Loop Practice

1. Write a program that simulates a stopwatch that records minutes, seconds, and hundredths-of-a-second. This program should start the stopwatch at time 0:00.00 (zero mins, secs, and 1/100 secs), and stop at 5:59.99. Use three nested loops to print all of these times increasing in order of time. Hint: the inner loop should keep track of the 1/100 seconds part; write this loop first, then add a loop outside of that one, then another one outside of that one.

   Hint: You can use print(format(number, "02")) to print a number with leading zeros.

2. Write a program that lets the user type in a number from the keyboard. The program should print out the pseudo-Roman numeral equivalent of the number. I say "pseudo" because we will simplify Roman numerals a bit by getting rid of the weird subtraction rules for Roman numerals. For example, normally 9 is written as IX = 10 – 1, but your program can print VIIII.

   In Roman numerals, M = 1000, D = 500, C = 100, L = 50, X = 10, V = 5, and I = 1.

   Use a loop that runs until the user's number becomes equal to zero. Inside the loop, write if statements that test how big the number is. If the number is bigger than or equal to one of the exact Roman numerals above, print that numeral, subtract the value from the user's number, and loop again.

   Challenge: make this work with "real" Roman numerals; e.g., for 9 it should print IX, not VIIII.

3. Write a function called count_factors that takes a single parameter called num. This function returns the number of positive factors of num; this is the number of positive integers between 1 and num, inclusive, that divide into num evenly. For instance, the number 10 has 4 factors: 1, 2, 5, and 10. So calling count_factors(10) should return 4.

   Do this by writing a for loop that counts from 1 to num and tests the remainder of dividing num by whatever the counter variable is. (You've done this before with a while loop, so use a for loop this time.)

4. Write a graphical game program, "Find the Hole". The program should use a random number generator to determine a circular "hole", selecting a point and a perhaps the radius around that point. These determine the target and are not revealed to the player initially. The user is then prompted to click around on the screen to "find the hidden hole". You should show the points the user has tried. Once the user selects a point that is within the chosen radius of the mystery point, the mystery circle should appear. There should be a message announcing how many steps it took, and the game should end.

5. Write a program that generates a random subtraction quiz for 1st graders. Your quiz should always include 5 questions. For each question, randomly generate two single-digit numbers; be sure to set the question up so that the smaller number is always being subtracted from the larger number. Ask the user for the solution. If the user is correct, generate the next question. If they are incorrect, keep asking them the same question until they get it correct. At the end of the quiz, display the total number of attempts it took for them to correctly solve all 5 problems.

6. Write a program that displays, ten per line, all the leap years in the 21st century (from year 2001 to 2100). The years should be separated by exactly one space.

7. Write a program that simulates a graphing calculator for a specific type of function (e.g., parabolas). For instance, let the user type in values for a, b, and c, and graph the equation y=ax^2 + bx + c.