Python Fundamentals
Exercises

1. Write a Python script that prompts the user for a number and displays a message that does indicate if the number is odd or even.
   Note: do consider 0 has being neither odd nor even.
   After having tested a first number, the script should prompt the user for other numbers as long as he or she would like to.

2. Write a Python script that prompts the user for several numbers (when the user enter the string "stop", the script stop asking for numbers).
   The numbers entered are stored into a list one after the other.
   After having retrieved all the numbers, print the list.
   The script will then computes and print the minimum, the maximum and the mean of the different numbers present in the list.

3. Write a Python scripts that implements the "Guess the Number" game.
   The script will generate of a random integral number (use the module random) from 1 to 100, and ask you to guess it.
   The script will tell you if each guess is too high or too low.
   You win if you can guess the number within six tries.

4. Write a Python script that defines a small French/English dictionary (5 or 6 words will be enough). The user of the script will be prompted for a French word and the script will display its corresponding translation in English.
   If the word is not present in the dictionary, the script will print all the words it is able to translate.
   Note: the script should be case insensitive.

5. Write a pythons scripts that, given a list of words (a list of str), creates a dictionary where the keys do correspond to the word lengths and the values a list of the words with the corresponding length.
   For instance, if the list is:

   ```
   words=["He", "does", "confess", "he", "feels", "himself", "distracted", "but", "from",
   "what", "cause", "he", "will", "by", "no", "means", "speak"]
   ```

   The dict should be:

   ```
   {   2:['He', 'he', 'by', 'no'],
       3:['but'],
       4:['does', 'from', 'what', 'will'],
       5:['feels', 'cause', 'means', 'speak'],
       7:['confess', 'himself'],
       10:['distracted']}
   ```

6. Write a function that do test that the argument it receives is a prime number.
   A prime number (or a prime) is a natural number greater than 1 that has no positive divisors other than 1 and itself.
   This function will take as argument the number to test and it will return a Boolean result: True is the number is primer, False if not.
   For instance:
   isPrime(5) --> True
   isPrime(12) --> False

   Note: If a number n is not a prime, it can be factored into two factors a and b: n = a*b
   If both a and b were greater than the square root of n, a*b would be greater than n. So at least one of those factors must be less or equal to the square root of n**: to check if n is prime, we only need to test for factors less than or equal to the square root of n**.

7. The Fibonacci numbers are defined recursively by the following equations:

   Fn=Fn-1 + Fn-2
   F1=1
   F0=0

   Here are the first elements in the Fibonacci sequence:

   0,1,1,2,3,5,8,13,21,34, ...

   Write a function that returns the n-th Fibonacci number (n being a positive integral argument received by the function).

   You can implement this function in 3 different ways:
   - Using a looping technique
   - Using recursion

8. Write a Python script that do define a function `addList()` that is able to add a list of numbers to another list of numbers:
   `addList([3,4,5] , [6,8,9,3,5])` should return *`[9,12,14,3,5]`*

   If the two lists are not of the same size, an optional argument will allow you to indicate which value should be added to the elements that have no counterpart in the other list:
   `addList([3,4,5] , [6,8,9,3,5], 5)` should return *`[9,12,14,8,10]`*

9. Modify exercise 4: a function should now be defined that given a French word returns its translation in English or raise an exception if no translation is available.

10. Write a Stack class using a composition mechanism (the Stack class will use internally the List class).
    The Stack class should offer the following facilities:
    - a push() method to add an element on top of the Stack
    - a pop() method to get and remove the element on top of the Stack
    - a peek() method to get the element on top of the Stack (without removing it)
    - a way to test if it is empty or not,
    - a way to determine its size (its current number of elements),
    - a way to test that a Stack is equal to another
    - a way to print easily the content of the Stack
    - a way to test if an element belongs to the stack or not

11. Now, implement the Stack class using an inheritance mechanism (the Stack class will inherit from the List class).
    Compare this version of the Stack class with the previous one.

12. Write a class Date to represent a date (day, month and year).
    A class Time to represent a time (hour, minute seconds).
    These classes should provide methods to create, initialise, modify, display, compare, … dates and times respectively.
    Then create a class DateTime that inherits from both Date and Time.

13. Given a file data.txt with the following content:

```
City  Temperature
Gland 23.4
Nyon 23.6
Geneva 22.8
Lausanne 24.4
Versoix 23.4
Gland 22.4
Nyon 21.8
Geneva 22.8
Lausanne 28.4
Gland 22.9
Nyon 22.8
Geneva 23.2
Lausanne 23.5
Versoix 22.9
….
```

Read the file "data.txt" and use it to create a dictionary within which the key will be a city name and the value associated with it a list of temperature.

Write a Python script that does answer the following questions:

- How many measures have been recorded for a given city (Geneva for instance) ?
- What are the temperatures recorded for a given city (Lausanne for instance)?
- What is the average of the temperature for a given city (Nyon for instance)?
- How many cities are present?

Create a summary file with a format like the following:

```
Nyon: [23.6, 21.8, 22.8] Average:22.73
Gland: [23.4, 22.4, 22.9] Average:22.90
```