Lab – `for` loops

For loop syntax:

```
for var in range(start, finish, increment):     # var is a counter variable
    [ do whatever you want with var ]
```

1. Write a function called `f_to_c` that takes one float argument: a temperature in degrees Fahrenheit. This function will **return** the equivalent temperature in degrees Celsius. To convert a temperature from Fahrenheit to Celsius, take the Fahrenheit temperature, subtract 32 from it, then divide the result by 1.8. This gives you the equivalent temperature in degrees Celsius. ***This function will not use any input or print statements.*** Test this function from the Python shell by calling it with various Fahrenheit temperatures.

   Examples:

   ```
   >>> f_to_c(32)
   0.0
   >>> f_to_c(212)
   100.0
   ```

2. Without changing the `f_to_c` definition, write a `main()` function that uses a `for` loop to count from -40 to 100 by 10s. Your counter variable here will represent a Fahrenheit temperature. Inside the for loop, call `f_to_c` on the counter variable (the Fahrenheit temperature) and capture the return value that comes back (the Celsius temperature). Print both temperatures on the same line (as shown below), in one print statement. The effect of this will be a table of temperatures showing Fahrenheit and Celsius:

   ```
   -40 F = -40 C
   -30 F = -34.4444 C
   -20 F = -28.8889 C
   (etc)
   100 F = 37.7778 C
   ```

   Do not worry about rounding your output.

3. So far, your program has no input statements. Add input statements to your main function so the user can enter the low temperature to start at, the high temperature to finish at, and the degrees increment.

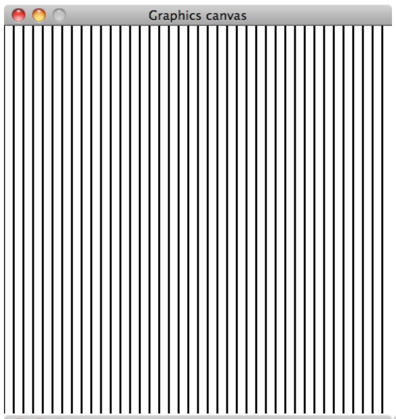
4. Write a function called `gcd` (for greatest common divisor) that finds the greatest common divisor between two integers. The GCD of *x* and *y* is the largest integer that both *x* and *y* can be divided by evenly. For example, the GCD of 50 and 20 is 10, because 10 goes into both 50 and 20 evenly, and there is no integer larger than 10 that does the same thing.

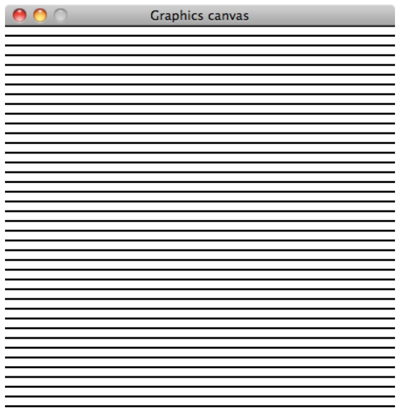
   To solve this using a `for` loop, have the loop test all the possible numbers that you could divide *x* and *y* by to see if the number goes in evenly. What should the starting and ending values of this loop be? Should the loop count up or down?

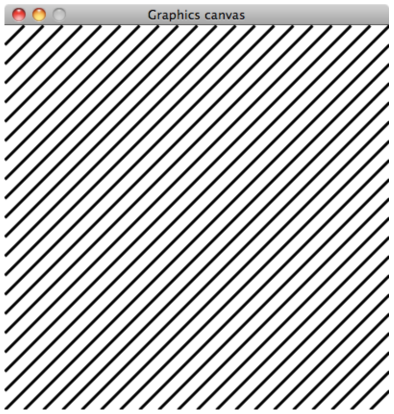
   The definition line should look like: `def gcd(x, y):`

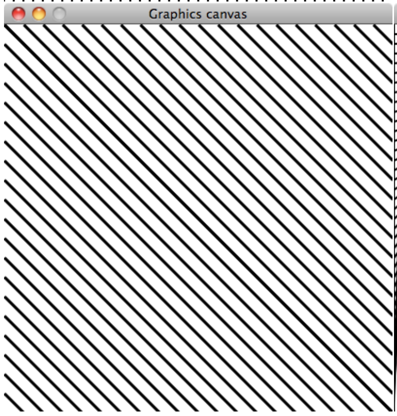
5. Use the `simplegraphics` library to draw the following designs. Use `for` loops to iterate through a set of numbers that you can use to calculate the coordinates of the endpoints of the lines in each figure. Each of the six pictures below represents a design completely covering a square canvas (you can pick the size of the canvas).

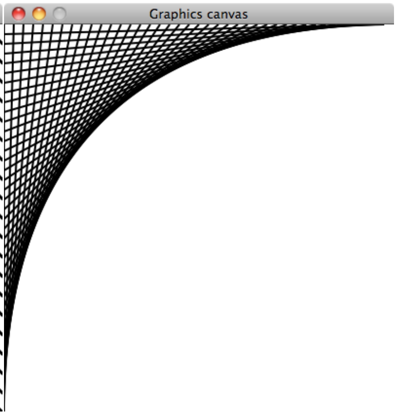
   (see back)

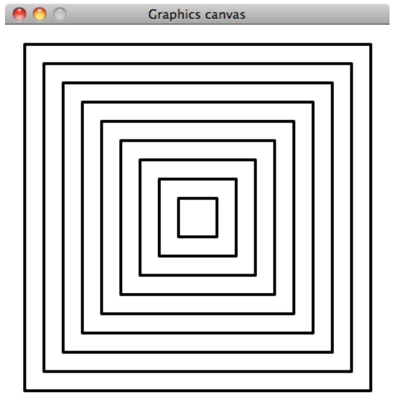












Also try drawing the fifth image with the curved design placed in the other three corners.

6. Challenge: draw a spiral. If you need sine/cosine, put "`import math`" at the top of your program, and then you will have access to the functions `math.sin` and `math.cos` functions.