FOR14
Algorithms and computer programming with Python
Autumn 2018

Homework 4

The goal of this homework is to help you with the understanding of object oriented programming and be a comprehensive exercise of the content of the course.

Problem 1 Recall the problem of the phone book developed in Homework 2. You have to develop further this task using object oriented programming with the following structure:

1. You must provide a class called `PhoneBook` that will allow someone to create phone book objects containing one's contacts. This class must provide the following methods:
   - `addContact` to add new contacts using the contact objects define below. This method must avoid duplicates. A duplicate in this case is define as two objects that have all attributes identical. It also must ensure that all the names and surnames are capitalize.
   - `search` to search for a contact using the combination first name and last name, if there are several contacts with the same combination, it must list them all. This method must print the information of this contact, if there are several print the information for all of them.
   - `delete` to delete a contact using its first name and last name. There may be several contacts sharing the same combination of first name and last name, so this method must ask the user which of the contacts to delete.
   - `listByType` to list only business contacts or family contacts on the screen with all their information, and provide the option to sort them alphabetically by first name or last name.
2. You must provide a base class `Contact` with the attributes name, surname, and phone.
3. Extend the class `Contact` to define the following two sub-classes:
   - `BusinessContact`: this class must add the attributes `company`, and `businessAddress`.
   - `PersonalContact`: this class must add the attributes `homeAddress`, and `relationship`, the last attribute could store values such as: family, friend, acquaintance, or classmate.

Problem 2 Consider the following cities: Oslo, Bergen, Stavanger, Trondheim, Kristiansand. The traveling distance between this cities is defined as follows:

	Oslo	Bergen	Stavanger	Trondheim	Kristiansand
Oslo	–	6.47	7.10	6.10	3.56
Bergen	7.14	–	4.42	9.30	7.39
Stavanger	7	4.48	–	13.37	3.14
Trondheim	6.24	9.34	13.36	–	10.12
Kristiansand	4.3	7.45	3.33	10.17	–

Table 1: Distances between cities. The table shows the traveling distance from the city on the left margin to the city on the top margin. Note that the table is not symmetric.

Your task is to help a travelling salesman to create a tour starting at one city, visiting all the cities just once and coming back to the starting city. The goal is to create a tour trying to minimize the distance using the measures presented in Table 1. Someone came up with the following idea: arbitrarily pick a start city; then, each time the salesperson has to pick the next city to visit, pick the closest unvisited city. Don't forget that at the end the salesman has to return to the starting city.
For this problem you have to do the following:

1. Provide an algorithm for the strategy proposed where the input, the output, and the steps are clearly defined.
2. Classify this algorithm in one of the three main strategies presented in class: Dynamic programming, recursion, greedy algorithms.
3. Implement your algorithm in python using object oriented programming. You have to provide the definition for the following two classes:
   - `Tour`: this class must have at least two methods:
     - `addCity`: this allow the user to add a city to the tour.
     - `findTour`: here is where you implement the algorithm to find a tour and it must return the order of the cities to visit and the total distance.
   - `City`: this class is used to represent a city in the tour and must store the distances to the other cities. Also, it must have a method called `distanceTo`, which takes a specific destination as an argument and returns the distance to that specific destination.
4. Test your implementation with the data provided in Table 1. For the test print the tours obtained when the starting city is Bergen, and when the starting city is Oslo.

For this homework you have to provide a pdf file with the answers for numerals 1 and 2 of Problem 2. All the code must have a good level of documentation. You must deliver two python files named as follows:

- g_your group number_phonebook.py
- g_your group number_TSP.py