This is a pair programming assignment. Continue working with the partner you worked with for Part 1 of this Lab.

You shall continue adding functions to the `FindMax.py` package, and add additional testing for them to `maxTest.py`. For this part of the lab, you shall design and develop the following five functions:

- **findSecondLinear(a)** - this function takes an array `a` as input and returns back the second largest element of the array. The function shall use the linear approach to finding the largest element, and then find the second largest element among those that were compared with the largest.

- **findSecondLinearWithCount(a)** - this function takes an array `a` as input and returns back two values: the second largest element of the array, and the number of times two elements of the array were compared to each other. The function shall use the linear approach to finding the largest element, and then find the second largest element among those that were compared with the largest.

- **findMaxDNCWithComps(a,i,j)** - this function takes an array `a` as input, and returns two values: the largest value among `a[i]...a[j]`, `(i<=j)` and a list of indexes in the array `a[i,...j]` of the elements that were compared with with largest element. This function shall use the Divide-and-Conquer approach to computing the largest element (i.e., be recursive).

- **findSecondDNC(a)** - this function takes as input an array `a` and returns back the second largest value in it. This function shall call `findMaxDNCWithComps()` to find the largest element of the array. It can use a linear scan approach to finding the second largest element of the array, once the largest element is found.

- **findSecondDNCWithCount(a,i,j)** - this function takes as input an array `a` and returns back two values: the second largest value in the array `a` and the number of comparisons between two elements of the array. This function shall call `findMaxDNCWithComps()` to find the largest element of the array. This function is allowed to directly assume that the number of comparisons `findMaxDNCWithComps()` takes to be `n-1` (where `n` is the size of the array `a`) - i.e. there is no need to count comparisons inside the `findMaxDNCWithComps()`. It can also use a linear scan approach to finding the second largest element of the array, once the largest element is found.

Please note, your functions shall take the input array `a` as an instance of the NumPy array class. You can use `len(a)` or any other NumPy-specific syntax to obtain the length of the array, in case you need it.
maxTest.py: Create a test function in maxTest.py that compares the number of comparisons that findSecondLinearWithCount() and findSecondDNCWithCount() functions take. Because the number of comparisons that is needed by findSecondLinearWithCount() may differ on different arrays of the same size, your experiment shall work as follows:

1. Select several values for an array size. (E.g., [10, 20, 50, 100, 200]).
2. Select a number of repeats k (e.g., k = 30)
3. For each array size, generate k random arrays of that size, and use both functions to find the second largest element, and compute the number of comparisons used.
4. For each array size, compute the mean (average) number of comparisons each function took. (If you want to, you can also compute the standard deviation). You can make your own computations or use the NumPy methods for it.
5. Output the computed averages.

(eventually, we will be visualizing such results, but for now just printing them out is sufficient).

Be ready to discuss your observations during the lab period on Friday, September 29.

Deliverables. Continue building and maintaining the findMax.py and maxTest.py files as specified above.

There is no submission for this part of the Lab. Instructions for Lab 1, Part 3 will be given on Friday, September 29. This will be the final part of Lab 1.