Neem Serra
Project 2
CSE 331 section 2

1. Run your program for data sizes in 10 steps. Choose these sizes so you have an even distribution and the maximum running time does not exceed 10 minutes (yes, I said minutes). For the number of elements you are sorting (not the number of items), indicate:

a. How long did the sort take?

b. How many actors, movies, and unique movies?

```
There are 8280 actors
Increment   : 680
Time passed : 0
Movies      : 17588
Verified    : true
Unque movies: 260

There are 8414 actors
Increment   : 690
Time passed : 0
Movies      : 17874
Verified    : true
Unque movies: 260

There are 8548 actors
Increment   : 700
Time passed : 0
Movies      : 18160
Verified    : true
Unque movies: 260

There are 8682 actors
Increment   : 710
Time passed : 0
Movies      : 18446
Verified    : true
Unque movies: 260

There are 8816 actors
Increment   : 720
Time passed : 0
Movies      : 18732
Verified    : true
Unque movies: 260

There are 8950 actors
Increment   : 730
```

Neem Serra
Project 2
CSE 331 section 2

```
Time passed : 0
Movies      : 19018
Verified    : true
Unque movies: 260

There are 9084 actors
Increment   : 740
Time passed : 0
Movies      : 19304
Verified    : true
Unque movies: 260

There are 9218 actors
Increment   : 750
Time passed : 0
Movies      : 19590
Verified    : true
Unque movies: 260

There are 9352 actors
Increment   : 760
Time passed : 0
Movies      : 19876
Verified    : true
Unque movies: 260

There are 9486 actors
Increment   : 770
Time passed : 0
Movies      : 20162
Verified    : true
Unque movies: 260
```

c. compare the maximum number of elements sorted to your results from Project 1.

I compared the same number of elements sorted from project 1 to project 2 and project 2 is substantially faster. In fact, all of the sorted things from project 2 say that there is a time of 0 that passed. The equivalent in project 1 varied from 7-20 in terms of time passed.

NOTE: to implement this in the code, simply change the following lines of code in Main.cpp:

- Line 40: unsigned increment = 10;

Neem Serra
Project 2
CSE 331 section 2

· Comment out lines 74 and 76.

2. The running time for Quicksort depends upon both the data being used and the partition rule used to select the pivot. Although Quicksort is *O(nlogn)* on average, certain partition rules cause Quicksort to take *Θ(n2)* time if the array is already sorted. For each of the following, justify your answer with a short paragraph.

a. Suppose we always pick the pivot element to be the key from the last position of the subarray. On a sorted array does Quicksort now take *Θ(n), Θ(nlogn)* or *Θ(n2).*

Quicksort will take $\Theta(n^2)$.

b. Suppose we always pick the pivot element to be the key from the middle position of the subarray. On a sorted array does Quicksort now take *Θ(n), Θ(nlogn)* or *Θ(n2).*

Quicksort will take *Θ(n log n).*

3. A stable sort is a sort that outputs equal keys in the same order that they were input. Prove or disprove the following: quicksort is a stable sort. Justify your answer in a few short paragraphs.

Quicksort is not a stable sort because it swaps elements of equal values. For a sorting algorithm to be stable, it has to output all of the keys in the same order that they were input. Quicksort does not do that.