      return a string representing this Passenger in the format:
 |      [name], flying [ticket_type] ([comma-separated suitcase weights])
 |      Example:  Farah, flying coach (1.5,0.8,3.2)
```

The list of suitcase weights can be accessed by the instance variable `suitcase_weights`. For example, if I have a `Passenger p`, I can get the list of that `Passenger's` suitcase weights with `p.suitcase_weights`.

You will write a new class called `Flight` in the file **flight.py**. `Flight` should include an initializer (constructor) that takes an integer parameter representing the flight number, and a list of `Passenger`s as an optional argument. If the argument is omitted, then an empty list is assumed.

Write a method `add_passenger` that takes a `Passenger` parameter, and adds that `Passenger` to the `Flight`.

Define the `__str__` method for `Flight` so that it returns a string containing the flight number and the `Passenger` information. For example, a `Flight` with flight number equal to 2, and two `Passengers` named Steve and Marta, might have a `__str__` that returns:

```
Flight #2
-----------------------------------------
```

```
Steve, flying coach (2.2,3.1,3.0)
Marta, flying economy (1.1,2.6,1.0)
```

If there are no `Passenger`s on the `Flight`, the output should say "No passengers", like this:

```
Flight #2
-------------------------------------------
No passengers
```

Finally, you will need to write a method `heaviest_passenger` that returns the `Passenger` with the heaviest luggage sum, and a method `lightest_passenger`, which returns the `Passenger` with the lightest luggage sum.

Raise exceptions in your code where appropriate. Feel free to define your own exception classes.

Your code will be evaluated based on correctness, style, design, and documentation.