Unit 4

Apply what you know

0. Study the programs in the *Learn something new* section until you can write them yourself from scratch without relying on this document or any other source of information. Here are the programs:

    0.1. Write a module that contains the `average` and `cleanedup` functions. Then use these functions in a simple test program, importing them from the module.

    0.2. Write a program that runs simulations to find out how many $1 bets it takes on average to either go broke or double your money, assuming that you start with $10 and have an even chance of winning or losing any single bet. Use a function to simulate one game, returning the number of bets, and call the function many times to determine the average.

    0.3. Modify the previous program so that it includes a function for running an experiment in which an average is determined for any given set of starting bankrolls and any given number of games. Use the function to find the average length of games starting with $10, $20 and $40, using 2,000 games of each kind.

    0.4. Write a program that runs simulations to find out the chance that at least one student in a class of 30 will get his or her own paper if the teacher hands them out for grading at random. Use a function to simulate passing back the papers one time and another to run experiments by calling the first function repeatedly.

1. Write a program that prints the name Fred 100 times, one time per line.

2. Write a program that produces word-scramble puzzles. The program should choose words at random from *Pride and Prejudice*, display the letters of the word in a random order and challenge the user to guess the original word.

    2.1 Start by writing a function called `wordlist` that takes a filename and compiles a list of all the unique words found in the file. For example, `wordlist('pap.txt')` should return a list of the words in *Pride and Prejudice*.

    In writing this function you will want to add strings to a list only if they are not already in it. The condition `x not in y` holds if `x` is *not* one of the items in the collection called `y`.

    2.2 In order to scramble a string, you'll have to start by turning it into a list of individual characters. After the list is rearranged randomly, you'll want to put the characters back together into a single string. Write a function called `rejoin` that does this. The call `rejoin(['c', 'a', 't'])` should return the string `'cat'`.

2.3 Write a function called `scramble` that takes in a string and returns a string with the same letters in a random order.

Finally, use these three functions to write the puzzle program.

3. Most of the puzzles produced by the previous program are much too hard. Modify the program so that it gives the user a three-letter puzzle to start and then adjusts the number of letters up by one every time a correct answer is given and down by one for every wrong answer. In this way, the program will generally produce puzzles that are at the user's level.

   Rather than keeping all possible words in a single list, you should use a dictionary. If the dictionary is called `wordlists`, then `wordlists[7]` should be a list of all words in *Pride and Prejudice* that are seven letters long.

4. Write a program that randomly chooses a hand of five cards from a standard deck of playing cards and displays it for the user to see. Use a function that returns a shuffled deck, ready for dealing.

   A standard deck consists of 52 cards. There are 13 each of four **suits**: clubs, diamonds, hearts and spades. Within each suit, the 13 **face values** are: ace, 2, 3, 4, 5, 6, 7, 8, 9, 10, jack, queen and king. We identify a card by its face value and suit, e.g. the 2 of hearts, the jack of clubs, etc.

5. Modify the previous program so that, in addition to displaying a five-card hand, it reports the number of aces and the number of clubs in the hand. Use one function to determine the face value, given a string like '`queen of diamonds`', and another one to determine the suit.

6. Collect the card-related functions you have written into a module called `cards.py` and rewrite the previous program so that it begins by importing this module.