Exercise: Electoral College

Table of Contents

Background

The United States Electoral College is the official government body that determines who will be the president of the United States. Each state gets one elector for each senator and representative it has in the US Congress. The District of Columbia also gets three electors. All states except Maine and Nebraska have laws that stipulate that electors must vote for the candidate who received the most votes in their state. (For this exercise, we will pretend that Maine and Nebraska work this way, too.)

For this assignment, you will write an electoral college calculator. You'll work with two dictionaries, built from data in two separate text files. One dictionary will indicate the number of electors per state; the other will assign a hypothetical winning party ("R" or "D") to each state. You'll use those two dictionaries to determine who would win a presidential election based on the outcome of voting in each state. By modifying the contents of the file that assigns a party to each state, you can experiment with different potential outcomes. (The initial outcomes listed in the file are the outcomes of the 2016 presidential election.)

Instructions

Using the `electoral_college.py` template (see below), write a function called `get_winner()` with two parameters: a dictionary of electors and a dictionary of hypothetical outcomes. Both dictionaries will have the names of states as their keys. The dictionary of electors will have integer values (the number of electors by state). The dictionary of potential outcomes will have either "R" or "D" as its values. Your function should calculate the number of electoral votes received by each candidate ("R" or "D") and return a tuple consisting of the winning candidate and total number of electoral votes that candidate received.

Your function should have a docstring that briefly states

- what the function does
- what argument(s) it requires
- what kind of value it returns and what this value means

Remember that a function docstring should be the first statement in the body of your function.

We will also assume that your script is named `electoral_college.py`.

Template

electoral_college.py

```
from argparse import ArgumentParser
import sys


# Replace this comment with your implementation of get_winner().


def to_dict(filename, value_type=str):
    """ Read a two-column, tab-delimited file and return the lines as
    a dictionary with data from the first column as keys and data from
    the second column as values.

    Args:
        filename (str): the file to be read.
        value_type (data type): the type of the values in the second
            column (default: str).

    Raises:
        ValueError: encountered a line with the wrong number of columns.

    Returns:
        dict: the data from the file.
    """
    lines = dict()
    with open(filename, "r", encoding="utf-8") as f:
        for n, line in enumerate(f, start=1):
            line = line.strip()
            if not line:
                continue
            values = line.split("\t")
            if len(values) != 2:
                raise ValueError("unexpected number of columns on line"
                                 f" {n} of {filename}")
            lines[values[0]] = value_type(values[1])
    return lines


def read_data(elector_file, outcome_file):
    """ Read elector data and outcome data and return as dictionaries.

    Args:
        elector_file (str): path to the file of electors by state.
        outcome_file (str): path to the file of hypothetical outcomes by
            state.

    Returns:
        tuple of dict, dict: the elector data and outcome data. Keys in
        each dictionary will be state names.
    """
    elector_dict = to_dict(elector_file, value_type=int)
    outcome_dict = to_dict(outcome_file)
    return elector_dict, outcome_dict


def parse_args(arglist):
    """ Parse command-line arguments.

    Args:
        arglist (list of str): list of arguments from the command line.

    Returns:
        namespace: the parsed arguments, as returned by
        argparse.ArgumentParser.parse_args().
    """
    parser = ArgumentParser()
    parser.add_argument("elector_file", help="file of electors by state")
    parser.add_argument("outcome_file", help="file of outcomes by state")
    return parser.parse_args(arglist)


if __name__ == "__main__":
    args = parse_args(sys.argv[1:])
    elector_dict, outcome_dict = read_data(args.elector_file, args.outcome_file)
    candidate, votes = get_winner(elector_dict, outcome_dict)
    print(f"{candidate} would win with {votes} electoral votes.")
```

Running your program

Your program is designed to run from the terminal. To run it, open a terminal and ensure you are in the directory where your script is saved.

The program takes at two command-line argument: the name of the electors file string) and the name of the outcomes file(string). Below are some examples of how to use the program. The examples assume you are using a Unix based OS (macOS, Linux) and your program is called `electoral_college.py`. If you are using Windows, replace `python3` with `python`.

Input:

```
python3 electoral_college.py electors.txt outcomes.txt
```

Output:

```
R would win with 305 electoral votes.
```