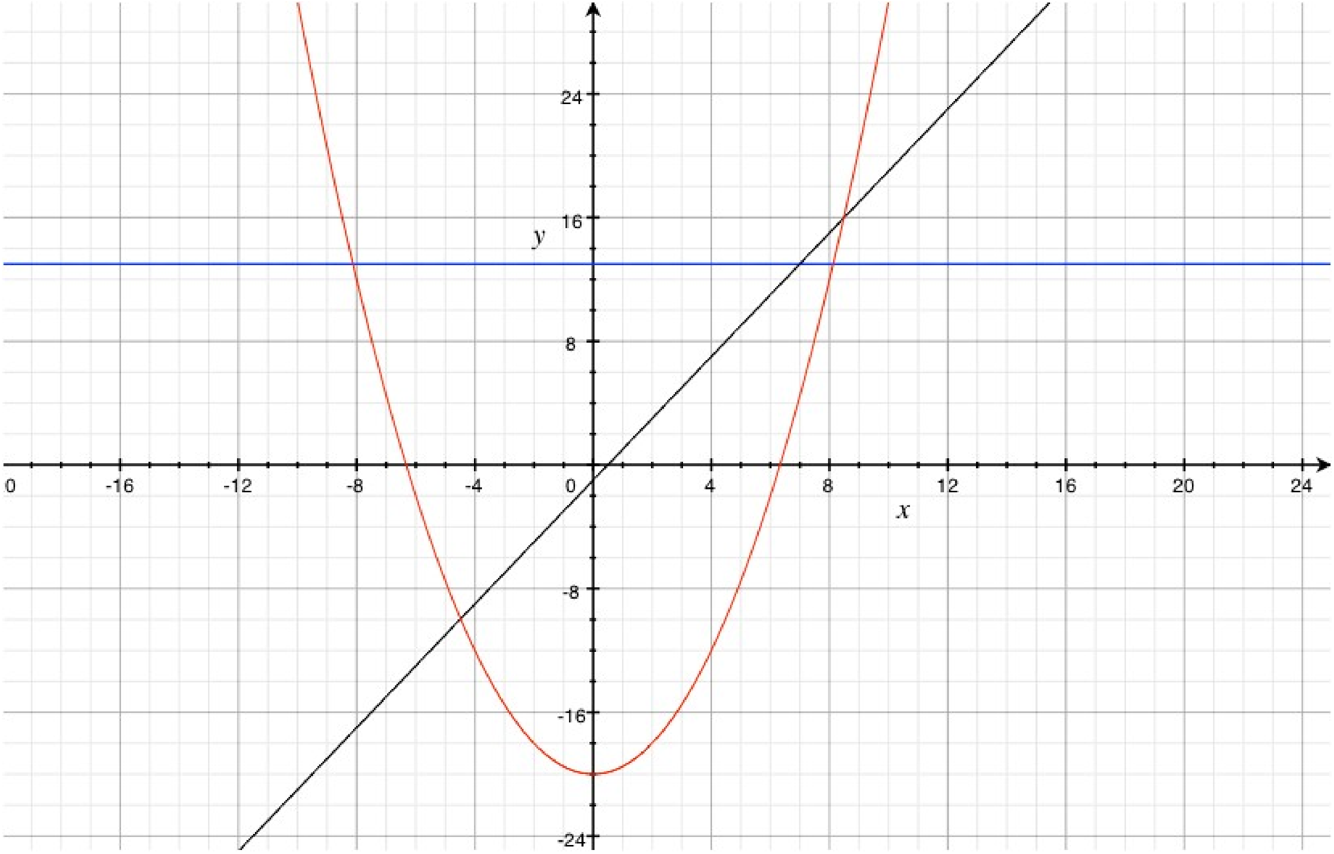
Part 1: Operations

A) determine the innermost loop and count the number of times the statement inside will execute. Why do we care about the statement in the innermost loop? Because that is the one that is executed the most times in the program.

B) order of magnitude corresponds with the rate of growth of the function. The fastest growing part of a function determines it's order of magnitude. For example, for the function f = 200N, at large values of N the magnitude of the value of F is determined by N, not by the factor of 200. Similarly, for f = 3N – 5, at large values of N the magnitude of the value of f depends on N, but the -5 in the equation doesn't change the magnitude of the value significantly.

C) which functions grow faster? Here I have plotted a constant function, a linear, function, and a quadratic function:



Part 2: Sorting

A) general tips: use the exact algorithms of the scripts provided in the homework. Write out an iteration of innermost and outermost loops of each algorithm to see their effect. Understand the method the algorithm uses to sort the list.

B) here you are simply counting the number of times you had to execute the inner loop on the given iteration of the outloop. For example, for outerloop #2, count how many times the inner loop is executed from the line “set j to i” to the line “replace item index of min...”. In this case j = 2...

C) again, use the exact algorithm of the given script. Don't forget to do this for all 3 given lists.

D) imagine that the numbers are TIED TO the names, so if a number moves, the name moves. Forget about what you know about high score lists.

GENERAL EXPLANATION OF IN-PLACE SELECTION SORT
the inner loop find the smallest item in the right hand side of the list
the outerloop determines where the right hand side of the list begins, and an item on the left with the smallest item from the right.

GENERAL EXPLANATION OF INSERTION SORT
The inner loop swaps two numbers in the list. It is executed only if the left item is NOT smaller than the right item, i.e., the items were out of order.
The outer loop determines where you start comparing pairs of list items.