

### 3.3.3 Pseudocoding

**After** giving you an overview of the election audit tasks, Tom wants to go over the steps required in detail. He's going to show you a technique commonly used by programmers to write steps of their code, which is called **pseudocode**. Pseudocode will make the audit easier to present to nontechnical colleagues and stakeholders.

To facilitate the design process, programmers use **pseudocode** to create models or flowcharts for their programs. Pseudocode is like a roadmap of what you think your code will look like or the steps you'll take to complete the task at hand.

*Pseudo* means "fake," so pseudocode is essentially fake code.

Pseudocode is an informal language that has no syntax rules and is not meant to be executed. The point of using pseudocode is to focus on the overall design of the program.

For example, let's say someone asks you how to wash clothes. You might break down this task into a series of basic steps, like this:

1. Open the lid of the washing machine.
2. Put clothes in the washing machine.
3. Turn on the water.
4. Add detergent.
5. Close the lid.

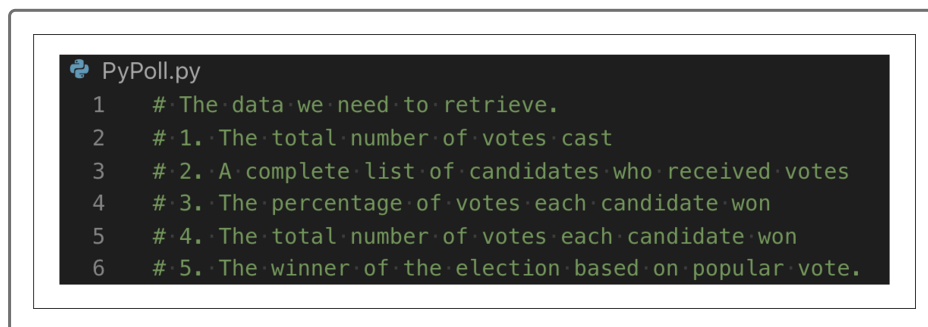
These well-defined, logical steps that are sequentially ordered is an example of an **algorithm**.

Similarly, a programmer or analyst may write the steps to count the votes of an election like this:

1. Open the data file.
2. Write down the names of all the candidates.
3. Add a vote count for each candidate.
4. Get the total votes for each candidate.
5. Get the total votes cast for the election.

A good place to start with when writing pseudocode is to think about the end goal. Then, consider the steps you need to take in order to reach that end goal. It's fine if you don't know all of the steps at first; this will come with practice. Also, the more you write pseudocode, the more of a feel you'll get for planning and design using pseudocode. Simply put, there is no right or wrong way to write pseudocode, as long as you're breaking down the problem into smaller, more manageable problems to solve.

Let's practice writing pseudocode. Launch VS Code and create a new file named `PyPoll.py`. Create a high-level list of the necessary steps given to you using pseudocode.



```
PyPoll.py
1 # The data we need to retrieve.
2 # 1. The total number of votes cast
3 # 2. A complete list of candidates who received votes
4 # 3. The percentage of votes each candidate won
5 # 4. The total number of votes each candidate won
6 # 5. The winner of the election based on popular vote.
```

Notice that in this pseudocode there is a hashtag, or number sign, before each comment: `#`. When the Python interpreter sees the `#` character, it ignores everything from that character to the end of the line. Using this technique is a great way to self-document your code so that others know what you are doing. We will have more practice at documenting our code later in this module.