Week 3:

I. Given an unsorted array of integers, design an algorithm and a program to sort the array using insertion sort. Your program should be able to find number of comparisons and shifts (shifts total number of times the array elements are shifted from their place) required for sorting the array.

Input Format:

The first line contains number of test cases, T.

For each test case, there will be two input lines.

First line contains n (the size of array).

Second line contains space-separated integers describing array.

Output Format:

The output will have T number of lines.

For each test case T, there will be three output lines.

First line will give the sorted array.

Second line will give total number of comparisons.

Third line will give total number of shift operations required.

Sample I/O Problem I:

Input:	**Output:**
3 8 -23 65 -31 76 46 89 45 32 10 54 65 34 76 78 97 46 32 51 21 15 63 42 223 645 652 31 324 22 553 -12 54 65 86 46 325	-31 -23 32 45 46 65 76 89 comparisons = 13 shifts = 20 21 32 34 46 51 54 65 76 78 97 comparisons = 28 shifts = 37 -12 22 31 42 46 54 63 65 86 223 324 325 553 645 652 comparisons = 54 shifts = 68

II. Given an unsorted array of integers, design an algorithm and implement a program to sort this array using selection sort. Your program should also find number of comparisons and number of swaps required.

Input Format:

The first line contains number of test cases, T.

For each test case, there will be two input lines.

First line contains n (the size of array).

Second line contains space-separated integers describing array.

Output Format:

The output will have T number of lines.

For each test case T, there will be three output lines.

First line will give the sorted array.

Second line will give total number of comparisons.

Third line will give total number of swaps required.

Sample I/O Problem II:

Input:	Output:
3 8 -13 65 -21 76 46 89 45 12 10 54 65 34 76 78 97 46 32 51 21 15 63 42 223 645 652 31 324 22 553 12 54 65 86 46 325	-21 -13 12 45 46 65 76 89 comparisons = 28 swaps = 7 21 32 34 46 51 54 65 76 78 97 comparisons = 45 swaps = 9 12 22 31 42 46 54 63 65 86 223 324 325 553 645 652 comparisons = 105 swaps = 14

III. Given an unsorted array of positive integers, design an algorithm and implement it using a program to find whether there are any duplicate elements in the array or not. (use sorting) (Time Complexity = O(n log n))

Input Format:

The first line contains number of test cases, T.

For each test case, there will be two input lines.

First line contains n (the size of array).

Second line contains space-separated integers describing array.

Output Format:

The output will have T number of lines.

For each test case, output will be **'YES'** if duplicates are present otherwise '**NO**'.

Sample I/O Problem III:

Input:	Output:
3 5 28 52 83 14 75 10 75 65 1 65 2 6 86 2 75 8 15 75 35 86 57 98 23 73 1 64 8 11 90 61 19 20	NO YES NO