Project 8 CS 261

Save your program for this project as `<emailID>_project8.py` *where* `<emailID>` *is the part of your Hampden-Sydney e-mail address before the @ symbol. When you are finished, e-mail your program to* `blins@hsc.edu`. *Your solution is due by noon on Friday, November 1.*

Hamilton's Method

Last week we introduced Hamilton's method. It was invented to apportion the seats in Congress to the states based on their populations. Hamilton's method can be used to apportion other things too, but we still borrow the language of **seats**, **states**, and **populations** to talk about apportionment. To find the number of seats a state deserves, you need to find the **divisor** which is the ideal size of a congressional district:

$$\text{divisor} = \frac{\text{total population}}{\text{number of seats}}.$$

Then the number of seats each state deserves is the

$$\text{quota} = \frac{\text{population of state}}{\text{divisor}}.$$

The **lower quota** is the quota rounded down and the **upper quota** is the quota rounded up. Hamilton's method starts by giving every state its lower quota of seats in Congress. If there are any seats left over, the extra seats go to the states with the largest fractional parts in their quotas. No state ever gets more than its upper quota of seats.

1. In project 7 you wrote a function called `get_quotas()` that converts a dictionary with states as keys and populations as values to a new dictionary containing the quota for each state. We will need that function again. You can copy and paste your code or rewrite it for this project. If you copy and paste your old code, you might want to rename the variables to talk about states and populations, instead of electives and students.

2. Write a function called `sorted_values_and_keys()`. It should input a dictionary and return a list of tuples that each contain a value and its key from the dictionary. The tuples should be sorted according to the values, from smallest to largest. For example, if the input is `{"a": 4, "b": 3, "c": 5}`, then the output should be: `[(3,"b"), (4, "a"), (5, "c")]`.

   Hint: Once you have created a list of value/key pairs, you can use the `sorted()` function to sort the list. It will automatically sort the list based on the first item in each pair. It will only use the second item to break ties.

3. Write a function `top_keys(d, n)` that inputs a dictionary `d` and a positive integer `n` and returns a list containing the `n` keys with the largest values. Hint: You should use the `sorted_values_and_keys()` to do this.

4. Write a function `hamiltons_method(population_by_state, seats)` that inputs a dictionary with the population of each state and an integer with the number of seats to apportion, and returns a dictionary with the Hamilton's method apportionment for each state.

Test Cases

Use the following test cases to check if your `hamiltons_method()` function works correctly.

1. At Collegeville University, there are 4 divisions, (Arts, Business, Humanities, and Science). The school wants to apportion its 250 faculty as fairly as possible between the four divisions. The number of students per division is shown below, along with the correct apportionment.

```
students = {"Arts": 884, "Business": 1675, "Humanities": 1225, "Science": 216}
faculty = {"Arts": 55, "Business": 105, "Humanities": 77, "Science": 13}
```

2. In 1792, Congress passed a bill to apportion 120 seats in the House of Representatives to the states based on the 1790 census using Hamilton's method. The populations of the states from that census are given below, along with the correct apportionment.

```
populations = {"CT": 236841, "DE":  55540, "GA":  70835, "KY":  68705, "MD": 278514,
               "MA": 475327, "NH": 141822, "NJ": 179570, "NY": 331589, "NC": 353523,
               "PA": 432879, "RI":  68446, "SC": 206236, "VT":  85533, "VA": 630560}

apportionment = {"CT":  8, "DE":  2, "GA":  2, "KY":  2, "MD":  9,
                 "MA": 16, "NH":  5, "NJ":  6, "NY": 11, "NC": 12,
                 "PA": 14, "RI":  2, "SC":  7, "VT":  3, "VA": 21}
```

Hints

1. You'll need a way to get the total population of all the states. You can add up all of the values in a dictionary `d` using the command `sum(d.values())`.

2. In order to use the function `top_keys()` to find the states with the largest fractional parts in their quotas, you'll need a dictionary with those fractional parts. Maybe call it `quota_fractions`?