**IF100**

**Practice 9 - BONUS**

**Introduction**

The aim of this example is to practice sequences (i.e. list, string, range) and loops (for and while statements, and nested loops). The use of sequences and loops is due to the nature of the problem; that is, you cannot finish this practice assignment without using sequences and loops.   
 **Description**

In Practice 9, we have explained the D'Hondt method in detail, how it is used to calculate the seat allocation among the political parties in a given electoral district. However, we ignored an important concept: the barrage, which is also known as the threshold. In some cases, a threshold or barrage is set, and a political party which does not achieve that threshold across the country, does not get any seats allocated to it in any electoral district, even if it received enough votes to have some seats in some districts. For example, in Turkey, a 10% barrage/threshold value is currently used (although there are discussions whether we should change this value for future elections). This means that, if a political party has less than 10% of all the votes in Turkey, then that party does not get any seats in any district. Examples of countries using the D'Hondt method with such a threshold are Denmark (2%); Spain (3%); Slovenia (4%); Croatia, Romania, and Serbia (5%); Russia (5%); Belgium (5%, on a regional basis); and so on.

In this practicing example, you will

* calculate the seat distribution of each political party **across the country** (not just for a given district as in Homework 3) and **for different threshold values** (not just for the current 10% threshold),
* print the results on the screen, and
* plot them by using matplotlib module of Python.

Thereby, you will have a chance to detect critical threshold values which have a significant effect on the seat distribution.

**Prepared Dataset**

You will be using the same dataset as in Practice 9. All detailed explanations regarding the variables in this file were given in the Practice 9 document.

**Inputs and Outputs**

In this practicing example, you will input two different integers from the user which are *the lower threshold value* and *the upper threshold* value respectively. For simplicity, you may assume that both inputs will be entered as valid positive integers; and *the lower threshold value* is smaller than *the upper threshold value*. One other assumption is that *the upper threshold value* is smaller than the greatest country-wide percentage value retrieved by any party in the elections.

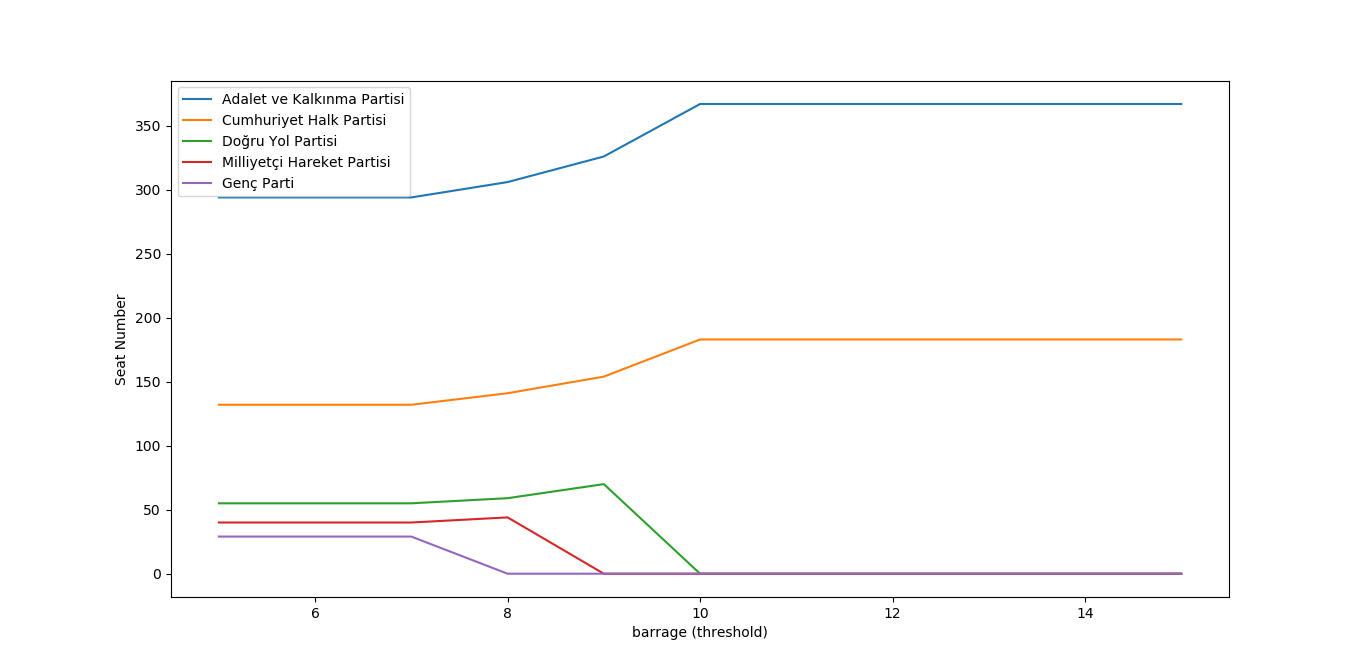
After you read the inputs from the user, you need to print seat distribution of the political parties in the same order that they occur in *partyNames* list for each integer threshold between *the lower threshold value* and *the upper threshold value* (both inclusive). In addition, you need to plot a chart that shows the seat distribution of each political party for different thresholds.

**Sample Runs**

Below, we provide some sample runs of the program that you will develop. The *italic* and **bold** phrases are inputs taken from the user. You may not change any of the prompt sentences. Your program should be presented exactly like these sample runs.

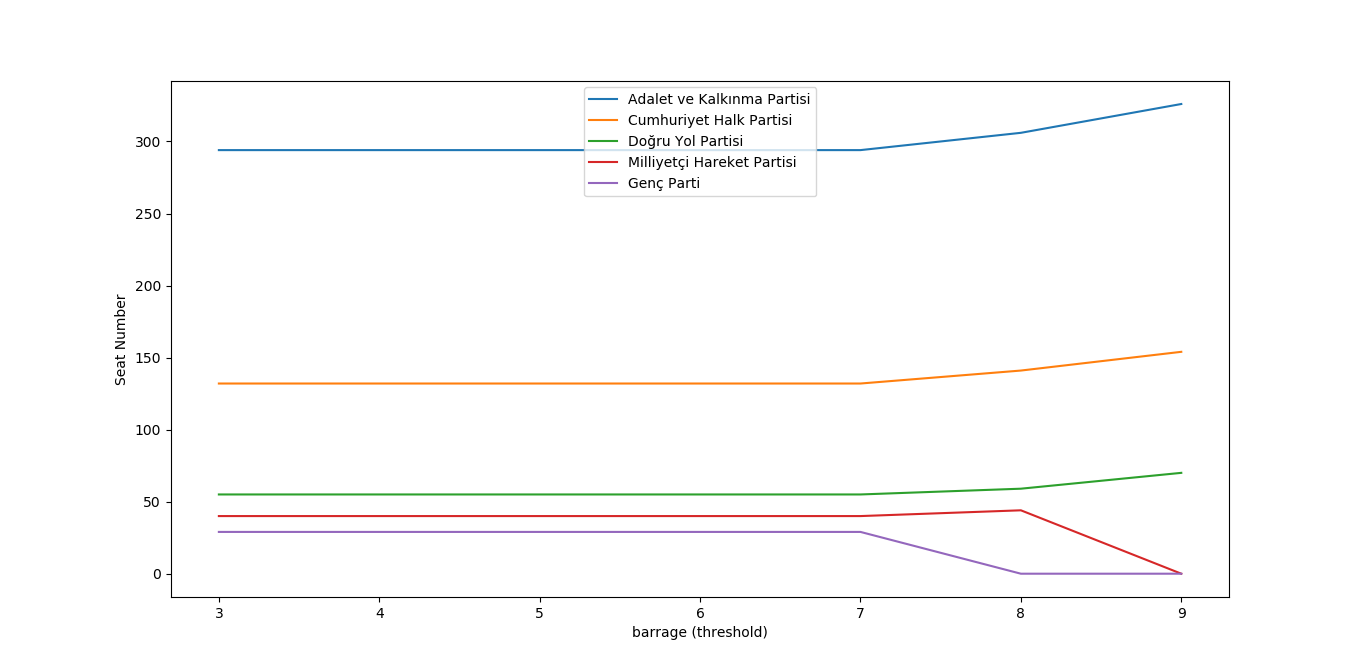
Sample runs are not %100 comprehensive. You are required to read the whole documentation and decide on what other cases you might try your program with.

**Sample Run 1**Please enter a lower bound for the threshold (barrage): ***5***  
Please enter an upper bound for the threshold (barrage): ***15***  
The result for threshold 5 is [294, 132, 55, 40, 29]  
The result for threshold 6 is [294, 132, 55, 40, 29]  
The result for threshold 7 is [294, 132, 55, 40, 29]  
The result for threshold 8 is [306, 141, 59, 44, 0]  
The result for threshold 9 is [326, 154, 70, 0, 0]  
The result for threshold 10 is [367, 183, 0, 0, 0]  
The result for threshold 11 is [367, 183, 0, 0, 0]  
The result for threshold 12 is [367, 183, 0, 0, 0]  
The result for threshold 13 is [367, 183, 0, 0, 0]  
The result for threshold 14 is [367, 183, 0, 0, 0]  
The result for threshold 15 is [367, 183, 0, 0, 0]

****

**Sample Run 2**

Please enter a lower bound for the threshold (barrage): ***3***  
Please enter an upper bound for the threshold (barrage): ***9***  
The result for threshold 3 is [294, 132, 55, 40, 29]  
The result for threshold 4 is [294, 132, 55, 40, 29]  
The result for threshold 5 is [294, 132, 55, 40, 29]  
The result for threshold 6 is [294, 132, 55, 40, 29]  
The result for threshold 7 is [294, 132, 55, 40, 29]  
The result for threshold 8 is [306, 141, 59, 44, 0]

The result for threshold 9 is [326, 154, 70, 0, 0] ****

**Sample Run 3**

Please enter a lower bound for the threshold (barrage): ***18***  
Please enter an upper bound for the threshold (barrage): ***30***  
The result for threshold 18 is [367, 183, 0, 0, 0]  
The result for threshold 19 is [367, 183, 0, 0, 0]  
The result for threshold 20 is [550, 0, 0, 0, 0]  
The result for threshold 21 is [550, 0, 0, 0, 0]  
The result for threshold 22 is [550, 0, 0, 0, 0]  
The result for threshold 23 is [550, 0, 0, 0, 0]  
The result for threshold 24 is [550, 0, 0, 0, 0]  
The result for threshold 25 is [550, 0, 0, 0, 0]  
The result for threshold 26 is [550, 0, 0, 0, 0]  
The result for threshold 27 is [550, 0, 0, 0, 0]  
The result for threshold 28 is [550, 0, 0, 0, 0]  
The result for threshold 29 is [550, 0, 0, 0, 0]  
The result for threshold 30 is [550, 0, 0, 0, 0]

