## 9.6 Exercises

1. The code below prints the numbers from 1 to 50. Rewrite the code using a while loop to accomplish the same thing.

   ```
   for i in range(1,51):
       print(i)
   ```

2. (a) Write a program that uses a while loop (not a for loop) to read through a string and print the characters of the string one-by-one on separate lines.

   (b) Modify the program above to print out every second character of the string.

3. A good program will make sure that the data its users enter is valid. Write a program that asks the user for a weight and converts it from kilograms to pounds. Whenever the user enters a weight below 0, the program should tell them that their entry is invalid and then ask them again to enter a weight. [Hint: Use a while loop, not an if statement].

4. Write a program that asks the user to enter a password. If the user enters the right password, the program should tell them they are logged in to the system. Otherwise, the program should ask them to reenter the password. The user should only get five tries to enter the password, after which point the program should tell them that they are kicked off of the system.

5. Write a program that allows the user to enter any number of test scores. The user indicates they are done by entering in a negative number. Print how many of the scores are A's (90 or above). Also print out the average.

6. Modify the higher/lower program so that when there is only one guess left, it says `1 guess`, not `1 guesses`.

7. Recall that, given a string `s`, `s.index('x')` returns the index of the first *x* in `s` and an error if there is no *x*.

   (a) Write a program that asks the user for a string and a letter. Using a while loop, the program should print the index of the first occurrence of that letter and a message if the string does not contain the letter.

   (b) Write the above program using a for/break loop instead of a while loop.

8. The GCD (greatest common divisor) of two numbers is the largest number that both are divisible by. For instance, gcd(18, 42) is 6 because the largest number that both 18 and 42 are divisible by is 6. Write a program that asks the user for two numbers and computes their gcd. Shown below is a way to compute the GCD, called Euclid's Algorithm.

   - First compute the remainder of dividing the larger number by the smaller number
   - Next, replace the larger number with the smaller number and the smaller number with the remainder.
   - Repeat this process until the smaller number is 0. The GCD is the last value of the larger number.

9. A 4000-year old method to compute the square root of 5 is as follows: Start with an initial guess, say 1. Then compute
$$\frac{1+\frac{5}{1}}{2}=3.$$
Next, take that 3 and replace the 1's in the previous formula with 3's . This gives
$$\frac{3+\frac{5}{3}}{2}=7/3\approx 2.33.$$
Next replace the 3 in the previous formula with 7/3. This gives
$$\frac{7/3+\frac{5}{7/3}}{2}=\frac{47}{21}\approx 2.24.$$
If you keep doing this process of computing the formula, getting a result, and plugging it back in, the values will eventually get closer and closer to $\sqrt{5}$. This method works for numbers other than 5. Write a program that asks the user for a number and uses this method to estimate the square root of the number correct to within $10^{-10}$. The estimate will be correct to within $10^{-10}$ when the absolute value of the difference between consecutive values is less than $10^{-10}$.

10. Write a program that has a list of ten words, some of which have repeated letters and some which don't. Write a program that picks a random word from the list that does not have any repeated letters.

11. Write a program that starts with an $5\times 5$ list of zeroes and randomly changes exactly ten of those zeroes to ones.

12. Write a program in which you have a list that contains seven integers that can be 0 or 1. Find the first nonzero entry in the list and change it to a 1. If there are no nonzero entries, print a message saying so.

13. In Chapter 4 there was a problem that asked you to write a program that lets the user play Rock-Paper-Scissors against the computer. In that program there were exactly five rounds. Rewrite the program so that it is a best 3 out of 5. That is, the first player to win three times is the winner.

14. Write a program to play the following simple game. The player starts with \$100. On each turn a coin is flipped and the player has to guess heads or tails. The player wins \$9 for each correct guess and loses \$10 for each incorrect guess. The game ends either when the player runs out of money or gets to \$200.

15. Write a program to play the following game. There is a list of several country names and the program randomly picks one. The player then has to guess letters in the word one at a time. Before each guess the country name is displayed with correctly guessed letters filled in and the rest of the letters represented with dashes. For instance, if the country is *Canada* and the player has correctly guessed *a*, *d*, and n, the program would display `-ana-da`. The program should continue until the player either guesses all of the letters of the word or gets five letters wrong.

16. Write a text-based version of the game *Memory*. The game should generate a 5 × 5 board (see the exercise from Chapter 8). Initially the program should display the board as a 5 × 5 grid of asterisks. The user then enters the coordinates of a cell. The program should display the grid with the character at those coordinates now displayed. The user then enters coordinates of another cell. The program should now display the grid with the previous character and the new character displayed. If the two characters match, then they should permanently replace the asterisks in those locations. Otherwise, when the user enters the next set of coordinates, those characters should be replaced by asterisks. The game continues this way until the player matches everything or runs out of turns. You can decide how many turns they player gets.

17. Ask the user to enter the numerator and denominator of a fraction, and the digit they want to know. For instance, if the user enters a numerator of 1 and a denominator of 7 and wants to know the 4th digit, your program should print out 8, because $\frac{1}{7} = .142856...$ and 8 is the 4th digit. One way to do this is to mimic the long division process you may have learned in grade school. It can be done in about five lines using the `//` operator at one point in the program.

18. Randomly generate a 6 × 6 list that has exactly 12 ones placed in random locations in the list. The rest of the entries should be zeroes.

19. Randomly generate a 9 × 9 list where the entries are integers between 1 and 9 with no repeat entries in any row or in any column.