Lab 1: Recursion and Python Testing

- Do not change any of the function names/parameters given in this assignment.

- In all cases, if the int_list parameter passed into a function is None, the function should raise the ValueError exception

- Write an iterative function to find the maximum integer in a list of integers. If list is empty, return None.

```
def max_list_iter(int_list):  # must use iteration not recursion
   """finds the max of a list of numbers and returns the value (not the index)
   If int_list is empty, returns None. If list is None, raises ValueError"""
```

- Write a recursive function to reverse a list of integers

```
def reverse_rec(int_list):   # must use recursion
   """recursively reverses a list of numbers and returns the reversed list
   If list is None, raises ValueError"""
```

- Write a recursive function to search a list of integers using binary search along with test cases. If the **target** of the search is in the list, the function returns its index. Otherwise, returns **None** (includes the case where the list is empty return **None**

```
def bin_search(target, low, high, int_list):  # must use recursion
   """searches for target in int_list[low..high] and returns index if found
   If target is not found returns None. If list is None, raises ValueError """
```

Details: The following files are given to you in your GitHub repository:

- **sample.py**
- **sample_test_cases.py**
- **lab1.py**
- **lab1_test_cases.py**

Test Cases

Many people tend to focus on writing code as the singular activity of a programmer, but testing is one of the most important tasks that one can perform while programming. Proper testing provides a degree of confidence in your solution. Systematic testing helps you to discover and then debug your code. Writing high quality test cases can greatly simplify the tasks of both finding and fixing bugs and, as such, **will save you time during development**. However, testing does not guarantee that your program is correct.

For this part of the lab you will practice writing some simple test cases to gain experience with the unittest framework. I recommend watching the first 20 minutes or so of the following video if you need more guidance on testing in Python. https://www.youtube.com/watch?v=6tNS--WetLI

Using your editor/IDE of choice, open the *lab1_test_cases.py* file. This file defines, using code that we will treat as a boilerplate for now, a testing class with a single testing function.
In the *test_expressions* function you will see some test cases already provided. You must add additional test cases to verify that your functions (max_list_iter, reverse_rec, bin_search) are correct.

Submission/Grading

Ensure that the following file have been pushed to GitHub by the due date:

- **lab1.py**
  - Correct and well documented iterative **max_list_iter**, recursive **reverse_rec**, and recursive **bin_search** functions based on the template provided (5 points)

- **lab1_test_cases.py**
  - A complete set of test cases for the functions above. (5 points) Your test cases should test boundary conditions and other possible errors based on the structure of your program. For each test provide a comment (docstring) that explains what it is testing. Your tests cases will be tested with known incorrect (buggy) versions of the functions in lab1.py and will also be tested for 100% code coverage.