CSCE 110: Programming I

Lab #11 (100 points)

Due: Sunday, November 13th by 11:59pm

1 Please make sure you understand the following.

You may only use what we have discussed during the last 11 weeks of class. Please make sure to name your files correctly, incorporate user-defined functions, and do not use the `global` keyword in your programs. Otherwise, your programs will be penalized as stated in the grading rubric.

Please label your Python programs `q<num>.py`, where `<num>` is the question number. Take your time and make sure you understand everything in this lab before getting started. Also, make sure your programs match the output EXACTLY as given for each question.

2 Lab Questions

1. Write a program called `q1.py` that analyzes and prints file statistics. First, prompt the user for the name of a file. Then, read and analyze the file and display the following statistics: total number of characters, number of sentences, and number of words. Here are some notes regarding the above statistics you will collect.

   - A character is any symbol in the text file. Examples of characters include letters, numerical digits, punctuation marks, and "invisible" characters such as whitespace and newline characters.
   - Assume a word is separated by whitespace and/or newline characters.
   - Assume sentences end in either a period, question mark, or exclamation point.

   Programming tips. You'll need to take advantage of the `strip()` and `split()` functions for strings. Please see the Topic 3 lecture notes (along with in-class programs) for examples of reading files and stripping and splitting strings.

   Example #1. The user enters the filename `five-letter-words.txt` (line 1). The output of the program is then displayed (lines 3–5).

```
1 Enter a filename: five-letter-words.txt
2
3 Characters: 16500
4 Sentences: 0
5 Words: 2750
```

Example #2. The user enters the filename `gettysburg.txt` (line 1). The output of the program is then displayed (lines 3–5).

```
1 Enter a filename: gettysburg.txt
2
3 Characters: 1447
4 Sentences: 10
5 Words: 267
```

2. *Yahtzee.* Write a Python program (called q2.py) that simulates how often certain hands appear in Yahtzee. In Yahtzee, you roll five dice and then use those dice to make certain combinations. For example, the Yahtzee combination is formed by all five dice having the same value. A large straight occurs when all of the dice are in consecutive order (e.g., 1,2,3,4,5 or 2,3,4,5,6). If you haven't played Yahtzee, you can find out more from various online sources such as Wikipedia.

   Once the five dice are rolled, the following 7 hands in Yahtzee are of interest.

   - **Yahtzee:** all five dice show the same value
   - **Four of a kind (but not Yahtzee):** four dice showing the same value
   - **Three of a kind (but not four of a kind, full house, or Yahtzee):** three dice showing the same value
   - **Full house (but not three of a kind):** three of a kind and a pair
   - **Large straight:** five sequential dice
   - **Small straight (but not large straight):** four sequential dice
   - **Miscellaneous:** none of the above combinations

   Assuming the five dice are rolled n times, your program will determine the probability that each hand can be made with a single roll of the five dice. For example, if the dice were rolled four times, 3 of the rolls might be in the Miscellaneous category and one of the rolls might be a Small straight. As such, your program would report that 3 (75%) of the dice rolls were Miscellaneous and 1 (25%) of the rolls were a Small straight.

Directions for writing your program.

a) `q2.py` has been provided, where `roll_dice()` function has been written for you. `roll_dice()` returns a list of 5 random integers between 1 and 6. You are responsible for writing the `main()` function and the functions representing the different

Yahtzee hands. You can add any additional user-defined functions you may need to complete your program.

b) The user will enter the number of dice rolls at the `>` prompt (see example output). Suppose the user enters 10. Then, your simulation will show the number (as a percent) of rolls that were Yahtzee, Four of a kind, Three of a kind, etc.

c) The five dice are represented as a list. For example, the list `[2, 3, 6, 6, 1]`, says that the first die has a face value of 2, the second die has a face value of 3, the third die has a face value of 6, etc.

d) The parameter `dice` to each function is a local variable that is a list that contains the result of rolling 5 dice.

e) In the program provided, there is a function for each of the dice hand combinations in Yahtzee. When writing these functions, return the Boolean values True or False if the dice contains the combination of interest. For example, if the dice input to the `yahtzee` function is `[1,1,1,1,1]` (e.g., `yahtzee`([1,1,1,1,1])), then the function should return True. If the input to the `yahtzee` function is `[2,1,6,4,6]`, then the function should return False.

f) You will use the results of your functions to help you calculate the probability of the combinations of interest.

g) You might find sets and the sorted() built-in function to be especially useful for your program.

h) When testing your program, don't be afraid to try a large number of rolls like 100,000 or 1,000,000.

Note: Since random numbers are required, your program will not run in the same manner as the following example unless the same random numbers are generated.

Example. The user throws the dice five times (line 1), where the following Yahtzee hands appear: large straight (twice for 40.0%), full house (once for 20.0%), small straight (once for 20.0%) and miscellaneous (one for 20.0%). Yahtzee, four of a kind, and three of kind did not appear for any of the five rolls so their count and probability is 0 (lines 2 – 8).

The user increases the number of dice rolls to 2345 (line 9) and 10000 (line 17) to see how the number of occurrences and probabilities change for each Yahtzee hand of interest (lines 10–16 and 18–24, respectively). Once the user is done with the simulation, he or she types quit (line 25).

```
> 5
yahtzee: 0 (0.0 %)
four of a kind: 0 (0.0%)
large straight: 2 (40.0%)
full house: 1 (20.0%)
small straight: 1 (20.0%)
three of a kind: 0 (0.0%)
misc: 1 (20.0%)
> 2345
yahtzee: 2 (0.1 %)
four of a kind: 34 (1.4%)
large straight: 71 (3.0%)
full house: 83 (3.5%)
small straight: 206 (8.8%)
three of a kind: 391 (16.7%)
misc: 1558 (66.4%)
> 10000
yahtzee: 5 (0.1 %)
four of a kind: 183 (1.8%)
large straight: 317 (3.2%)
full house: 414 (4.1%)
small straight: 901 (9.0%)
three of a kind: 1545 (15.4%)
misc: 6635 (66.3%)
> quit
```