# Midterm 1 practice

Consider an array of length N in descending order that we want to sort into ascending order

(1 point) How may inversions are there in the array?

(1 point) How many comparisons will be performed by insertion sort on this array, in Big O notation?

(3 point) Assuming we always choose the middle element of the subarray as the pivot, what is the expected runtime of running quicksort on this array? Why?

(2 points) What is the worst case runtime of this code snippet, using big O notation? `arr.length == N` and it contains integers between 0 and N (there could be duplicates or missing numbers).

```
for(int i = 0; i < N; i++){
    indexSum += linearSearch(arr, i);
}
```

(4 points) Fill in the method below so it returns the "largest" of the 3 parameters, in their natural ordering

```
public static <T extends Comparable<? super T> >
T maxOfThree(T x, T y, T z){
```