1. a. F is O of n, G is O is 1

B. Int h(int n)

{

return n;

}

2. G is O ½n^2
3.
4. a. n%2 is 0 is even 1 is odd, big o is first step

B. Big o is n, go through the list comparing the numbers until the number is found or the next item in the list is null

C. Big o is n, I is set to the first number in the list, as we go through the list comparing each new number to I, if the number is smaller, I = the new number.

D. Big o is NlogN, each list is a linked list with a head pointer and the next pointers are initialized to nullptr, start at the head of list 1, go through list 2 and compare, when the item is found remove the items from both lists then set the comparisons at the two new heads and repeat, one of the lists next is a nullptr, with the first list indicating that everything is complete, and second meaning the lists are not equal

E. Big o is N, start at the head of both lists, I is 0 if the item at position a[i]!=b[i] then they're not equal, if I==n they're the same

F. Big o is logn if the number is larger go to the right, if it's smaller go to the left, when you find it return true, if you reach a nullptry return false

5. Make an array where each letter is a different position, each time a letter arises in the first word, it increases the value at that position by 1, each time it arises in the second it decreases the value by 1. If all the values are 0 at the end, it isn't an anagram.