**worst-case cost for Quicksort's partition step?** Θ(*n*)**the time complexities the same for Quicksort?** Best and Average only Θ(*nlog(n)*)**running time of Quicksort when the input is an array where all record values are equal?** Θ(*n*​2​​)**Quicksort a good choice**to sort many records **Mergesort a good choice** fast algorithm with a good worst case cost Θ(*nlog(n)*) for all cases. **f pairs of records among**n*n***records is**n(n-1)/2*n*(*n*−1)/2 **Selection Sort is not stable, but with minor modifications it could be made so. Selection Sort** Θ(*n*​2​​).. no situation where Bubble Sort is the best choice**Bubble** Θ(*n*​2​​) **number of comparisons for Insertion Sort** *n*​2​​/2..is stable**Insertion worst-average** Θ(*n*​2​​)**best** Θ(*n*​​​) **Insertion Sort a good choice** The array contains only a few records**Insertion is stable ||||n an array-based list, the successive elements in the list:** Must occupy contiguous space in memoryARRAY Based List. Θ(*n*​​​) **The lower bound for a problem is defined as the cost of the best algorithm that we know.**False **The upper bound for a problem is defined as the upper bound cost for the best algorithm that we know.**True **The lower bound of the sorting problem is**O(n log n)*O*(*n*log*n*)**because we can prove that this is the best cost that any sorting algorithm could reach.**True

sum = 0;

for (i = 0; i < n; i++) {

for (j = 0; A[j] != i; j++) Θ(*n*​2​​)

sum++;

}

**Suppose that a particular algorithm has time complexity**T(n) = 8n*T*(*n*)=8*n***and that executing an implementation of it on a particular machine takes**t*t***seconds for**n*n***inputs. Now suppose that we are presented with a machine that is 64 times as fast. How many inputs could we process on the new machine in**t*t***seconds?** 64*n* **Hardware vendor XYZ Corp. claims that their latest computer will run 100 times faster than that of their competitor, Prunes, Inc. If the Prunes, Inc. computer can execute a program on input of size**n*n***in one hour, what size input can XYZ's computer execute in one hour for an algorithm whose growth rate is**n^2​​**?** 10*n*

**Suppose that a particular algorithm has time complexity**T(n) = 3 \times 2^n*T*(*n*)=3×2​*n*​​**and that executing an implementation of it on a particular machine takes**t*t***seconds for**n*n***inputs. Now suppose that we are presented with a machine that is 64 times as fast. How many inputs could we process on the new machine in**t*t***seconds?** *n*+6 **An algorithm is a series of steps that act as a recipe to solve a particular problem.**

**For integers**a*a***and**b, a \equiv b*b*,*a*≡*b***if and only if**a + b*a*+*b***is even**

**A problem instance is a specific selection of values for the problem input.**

**isAncestorOf on the set of people**

**isFatherOf on the set of people**False

**isOlderThan on the set of people**

**Which of these is more a concern for Software Engineering than for a data structures course?** To design an algorithm that is easy to understand, code, and debug