**UNIT 1**

Note: When the exercise asks you to “design an algorithm for…,” it always means that “designs an EFFICIENT algorithm for … and ANALYZES your algorithm and write pseudo code”. You should keep this in mind when writing solutions.

1. Prove that each of the following sorting algorithms is stable or show that it is unstable by giving a counter example; moreover, determine whether it is in place: *bubble sort,* *insertion sort*, *quick sort*, *heap-sort, selection sort*.
2. Design a data structure to represent a set with elements being positive integers, and then design algorithms for the following operations:

a. Compute the union of two sets.

b. Compute the intersection of two sets.

c. Determine if a given element is in a given set.

1. Given two **sorted** arrays *x*[1]…*x*[*m*], *y*[1]…*y*[*n*], design an algorithm to compute min *i*, *j* | *x*[*i*]− *y*[*j*]|.
2. The input is a sequence of *n* integers with many duplications, such that the number of distinct integers in the sequence is *O*(*logn*).
   1. Design a sorting algorithm to sort such sequences using at most *O*(*nloglogn*) comparisons in the worst case.
   2. Why is the lower bound of sorting (*nlogn*) not satisfied in this case?
3. **Exercises 2.2-2**

Consider sorting *n* numbers stored in array *A*[1 : *n*] by first finding the smallest element of *A*[1 : *n*] and exchanging it with the element in *A*[1]. Then find the smallest element of *A*[2 : *n*], and exchange it with *A*[2]. Then find the smallest element of *A*[3 : *n*], and exchange it with *A*[3]. Continue in this manner for the first *n* - 1 elements of A. Write pseudocode for this algorithm, which is known as selection sort. What loop invariant does this algorithm maintain? Why does it need to run for only the first *n* - 1 elements, rather than for all n elements? Give theworst-case running time of selection sort in ‚*Θ-notation*. Is the best-case running time any better?