**Lab 2 – Sebastian Isaak**

**2_1)**
**b)**The asymptotic time complexity is O(n). The worst case scenario for my isPrime is n-1 times if we scale this up to a broader time complexity only n remains scaled towards infinity -1 does not matter.

**c)** By multiplying I by itself in the for-loop we also check for multiples of it. If it would be bigger than n the square root I can not be a prime number. Therefore we cut the time complexity by its square root which ends up being √n.

**2-2)**
**b)** The time complexity of countRight is O(n^2).

**c)** The time complexity of fastCountRight is O(n*log(n)) which is faster than O(n^2).

**2_3)**
**a)** The non-static methods return the max distance from the distance table by comparing one slot in the table against another and going through entry. The biggest value becomes the new maximum and gets used for comparison against the following values. The final maximum value out of the table gets returned.

**b)** Both methods run equally against infinity with a time complexity from O(n^2) but maxDistance2 skips about half of the values as the same values can be found in other rows and is therefore faster than maxDistance1.

**c)**
Because of the nested for-loop structure of the code both functions have a best and worst asymptotic time complexity of O(n^2). Even though in this specific example it seems like maxDistance2 is faster.