CS 100 Programming I
Project 0

Revision Date: October 2, 2015

Preamble

You may develop your code anywhere, but you must ensure it runs correctly on a Linux distribution before submission.

Statistics!

There are lies, damned lies, and statistics – Benjamin Disraeli

Yes, it is the season for football. However, get ready for something completely different, as this project will involve the computation of statistics for tennis not football. After all, another exciting US Open Tennis tournament has just finished. Your task is to prompt the user of your program to enter eight distinct numbers related to an individual tennis player and use those numbers to compute and report the following three statistics:

- number of serves attempted
- double fault percentage
- ace percentage

Note: You must report these three statistics IN THE ORDER listed above.

Program Organization

Create a *proj0* directory off of your *cs100* directory and move into *proj0*.

Name your main python file *tennis.py*. Create one more file named *stats.py*. The main function in the main file (*tennis.py*) should prompt the user of your program for information about an individual tennis player that includes the following:

- number of points won off of player's serve
- number of points lost off of player's serve
- number of first serves in

- number of second serves in
- number of lets
- number of aces on the first serve
- number of aces on the second serve
- number of double faults

Note: These eight numbers must be input IN THIS ORDER.

Your main function should call three different functions; each one should calculate and return one of the three required statistics specified above. The definitions of these three functions should be placed in the *stats.py* file. You will need to import the functions into your main program by placing the line:

```
from stats import *
```

at the top of *tennis.py*.

1 Sample statistics

Suppose you were asked to calculate the percentage of 'first serves in' that were not aces. You would place the following function into *stats.py*:

```
def notAcePercentage(fsi,fsa):
    return (fsi-fsa) / fsi * 100.0
```

where the *fsa* formal parameter is bound to the number of first serve aces. The meanings of the other formal parameters may or may not be apparent. Indeed, you will be docked points for having formal parameters, variables and function names that are not self-explanatory. In this example, the function name is self-explanatory while the formal parameter names are not.

You would call this function from your main function in *tennis.py* like this:

```
def main():
   # read in the information
   ...

   nap = notAcePercentage(firstServIn,firstServAces)
   print("The percentage of first serves not aces: ",nap)


   ...

main()
```

The following is an example only; do NOT include this code in the final version of your program.

Getting the numbers

Here is how to get a number and make the variable m point to it:

```
m = eval(input("Give me a number: "))
```

What is number of serves attempted, etc.?

The number of serves attempted includes first serves in, second serves in (notice this is the SECOND serve), number of lets and double faults (the important word here is DOUBLE). The double fault percentage is the number of double faults divided by the number of first serves in, second serves in and double faults. Notice that lets are not included since they are considered a redo. The ace percentage is the number of aces on either the first or second serve divided by the number of points won or lost on the player's serve.

Compliance Instructions

To make sure that you have implemented your program correctly, create a file with seven numbers. The numbers should be in the same order as specified in the previous section.

You should then be able to run the following command:

```
cat test0 | python3 tennis.py
```

The number of serves attempted calculated by your program should be 48, the double fault percentage should be 7.14, and the ace percentage should be 10.71. It's OK to have more decimal places.

This method of running the program is called "piping in the input from a file." When you actually do this, the prompts your program makes for information will all be strung together on a single line. Don't worry about it; it's a natural consequence of the way the program was run.

If your code fails with a runtime error while running this test, then you will receive a zero for this assignment.

Note that your answers do not have to be correct for your program to be graded, only that it not fail. Of course, correct answers will yield a much higher grade.

Note also, that to receive full credit, your stats functions, in *stats.py*, must return numbers, not strings.

Challenges

Try figuring out how to get your functions to return numbers like 0.326 rather than 0.325809201928.

Submission Instructions

Change to the *proj0* directory containing your assignment. Do an *ls* command. You should see something like this:

```
tennis.py   stats.py   test0
```

Extra files are OK. Submit your program like this:

```
submit cs100 xxxxx project0
```

Remember to replace `xxxxx` with your instructor's name: e.g. galloway, brown or vrbsky.

Note that *project0* ends in a zero, not a capital O.