Handling data stored in different formats

1 Simple text file

As a first example, we want to handle a file containing contacts (names and phone numbers), see Figure 1 and http://www.barsamian.am/2021-2022/S7ICTB/TP8_Contacts.txt[1].

Alice	0606060608	Frédérique	0606060613
0606060606	Djamel	0606060611	Isabelle
Bob	0606060609	Guillaume	0606060614
0606060607	Étienne	0606060612	Jérôme
Charles	0606060610	Hector	0606060615

Figure 1: File containing our contacts' phone numbers.

A skeleton of python code to manipulate these contacts, given in Listing 1, can be downloaded from http://www.barsamian.am/2021-2022/S7ICTB/TP8_Contacts.py. It is important that you put the text file with the python file in the same folder (if you rename the text file, also change the name in the python program on line 7).

```
1  nbContacts = 10
2  names = [""]*nbContacts
3  phones = [0]*nbContacts
4
5  # "iso-8859-1" is a common encoding. On Linux, the standard encoding is
6  # "utf-8" and on Windows you can also encounter "cp1252".
7  f = open("TP8_Contacts.txt", "r", encoding="iso-8859-1")
8  # strip() removes blank characters at the beginning and the end of the string,
9  # here in particular the end of line characters left by readline()
10 for i in range(nbContacts):
11     names[i] = f.readline().strip()
12     phones[i] = f.readline().strip()
13 f.close()
14
15 s = input("What contact do you want to search for ? ")
16 for i in range(nbContacts):
17     if (s == names[i]):
18         print(phones[i])
```

Listing 1: Skeleton code to handle our contacts file.

1. When prompted, you type "Alice". What does the program print?
2. When prompted, you type "charles". What does the program print? Modify the program so that it prints what you would expect in this situation[2].
3. When prompted, you type "etienne". What does the program print? This time, it is not easy to modify the program to have it return what you want. Use the function given in Listing 2 to perform the task — and don't forget the remark from the previous question.

```
1 import unicodedata
2 # This removes diacritics: accents, diaereses, umlauts, tildes, cedillas...
3 def normalize(text):
4     nfkd_form = unicodedata.normalize('NFKD', text)
5     return "".join([c for c in nfkd_form if not unicodedata.combining(c)])
```

Listing 2: Skeleton code to normalize strings.

[1] You can also use http://www.barsamian.am/2021-2022/S7ICTB/TP8_Contacts_utf8.txt, encoded in utf-8.
[2] We have seen it in exercise 3 of Work n°3 : http://www.barsamian.am/2021-2022/S7ICTB/TP3_Loops.pdf.

4. Add another person in `TP8_Contacts.txt`, then rerun the program. What happens? How can you fix the error? Can you fix the error so that no matter the number of contacts, the program will run without errors?

5. You now know another person also named Bob. Add this person in `TP8_Contacts.txt`, then rerun the program and ask for Bob's phone number. What happens? What would you expect? Can you fix this problem?

BONUS In previous questions, the program did not print anything when no contact was found. Please print an error message in that case.

PS: This first way of looking at the data is not the best. In fact, in this situation, we would use a data structure called a dictionary, but we won't look at those details.

2 Comma-separated values file

In previous section, the contact data was organized in a very simple fashion:

Line1: Name of contact 1

Line2: Phone of contact 1

Line3: Name of contact 2

Line 4: Phone of contact 2

etc.

You can find elements of correction:
http://www.barsamian.am/2021-2022/S7ICTB/TP8_Handling_data_1_correction.pdf.

Now, the goal is to learn how to handle data formatted into a specific way. We will work on the ".csv" format. csv means comma-separated values. This means that on each line you have different values (as opposed to only one value per line), and that those values are (usually) separated by commas (","). In fact, in the file we will work on, the values are separated by semi-colons (";").

The basic way to read the file is still the same: we open the file, and read it line by line. We will see a more convenient way to loop through the lines, and we will see how to make `Python` understand that we have different values on each line.

In this problem, we want to handle a file containing GPS coordinates of French cities. The file is in French, but it should not be a problem. You can see the beginning of the file on Figure 2, and you can download it at http://www.barsamian.am/2021-2022/S7ICTB/TP8_Cities.csv.

```
Code_commune_INSEE;Nom_commune;Code_postal;Libelle_acheminement;Ligne_5;coordonnees_gps
80355;FRESNEVILLE;80140;FRESNEVILLE;;49.9469630616, 1.753960976
80365;FRICAMPS;80290;FRICAMPS;;49.7720118421, 1.95186211928
80368;FRIVILLE ESCARBOTIN;80130;FRIVILLE ESCARBOTIN;;50.0912781795, 1.52364516053
80379;GLISY;80440;GLISY;;49.8341850031, 2.39954269272
80387;GRATTEPANCHE;80680;GRATTEPANCHE;;49.8142899245, 2.29952854065
80393;GRUNY;80700;GRUNY;;49.7015900422, 2.77539756139
80399;GUIGNEMICOURT;80540;GUIGNEMICOURT;;49.872686231, 2.06894684401
```

Figure 2: File containing GPS coordinates of French cities.

A skeleton of `python` code to manipulate this file is given in Listing 3, and can be downloaded from http://www.barsamian.am/2021-2022/S7ICTB/TP8_Cities.py.

```
city = input("What city are your searching for? ")

# "iso-8859-1" is a common encoding. On Linux, the standard encoding is
# "utf-8" and on Windows you can also encounter "cp1252".
f = open("TP8_Cities.csv", "r", encoding="iso-8859-1")
# strip() removes blank characters at the beginning and the end of the string,
# here in particular the end of line characters left by readline()
for line in f:
    values = line.strip().split(";")
    # add your code here !
f.close()
```

Listing 3: Skeleton code to handle our cities file.

1. The `split` method transforms a string into an array of strings. Each string of the array is a sub-string of the original string, contained in two consecutive places where the pattern (here, ";") is found in the original string.

   For example, `"a+24x+30-53t+10".split("+")` is an array of size 4 (there are 3 "+" in the string, thus they separate the string in 4 distinct sub-strings), where the first cell is `"a"`, the second cell is `"24x"`, the third one is `"30-53t"` and the last one is `"10"`. In other words:

   `"a+24x+30-53t+10".split("+")` is the array `["a", "24x", "30-53t", "10"]`.

   When the python program reads the first line of the csv file, what is the content of the variable `values` after the instruction `values = line.strip().split(";")`?

2. You want to find the line containing information about a particular city (e.g., you're looking for LESBOEUFS). Modify the python file to print the value of the variable `values` when you have found the correct city. Which cell of the variable `values` has to be read if you want to extract its GPS coordinates?

3. Modify the program to extract the GPS coordinates of a city given as input by the user. The coordinates will be printed on the screen.

BONUS Modify the program to extract the GPS coordinates of two cities given as input by the user. Then, have the program print the distance between the two cities.

Hint: the cell that holds the GPS coordinates can itself be split, because it is always "latitude, longitude".

Hint2: some explanation on how to deduce the distance from the coordinates can be found on the following link:
https://stackoverflow.com/questions/365826/calculate-distance-between-2-gps-coordinates