3.5 Practice Programs

1) Write a program that asks the user for a positive even integer input n, and the outputs the sum 2+4+6+8+...+n, the sum of all the positive even integers up to n.

2) Write a program to take in a positive integer n > 1 from the user and print out whether or not the number the number is a perfect number, an abundant number, or a deficient number. A perfect number is one where the sum of its proper divisors (the numbers that divide into it evenly not including itself) equals the number. An abundant number is one where this sum exceeds the number itself and a deficient number is one where this sum is less than the number itself. For example, 28 is perfect since 1 + 2 + 4 + 7 + 14 = 28, 12 is abundant because 1 + 2 + 3 + 4 + 6 = 16 and 16 is deficient because 1 + 2 + 4 + 8 = 15.

3) Write a program that allows a user to play a guessing game. Pick a random number in between 1 and 100, and then prompt the user for a guess. For their first guess, if it's not correct, respond to the user that their guess was "hot." For all subsequent guesses, respond that the user was "hot" if their new guess is strictly closer to the secret number than their old guess and respond with "cold", otherwise. Continue getting guesses from the user until the secret number has been picked.

4) Write a program that asks the user to enter two positive integers, the height and length of a parallelogram and prints a parallelogram of that size with stars to the screen. For example, if the height were 3 and the length were 6, the following would be printed:

```
******
 ******
  ******
```

or if the height was 4 and the length was 2, the following would be printed:

```
**
 **
  **
   **
```

5) Write a program that prints out all ordered triplets of integers (a,b,c) with a < b < c such that a+b+c = 15. (If you'd like, instead of the sum being 15, you can have the user enter an integer greater than or equal to 6.) You should print out one ordered triplet per line.

6) Write a program that prompts the user for a positive integer $n \leq 46$ and prints out the n^{th} Fibonacci number. The Fibonacci numbers are defined as follows:

$F_1 = 1$, $F_2 = 1$ and $F_n = F_{n-1} + F_{n-2}$.

The reason the input is limited to 46 is that the 47^{th} Fibonacci number is too large to be stored in an integer. Your program should calculate the numbers the same way one would do by hand, by adding the last two numbers to get the following number in succession: 1+1 = 2, 1+2 = 3, 2+3 = 5, 3 + 5 = 8, etc.

7) Write a program using the turtle that draws a spiral triangle design, similar to the sprial square.

8) Write a program using the turtle that asks the user if they want a 5-pointed or 6-pointed star and draw the star. to extend the program, allow the user to enter any number 5 or greater and draw the corresponding star.

9) Write a program using the turtle that draws a track with several lanes. You may ask the user to enter an integer in between 1 and 8 for the number of lanes. A track typically consists of two straightaways with semicircles on both sides. A lane is enclosed between two of these shapes. Thus, a track with 6 lanes should have 7 figures of similar shape, enclosed in one another.

10) Write a program using the turtle that creates a random path. At each step, pick a random number of pixels to walk, followed by a random turn, anywhere in between 0 and 359 degrees. Allow the user to choose how many random steps the turtle will take. Adjust your program to allow the user to choose further parameters which direct the random walk. If this idea interests you, look up the term "random walk."