Topics:
Tuples. Dictionaries. Lists. Strings.

---

1. Sorted List

Given a list of students with multiple values (*name*, *age*, *height* as a `tuple`), print it out in a sorted order so that it compares by names first, then by age, and by height in the end.

```
t = [('Serik',18,190),('Almas',19,175),('Abai',17,180),
('Almas',18,170),('Berik',17,180),('Serik',18,185)]
```

Ex:

```
>>>
Abai 17 180
Almas 18 170
Almas 19 175
Berik 17 180
Serik 18 185
Serik 18 190
>>>
```

---

2. Sum All

Write a function called `sumall` that takes any number of arguments and returns their sum.

Ex:

```
>>> sum(1, 2, 3)
TypeError: sum expected at most 2 arguments, got 3

>>> sumall(1, 2, 3)
6
>>> sumall(1, 2, 3, 4)
10
```

---

3. Most Frequent

(Exercise 12.1 in the textbook)
Write a function called `most_frequent` that takes a string and prints the letters in decreasing order of frequency. Find text samples from several different languages and see how letter frequency varies between languages. Compare your results with the tables at http://en.wikipedia.org/wiki/Letter_frequencies.
Get some book in Kazakh language to check your results for your language as well.

---

4. Message Statistics

Suppose that "*WhatsApp*" had a log file for all the messages on your device in the following format:

To Mama Mon Mar 8 23:59:03 2015
From Berik Fri Dec 13 13:02:45 2016
...

Write a program that finds the following:

- **a)** *Top "To" Man:* the person that you write the most to;
- **b)** *Top "From" Man:* The person that writes to you the most;
- *c) Day-Hour Stats:*
  - **i.** the distribution of the day of the week for each of the messages (sorted in descending order);
  - **ii.** the distribution of the hour of the day for each of the messages (sorted by hours).

Ex:

```
>>> message_stats('whatsapp.txt')
To: Zhan
From: Mama
-----
Sun 1155
Sat 994
Fri 177
Mon 56
Thu 11
Wed 2
Tue 5
-----
00 408
01 194
02 54
03 6
07 2
08 5
09 4
12 7
14 5
17 6
18 92
19 164
20 197
21 218
22 263
23 372
```

4B. Bonus

- **a)** Rewrite the above *Day-Hour Stats* to compute the <u>average</u> data (**Ex:** number of messages each Sunday on average, not in total);
- **b)** Add an extra <u>optional</u> parameter to the function to accept a year to get statistics for. By default it should do it for all time.

  Ex:

  ```
  >>> message_stats('whatsapp.txt',  2016)
  # should give the result for 2016 only
  ```