Lab 2 – Functions, Generators, and List Comprehension

In your code file, add a comment line before each problem stating the problem number, for example "`# Problem 1`", etc.).

1. Create a function called `between()` that will return `True` if a given value lies in the range between two other values (inclusive), and `False` if it does not. The function must take 3 arguments: the first argument is the number to test, and the second and third arguments are the lower and upper bounds of the range, respectively. Design the function so that it may be called with 1, 2, or 3 arguments, with default values of the lower and upper bounds given by 0 and 0.3.

2. The Python `range()` function only works with integer values and integer step sizes, but often we need to be able to iterate over a sequence of floating point values and float point steps. In this problem you will make a custom function that will address this need, using a Python generator to yield an iterator that will produce values starting at 0 and ending at some maximum number (int or float), with a step between successive values that can also be either an int or float.

   Create a new function called `rangef(max,step)`, where `max` is the maximum value the iterator can produce, and `step` is the step size. Design the function as a generator, allowing it to be iterated over. Demonstrate that your generator works by executing the following code:

   ```
   for i in rangef(5,0.5): print(i, end=' ')
   ```

   which should produce the following output:

   ```
   0 0.5 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
   ```

3. Using your generator function from Problem 2, create a Python list called `alist` consisting of numbers from 0 to 1 in steps of 0.25 (*hint*: remember that `list()` will convert a non-list iterable into a list), and perform the operations in steps (a) and (b) below on the resulting list. Display the results via `print(alist)` after each step, including the initial list generation. Your initial print statement should yield `[0, 0.25, 0.5, 0.75, 1.0]`.

   a. Append an inverted version of the list onto itself and display the result. You may need to create a copy of the initial list in your solution. If so, make sure to use a deep copy rather than a shallow copy using one of the methods discussed in lecture. Your result should look like this:

      ```
      [0, 0.25, 0.5, 0.75, 1.0, 1.0, 0.75, 0.5, 0.25, 0]
      ```

   b. Using your `between()` function from Problem 1 and the list `sort()` method, sort the list such that values outside the range [0,0.3] are at the front of the list, and values inside the range [0,0.3] are at the end of the list.

4. Using list comprehension, and taking advantage of the `range()` function, write a single line of code to generate a list containing all integers in the range [0,16] that are evenly divisible by either 2 or 3.