Data Structures Assignment 3

1- **Implement Bubble Sort, Selection Sort and Merge Sort**
2- **Implement & illustrate & Explain Quick Sort (Self Study)**
3- **Implement the Binary Search iteratively & recursively**
4- **Edit the Linked List in Day 1 to be a sorted Linked List**

(use built-in method Sort):

1- Sort the given matrix

Given an array arr[] of integers, determine the Next Greater Element Given a m x n matrix. The problem is to sort the given matrix in strict order. Here strict order means that the matrix is sorted in a way such that all elements in a row are sorted in increasing order and for row 'i', where 1 <= i <= m-1, the first element is greater than or equal to the last element of row 'i-1'.

Examples:
Input : mat[][] = { {5, 4, 7},
{1, 3, 8},
{2, 9, 6} }
Output :
1 2 3
4 5 6
7 8 9
Input: mat[][] = { {5, 4, 7},
{1, 3, 8} }
Output:
1 3 4
5 7 8

2- Sort elements by frequency

Given an array of integers arr[], sort the array according to the frequency of elements, i.e. elements that have higher frequency comes first. If the frequencies of two elements are the same, then the smaller number comes first.

Input: arr[] = [5, 5, 4, 6, 4]
Output: [4, 4, 5, 5, 6]
Explanation: The highest frequency here is 2. Both 5 and 4 have that frequency. Now since the frequencies are the same the smaller element comes first. So 4 comes first then comes 5. Finally comes 6. The output is 4 4 5 5 6.

Input: arr[] = [9, 9, 9, 2, 5]
Output: [9, 9, 9, 2, 5]
Explanation: The highest frequency here is 3. Element 9 has the highest frequency So 9 comes first. Now both 2 and 5 have the same frequency. So we print smaller elements first. The output is 9 9 9 2 5.