Computer Laboratory 3
CSci 1913: Introduction to Algorithms,
Data Structures, and Program Development
February 7/8, 2017

0. Introduction.

This assignment asks you to write two recursive Python functions that take other Python functions as their arguments. This is the last Python lab for this course—although there is still one Python project yet to be assigned. It is worth 20 points.

1. Implementation.

Define and test these two Python functions. You must use recursion to define them: *you are not allowed to use loops or local variables*. Both these functions are short: if you find yourself writing many pages of code, then you do not understand the assignment.

`most(P, S)`

Here `P` is a function of one argument that returns either `True` or `False`, and `S` is a list. The function `most` calls `P` on each element of `S`. It must return `True` if `P` returns `True` more often than it returns `False`. It must return `False` otherwise. Here are some examples of how `most` must work, where the symbol '⇒' means *returns,* and where the function `odd` tests if a number is odd.

```
 most(odd, []) ⇒ False
most(odd, [0]) ⇒ False
most(odd, [1]) ⇒ True
most(odd, [1, 2]) ⇒ False
most(odd, [1, 2, 3]) ⇒ True
```

These are only examples! Your function `most` must work correctly for any `P`, and any `S` whose elements are compatible with `P`.

`sigma(F, B, E)`

Here `F` is a function of one argument that returns a number, `B` is a number, and `E` is a number. The function `sigma` must call `F` on all numbers from `B` to `E` and returns the sum of those calls. If `B` > `E` then `sigma` must return `0`. Here are some examples of how `sigma` must work, where the function `sqr` returns the square of its argument.

```
sigma(sqr, 0, 0)   ⇒ 0
sigma(sqr, 1, 0)   ⇒ 0
sigma(sqr, 0, 4)   ⇒ 30
sigma(sqr, 1, 1)   ⇒ 1
sigma(sqr, 2, 100) ⇒ 338349
```

These are only examples! Your function `sigma` must work correctly for any `F`, `B`, and `E`.

Hint: you may write additional functions that are called by `more` and `sigma` to help them do their jobs. However, these "helper" functions must also be recursive.

2. Tests.

The file `tests.py` on Moodle contains a series of tests. The tests call `most` and `sigma` and print what they return. To grade your work, the TA's will look briefly at your functions to see if they are recursive. If they are, then they will run the tests using your functions. If they are not, then you will receive 0 points for this lab. If a test behaves exactly as it should, then you will receive all the points for that test. If a test does anything else, then you will receive no points for that test. Your score for this assignment is the sum of the points you receive for all all the tests.

3. Deliverables.

Run the tests in the file `tests.py`. Then submit the Python code for your functions `most` and `sigma`. Do not submit the results of the tests. Your lab TA will tell you how and where to turn in your work. If your lab is on Tuesday, February 7, then your work must be submitted by Tuesday, February 14 at 11:55 PM. If you lab is on Wednesday, February 8, then your work must be submitted by Wednesday, February 8 at 11:55 PM. *To avoid late penalties, do not confuse these two dates!*