Algorithm HW1

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*, *merge sort*, *heap-sort*.

2. Design a data structure to represent a set with elements being positive integers, and then design algorithms for the following operations:
   Compute the union of two sets.
   Compute the intersection of two sets.
   Determine if a given element is in a given set.

3. Given two **sorted** arrays *x[1]…x[m], y[1]…y[n]*, design an algorithm to compute $\min_{i,j} |x[i] - y[j]|$.

4. Solve the recurrence $T(n) = 2T(n/2) + n - 1$ where $n = 2^k$ is assumed. T(1) = 0

5. Given a set *S* of *n* integers, and another number *M*, we want to determine whether or not there exist 2 numbers in *S* whose sum is exactly *M*. The algorithm of testing all possible 2 numbers in *S* will take $O(n^2)$ time and it is unacceptable.
   a) Design a more efficient algorithm for solving this problem. Analyze the time complexity of your algorithm.
   b) Extend your algorithm for the following case: determine whether or not there exist 3 numbers in *S* whose sum is exactly *M*.
   c) What about the case: determine whether or not there exist *k* (> 3) numbers in *S* whose sum is exactly *M*.

6. How to merge *k* sorted lists with total length *N* efficiently. What is the execution time of your algorithm.

7. The input is a sequence of *n* integers with many duplications, such that the number of distinct integers in the sequence is O(log*n*).
   (a) Design a sorting algorithm to sort such sequences using at most O(*n*loglog*n*) comparisons in the worst case.
   (b) Why is the lower bound of sorting $\Omega(n\log n)$ not satisfied in this case?