Homework #3 - Make a Animal Fact Generator

For this assignment, you will be writing a *Animal_Fact_Generator* class with the following:

- **A constructor (__init__) method**: The constructor will initialize a new *Animal_Fact_Generator object* from the passed list of all possible facts about animals and the respective animals.
    - Set **fact_list** to the passed list of all possible facts
    - Set **animal_list** to the passed list of animals
    - Set **fact_history_list** to an empty list. This will hold the indices of all of the facts that have been generated.

- ***__str__*** **method**: It should return a string with all of the animals in **animal_list** separated by commas, For example : "elephant, cat, wolf, giraffe, panda, tiger"

```
Testing __str__ method
elephant, cat, wolf, giraffe, panda, tiger
```

- ***random_fact*** **method**: Returns a random fact and its animal from the **fact_list** and the **animal_list** respectively**.** It randomly picks an index from 0 to the number of possible facts minus one *(hint: use the random module)*. It adds the index for the fact to the end of the **fact_history_list**. It returns a string containing the fact and the animal at that index (not the index itself) in the following format:

```
fact : This animal can go for more than a week without eating. - wolf
```

- ***get_fact_for_animal*** method : It takes the name of the animal as input and if there is a fact for the animal, it returns the fact or else it returns "Sorry! I do not have any facts for {name_of_animal}"

- ***print_history*** **method**: Prints the content of the **fact_history_list** with the index number in [] and each fact and animal on a separate line. It does not return anything.

```
[1] This animal's tail contains nearly 10 percent of all the bones in its body.
- cat
[1] This animal's tail contains nearly 10 percent of all the bones in its body.
- cat
[4] This animal eats half the day. - panda
[5] These animals are the only big cats that can't roar. - tiger
[1] This animal's tail contains nearly 10 percent of all the bones in its body.
- cat
```

Example Output From HW3.py

```
Testing the first animal_fact_generator:

Generating a random fact

fact : This animal eats half the day. - panda
[
Testing __str__ method
elephant, cat, wolf, giraffe, panda, tiger

Testing that it can get fact for given animal : panda
This animal eats half the day.

Printing the full history:
[4] This animal eats half the day. - panda

==========================================================================================

Testing the second animal_fact_generator:

Testing when no facts have been generated yet
None

Getting a cat fact

fact : This animal's tail contains nearly 10 percent of all the bones in its body.

Getting a dog fact

fact : Sorry! I do not have any facts for dog

Generating five facts randomly
fact : These animals are the only animals that can't jump. - elephant
fact : This animal's heart weighs about 25 pounds. - giraffe
fact : This animal can go for more than a week without eating. - wolf
fact : This animal's tail contains nearly 10 percent of all the bones in its body. - cat
fact : This animal can go for more than a week without eating. - wolf

Printing the full history:
[0] These animals are the only animals that can't jump. - elephant
[3] This animal's heart weighs about 25 pounds. - giraffe
[2] This animal can go for more than a week without eating. - wolf
[1] This animal's tail contains nearly 10 percent of all the bones in its body. - cat
[2] This animal can go for more than a week without eating. - wolf

Testing generate_n_facts method with 200 facts
longest run was length of 5 for index 3
```

NOTE: Your output will not look *exactly* like this because we are using *random* and can't predict what it will return.

NOTE 2: You are welcome to replace the facts and names we have provided in the *main function* with your favorite facts and animals

Grading Rubric - Total of 60 points

5 points - the *__init__* method sets the object's fact_list and animal_list correctly (the instance variables)
5 points - the *__init__* method sets the object's fact_history_list to an empty list
10 points - the *fact* method correctly picks a random index between 0 and the number of facts in the fact_list minus one
10 points - the *__str__* method returns the string with all animals in animal_list separated by commas : "elephant, cat, wolf, giraffe, panda, tiger"
5 points - the *random_fact* method saves the picked index at the end of the fact_history_list
5 points - the *random_fact* method returns the fact and the corresponding animal
5 points - the *get_fact_for_animal* method returns the fact of the animal
5 points - the *get_fact_for_animal* method returns "Sorry! I do not have any facts for {name_of_animal}" when there is no fact for that animal
10 points - *print_history* prints "[index] Fact - Animal" for each of the facts in the fact_history_list in order and on a separate line.

This grading rubric shows how you will gain points, but not all the ways you could lose points.

Extra Credit - 6 points

Implement the following method: Create the ***generate_n_facts*** method. It takes a number as an input: n, Ex: 200. It generates random fact n times and returns the index and length of the longest consecutive run for an animal index. A run is a repetition of the same number consecutively in a list.

Ex: If 10 facts generated are [1,5,6,3,2,4,1,**4,4,4**] then three 4's is the longest run

Hence the function should return "**longest run was length of 3 for index 4"**

Extra Credit Example Output:

```
Testing generate_n_facts method with 200 facts
longest run was length of 5 for index 3
```

Sources for facts : https://www.nationalgeographic.com/animals/facts-pictures/
https://www.indiatoday.in/education-today/gk-current-affairs/story/20-interesting-general-knowledge-facts-divd-1591651-2019-08-26
https://www.purina.co.uk/dogs/behaviour-and-training/understanding-dog-behaviours/amazing-dog-facts
https://viagenpets.com/fun-cat-facts/