In a single file called `hw3.py` write a Python function to do each of the following (one function per numbered question):

1. Play the [FizzBuzz game](https://en.wikipedia.org/wiki/Fizz_buzz) with the numbers 1-100, inclusive. This is a well-known intro task to get you comfortable with basic control flow logic (hint: in Python, “else if” is written as “elif”)
   1. Print the numbers 1-100, replacing all numbers divisible by 3 by “Fizz”, all numbers divisible by 5 by “Buzz”, and all numbers divisible by both with “FizzBuzz”.
   2. At the end, print the time (in seconds) that it took for your program to do the above.
2. The above is the formula for the volume of a sphere. Given a number R as input, return the volume of a sphere with radius R.Text

   Description automatically generated
3. Comma-separated values (CSV) is a popular format for storing data. An example excerpt of a .csv file is shown above. Given a dictionary that maps each of the column names in the above (“Title”, “Author”, “ISBN13”, “Pages”) to a list of values (e.g. [“1984”, “Animal Farm”, …]), write a CSV file containing that information in the form shown above, and return the filename.  
     
   You don’t have to recreate all of the input above! Feel free to make a dict with a few rows of dummy data for testing purposes.  
     
   Text

   Description automatically generated
4. Do the reverse of the above; given a CSV filename with the formatting from question 3, return a dictionary that maps column names to lists of values in that column.  
     
   Note: You can do questions 3 and 4 in any order. The output of one will be the input to the other. It may be easier to start with a sample CSV, do question 4, and then use the output as input for question 3.
5. Often, when testing functions that write or read from files, we want to create temporary resources that will be automatically cleaned up at the end of a test, since unit tests will often run thousands of times in the span of a couple days. Combine questions 3 and 4 into a single function (so it will take a dictionary and return the same dictionary, after writing to and reading from a file). The twist is that **you must use the `tempfile` library** (https://docs.python.org/3/library/tempfile.html) to make sure the CSV you create and read from is automatically cleaned up at the end of the function!