CS3230
Tutorial 4

1. Show step by step how merge sort will sort the following array:

   5 6 3 2 7

   Ans:

   $Mergesort(A, 1, 5)$
   1. $Mergesort(A, 1, 2)$
      1.1. $Mergesort(A, 1, 1)$: no changes
      1.2. $Mergesort(A, 2, 2)$: no changes
      1.3. $Merge(A, 1, 1, 2)$: no changes
   2. $Mergesort(A, 3, 5)$
      2.1 $Mergesort(A, 3, 3)$: no changes
      2.2. $Mergesort(A, 4, 5)$
         2.2.1 $Mergesort(A, 4, 4)$: no changes
         2.2.2 $Mergesort(A, 5, 5)$: no changes
         2.2.3 $Merge(A, 4, 4, 5)$: no changes
      2.3. $Merge(A, 3, 3, 5)$: Will change $A[3..5]$ to $2, 3, 7$
   3. $Merge(A, 1, 2, 5)$: Will change $A[1..5]$ to $2, 3, 5, 6, 7$
      Merge(A,1,2,5)

2. Show step by step how quick sort will sort the following array:

   5 6 3 2 7

   (for this, you should use the first algorithm done in class, that is when pivot is the first element)

   Ans: Initial pivot is 5.

   6 is left as is: $5, 6, 3, 2, 7$

   3 will then be swapped with 6: $5, 3, 6, 2, 7$

   2 will then be swapped with 6: $5, 3, 2, 6, 7$

   7 will be left as is: $5, 3, 2, 6, 7$

   5 will be swapped with 2 : $2, 3, 5, 6, 7$

   After this, there will be no more changes

3. Suppose A is a sorted array of n **distinct** integers. Give an algorithm to find if there exists an i such that $A[i] = i$ (and in case such an i exists, then return one such i).

   For example, if the array contains $-4, -2, 2, 4, 7$, then $A[4] = 4$.

   One can find such an i by just checking whether $A[j] = j$, for each j between 1 and n (both inclusive). However this is $O(n)$ algorithm.

   Can you give a better algorithm? Give time complexity (worst case) analysis of your algorithm.

   Ans: As the values in the array are integers and monotonically increasing, we immediately have that $A[i]-i$ is monotonically non-decreasing. Thus, if $A[i]-i > 0$, then $A[j]-j > 0$, for $j > i$. Similarly, if $A[i] - i < 0$, then $A[j] - j < 0$, for $j < i$.

   The following algorihtm solves the question.

   Find-id(A, i, j)
   (* Initially the above algorithm is called with $i = 1$ and $j = n$; Iteratively, we know that only r such that $i \leq r \leq j$ may satisfy the property that $A[r] = r$. *).
   1. If $i = j$, and $A[i] = i$, then return i.
   2. If ($i = j$ and $A[i] \neq i$) or $i > j$), then return NONE.
   3. Otherwise, let $m = \lfloor \frac{j+i}{2} \rfloor$.
      If $A[m] = m$, then return m.
      Else If $A[m] > m$, then return Find-id$(A, i, m-1)$.
      Else if $A[m] < m$, then return Find-id$(A, m+1, j)$.

   Complexity:

   $T(n) = T(n/2) + C$.

   Solving the recurrence gives $T(n) = O(\log n)$.

4. Give an algorithm which finds the distance between closest pair of points, when all the points lie on a straight line (though not necessarily all on the x axis). Try to make the algorithm simpler than the one done in class.

   Ans:

   The following does it for 2D. The algorithm can be easily generalised to higher dimensions. (A) Sort the points based on x-coordinate (as long as the points are not on a line perpendicular to x axis).

   (B) Find distances between adjacent points in the above sorted order, and report the shortest distance among these.

   In case the points are on a line perpendicular to x axis, then sort them based on y axis, and do (B) above.

5. Suppose we are given open intervals (a_i, b_i), $1 \leq i \leq n$, where $a_i < b_i$ are real numbers.

   Give an algorithm to determine a point, if any, where the maximum number of intervals overlap.

Ans: First sort all the a_i's and b_i's, where if $a_i = b_j$, then treat $b_j < a_i$ (that is think of adding a very very small quantity to each a_i in the sorting).

Use a counter D. Traverse the sorted list from smallest to largest, where each time a_i is encountered D is incremented, and each time b_j is encountered the counter is decremented.

The highest value of D occurs between some a_i and the next b_j. Any point between (a_i, b_j) is a point which occurs in maximum number of intervals.

6. Multiply the following two numbers using the method done in class.

   (a) 9437, 5428

   (b) 3642, 7529

   Here, you may multiply the numbers directly when they become two digit numbers.

   Ans: (a) $z_2 = 94 * 54 = 5076$

   $z_0 = 37 * 28 = 1036$

   $z_1 = (94 + 37) * (54 + 28) - (5076 + 1036) = 131 * 82 - 6112 == 10742 - 6112 = 4630$

   $50760000 + 463000 + 1036 = 51224036$

   (b) can be done similarly.