Laboratory 1 - Part I (Python)

Exercise 1 (adapted from Computer Sciences exam 03/09/2012)

Write a program able to handle the scores of an artistic gymnastic event. The event scores are contained in a file whose name is passed from the command line. Every line of the file contains: the competitor's name and surname, competitor's nationality, and the assigned evaluations provided by 5 judges. For example, the file (`score.txt`) contains:

```
Donald Duck ITA 9.3 8.9 9.7 9.7 9.8
Mickey Mouse ITA 9.0 9.0 9.0 9.2 9.5
Bugs Bunny USA 8.4 8.7 8.5 8.6 9.0
Daffy Duck RUS 8.3 8.8 9.5 9.6 9.0
Charlie Brown GRB 8.2 8.9 8.9 8.6 9.3
```

Assumptions:

- The total number of records (lines) is not known
- The competitor's name and surname do not contain spaces
- There are always 5 evaluations for every competitor, and these numbers are separated by a space

The program should show:

- The final ranking for the best 3 competitors: consider that for computing the final records, the highest and lowest evaluations are ignored and the score is determined by the sum of the remaining 3 values.
- The best country: the one that obtained the best score considering the sum of all the competitors of this country (always ignoring for every competitor the highest and lower evaluations).

For the provided file, the program output should be

```
final ranking:
1: Donald Duck - Score: 28.7
2: Daffy Duck - Score: 27.3
3: Mickey Mouse - Score: 27.2

Best Country:
ITA - Total score: 55.9
```

Exercise 2 (adapted from Computer Sciences exam 23/06/2017)

You are asked to implement a program for managing the database of a city public transportation center. The information is stored in a file whose name is passed as command line argument. Each line in the file contains: the ID number of the public transport bus, the number of the route the bus serves, the geometric coordinates in meters (abscissa and ordinate, i.e., x-axis and y-axis) and the time in seconds (all the time values belong to the same day) when the bus is checked. For example, the file can contain:

```
2187 13 10 1003 18000
3002 4 5000 5 18100
2187 13 100 2030 18500
3002 4 5000 1100 18600
2187 13 300 3300 19200
3002 4 5000 2200 19200
1976 4  5000 5 18600
1976 4  5000 1100 19600
1976 4  5000 2200 20100
```

We make the following assumptions:

- All the file can be loaded in memory
- Each bus serves only one line
- Multiple buses can serve the same line

The program receives the following parameters from the command line: 1) the name of the file containing the database and, 2) a flag, followed by an additional parameter.

- If the flag is `'-b'`, the next parameter is a busId. The program should print the total distance traveled by the given bus
- If the flag is `'-l'`, the next parameter is a lineId. The program should print the average speed of buses traveling on the line

For example:

```
$> python lab_1_2.py lab_1_2.txt -b 1976
1976 - Total Distance: 2195.0

$> python lab_1_2.py lab_1_2.txt -l 4
4 - Avg Speed: 1.6884615384615385
```

Exercise 3 (adapted from Computer Sciences exam 14/02/2020)

A text file contains information on a group of people born in a given year. The format is the following:

`<name> <surname> <birthplace> <birthdate>`

The first three fields are strings (with no blanks), `<birthdate>` is a string with format DD/MM/YYYY/ Each line corresponds to a person, and births are not sorted. Write a program that computes

- The number of births for each city
- The number of births for each month
- The average number of births per city (number of births over number of cities)

Example:

```
Mario Rossi Torino 02/03/2019
Franca Valeri Asti 10/05/2019
Marco Verdi Torino 05/04/2019
Giancarlo Magalli Torino 01/06/2019
Giovanna Bianchi Asti 10/03/2019
```

The program should output (in no particular order)

```
Births per city:
  Torino: 3
  Asti: 2
Births per month:
  March: 2
  April: 1
  May: 1
  June: 1
Average number of births: 2.50
```

Exercise 4 (adapted from Computer Sciences exam 04/02/2013)

Write a program to track available copies and sales of a bookstore. Sales informations are provided in a file, with format

`<ISBN> <BUY/SELL> <DATE> <#-OF-COPIES> <PRICE-PER-COPY>`

The `<BUY/SELL>` field contains either `B` (the books were bought) or `S` (the books were sold). `#-OF-COPIES` represents the number of bought/sold copies for the transaction. Each line of the file contains one transaction. `<DATE>` is in the DD/MM/YYYY format.
The program should output

- The number of available and sold copies for each book (ISBN)
- The number of books sold for each month / year combination (print only months in which books were sold)
- The total gain and average (per copy) gain for sold books.

Example:

For an input file with contents

```
978-1-932698-18-3 B 01/09/2012 3 34.56
988-1-942768-22-4 B 05/09/2012 5 56.12
956-2-123568-58-9 B 11/10/2012 7 22.12
945-5-896589-36-5 B 21/10/2012 6 12.56
988-1-942768-22-4 S 05/11/2012 1 76.12
978-1-932698-18-3 S 22/11/2012 1 44.86
956-2-123568-58-9 S 04/12/2012 4 32.52
945-5-896589-36-5 B 11/12/2012 8 16.78
945-5-896589-36-5 S 21/12/2012 3 24.66
988-1-942768-22-4 S 23/12/2012 1 76.12
```

the output should be:

```
Available copies:
    945-5-896589-36-5: 11
    956-2-123568-58-9: 3
    978-1-932698-18-3: 2
    988-1-942768-22-4: 3
Sold books per month:
    November, 2012: 2
    December, 2012: 8
Gain per book:
    945-5-896589-36-5: 29.1 (avg 9.7, sold 3)
    956-2-123568-58-9: 41.6 (avg 10.4, sold 4)
    978-1-932698-18-3: 10.3 (avg 10.3, sold 1)
    988-1-942768-22-4: 40.0 (avg 20.0, sold 2)
```

Exercise 5 (adapted from Computer Sciences exam 23/06/2014)

A room is composed of $N \times N$ tiles. A file contains the value of N, followed by the coordinates of lightspots (one per line) that illuminate the room. Each lightspot illuminates the tile it's placed on with intesity 1, the eight adjacent tiles with intesity 1/2, and the 16 surrounding tiles with intesity 1/5, as:

0.2	0.2	0.2	0.2	0.2
0.2	0.5	0.5	0.5	0.2
0.2	0.5	1.0	0.5	0.2
0.2	0.5	0.5	0.5	0.2
0.2	0.2	0.2	0.2	0.2

Write a program that computes the light intensity of each tile.
Suggestion: you can implement the matrix that represents the room as a list of lists `[[v00, v01, v02],[v10, v11, v12], [v20,v21, v22]]` or as a dictionary of keys `{(0,0): v00, (0,1): v01 ...` }. Try both solutions.

Example ($N = 7$):

Spotlight file:

```
7
0 0
2 3
4 3
```

Output:

```
1.0  0.7  0.4  0.2  0.2  0.2  0.0
0.5  0.7  0.7  0.5  0.5  0.2  0.0
0.2  0.6  0.9  1.2  0.7  0.4  0.0
0.0  0.4  1.0  1.0  1.0  0.4  0.0
0.0  0.4  0.7  1.2  0.7  0.4  0.0
0.0  0.2  0.5  0.5  0.5  0.2  0.0
0.0  0.2  0.2  0.2  0.2  0.2  0.0
```

Laboratory 1 - part II (numpy)

Exercise 6

Solve Excercise 5 using a 2D `numpy` array to represent the room.

Exercise 7

a. Write a function that, given two integers `m` and `n`, returns a $m \times n$ numpy 2D array with dtype `numpy.float64` whose element in position i, j has value $i * j$.

Example of return value for $m = 3, n = 4$:

```
array([[0., 0., 0., 0.],
       [0., 1., 2., 3.],
       [0., 2., 4., 6.]])
```

b. Write a function that, given a 2D numpy array, computes a normalized version of the array, where the normalization consists in scaling all the array columns so that the sum of the elements of each column is one (assume that no column has a sum of elements that is zero, i.e., there's no need to check for division by zero). The function should not modify the input array, but should return a new array.

Example: given the matrix

```
1.0 2.0 6.0 4.0
3.0 4.0 3.0 7.0
1.0 4.0 6.0 9.0
```

the result should be

```
array([[0.2 , 0.2 , 0.4 , 0.2 ],
    [0.6 , 0.4 , 0.2 , 0.35],
    [0.2 , 0.4 , 0.4 , 0.45]])
```

c. Repeat the excercise b. but normalize the matrix rows so that the elements of each row sum up to one.

Example: given the matrix

```
1.0 3.0 1.0
2.0 4.0 4.0
6.0 3.0 6.0
4.0 7.0 9.0
```

the result should be

```
array([[0.2 , 0.6 , 0.2 ],
       [0.2 , 0.4 , 0.4 ],
       [0.4 , 0.2 , 0.4 ],
       [0.2 , 0.35, 0.45]])
```

d. Write a function that, given a numpy array of any shape, returns an array with the same shape but all negative elements set to 0. The function should not modify the original array.

e. Write a function that computes the sum of the elements of a product of two matrices $\sum_{i,j}(\mathbf{AB})_{i,j}$

Exercise 8

Solve the first part (final ranking question) of Excercise 1 using `numpy` arrays to represent the scores (ignore the best country question). Assume the number of competitors is given in the first row of the scores file.
Suggestion: You can store the competitors' names in a list and the scores in a matrix, i.e. a 2D numpy array, and use array operations to compute the actual score of each competitor and to find the best three competitors. For the last point, check the documentation on `numpy.sort` and `numpy.argsort` for a simple way to extract the best three competitors.