**Exercise 1: Playing around with Strings.**

To extend the class for the strings to support the following operations:

1. to check if the string is palindrome, a string is palindrome when the represented sentence can be read the same way in either directions in spite of spaces, punctual and letter cases, e.g., detartrated, "Do geese see God?", "Rise to vote, sir.", ...
2. to subtract the letters in a string from the letters in another string, e.g., "Walter Cazzola"-"abcwxyz" will give "Wlter Col" note that the operator - is case sensitive and that the target should be a name containing an instance of the child class
3. given a dictionary of strings, to check if the string is an anagram of one or more of the strings in the dictionary

**Exercise 2: Social Networks.**

A **social network** is a social structure made of individuals (or organizations) called **nodes**, which are tied (**connected**) by one or more specific types of interdependency, such as friendship, kinship, financial exchange, dislike, sexual relationships, or relationships of beliefs, knowledge or prestige.

A **graph** is an abstract representation of a set of objects where some pairs of the objects are connected by links. The interconnected objects are represented by mathematical abstractions called **vertices**, and the links that connect some pairs of vertices are called **edges**.

The exercise consists of:

1. to implement the social network as a graph, i.e., to define the graph data structure with the operations you consider necessary to implement a social network
2. to implement an operation that visits in a convenient way **all** the elements of the graph, such an operation should be associated to the \_\_str\_\_ operation of the graph implementation
3. to test it against a dummy social network.

**Exercise 3: Pascal's Triangle.**

**Pascal's triangle** is a geometric arrangement of the binomial coefficients in a triangle.

The rows of Pascal's triangle are conventionally enumerated starting with row 0, and the numbers in each row are usually staggered relative to the numbers in the adjacent rows. A simple construction of the triangle proceeds in the following manner. On row 0, write only the number 1. Then, to construct the elements of following rows, add the number directly above and to the left with the number directly above and to the right to find the new value. If either the number to the right or left is not present, substitute a zero in its place.

Implement an iterator for the Pascal's triangle that:

1. at each step will return another iterator containing all the elements in the corresponding stage sorted as in the triangle
2. the iterator can go forward and backward
3. (**advanced**) it is possible to define the working algebra for the triangle, e.g., it could be on Z7 or on the alphabet