

Code : 1

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.util.Scanner;
```

```
public class A_BubbleSort {
```

```
    public static void bubbleSort(int[] arr) {
        int n = arr.length;
        for (int itr = 1; itr < n; itr++) {
            for (int j = 0; j < n - itr; j++) {
                if (isSmaller(arr, j + 1, j) == true) {
                    swap(arr, j + 1, j);
                }
            }
        }
    }
}
```

```
// used for swapping ith and jth elements of array
```

```
public static void swap(int[] arr, int i, int j) {
    System.out.println("Swapping " + arr[i] + " and " + arr[j]);
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}
```

```
// return true if ith element is smaller than jth element
```

```
public static boolean isSmaller(int[] arr, int i, int j) {
    System.out.println("Comparing " + arr[i] + " and " + arr[j]);
    if (arr[i] < arr[j]) {
        return true;
    } else {
        return false;
    }
}
```

```
public static void print(int[] arr) {
    for (int i = 0; i < arr.length; i++) {
        System.out.println(arr[i]);
    }
}
```

```
public static void main(String[] args) throws Exception {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    bubbleSort(arr);
    print(arr);
}
```

```
}
```

Code : 2

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.util.Scanner;
```

```
public class B_SelectionSort {
```

```
    public static void selectionSort(int[] arr) {
        int n = arr.length;
        for (int i = 0; i < n - 1; i++) {
            int minidx = i;
            for (int j = i + 1; j < n; j++) {
                if (isSmaller(arr, j, minidx)) {
                    minidx = j;
                }
            }
            swap(arr, i, minidx);
        }
    }
```

```
// used for swapping ith and jth elements of array
```

```
public static void swap(int[] arr, int i, int j) {
    System.out.println("Swapping " + arr[i] + " and " + arr[j]);
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}
```

```
// return true if ith element is smaller than jth element
```

```
public static boolean isSmaller(int[] arr, int i, int j) {
    System.out.println("Comparing " + arr[i] + " and " + arr[j]);
    if (arr[i] < arr[j]) {
        return true;
    } else {
        return false;
    }
}
```

```
public static void print(int[] arr) {
    for (int i = 0; i < arr.length; i++) {
        System.out.println(arr[i]);
    }
}
```

```
public static void main(String[] args) throws Exception {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    selectionSort(arr);
    print(arr);
}
```

```
}
```



Code : 3

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class C_InsertionSort {
```

```
    public static void insertionSort(int[] arr) {  
        for (int i = 1; i < arr.length; i++) {  
            for (int j = i - 1; j >= 0; j--) {  
                if (isGreater(arr, j, j + 1)) {  
                    swap(arr, j, j + 1);  
                } else {  
                    break;  
                }  
            }  
        }  
    }  
}
```

```
// used for swapping ith and jth elements of array
```

```
public static void swap(int[] arr, int i, int j) {  
    System.out.println("Swapping " + arr[i] + " and " + arr[j]);  
    int temp = arr[i];  
    arr[i] = arr[j];  
    arr[j] = temp;  
}
```

```
// return true if jth element is greater than ith element
```

```
public static boolean isGreater(int[] arr, int j, int i) {  
    System.out.println("Comparing " + arr[i] + " and " + arr[j]);  
    if (arr[i] < arr[j]) {  
        return true;  
    } else {  
        return false;  
    }  
}
```

```
public static void print(int[] arr) {  
    for (int i = 0; i < arr.length; i++) {  
        System.out.println(arr[i]);  
    }  
}
```

```
public static void main(String[] args) throws Exception {  
    Scanner scn = new Scanner(System.in);  
    int n = scn.nextInt();  
    int[] arr = new int[n];  
    for (int i = 0; i < n; i++) {  
        arr[i] = scn.nextInt();  
    }  
    insertionSort(arr);  
    print(arr);  
}
```

```
}
```



Code : 4

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class D_MergeTwoSortedArray {

    public static int[] mergeTwoSortedArrays(int[] a, int[] b) {
        // write your code here
        int alen = a.length;
        int blen = b.length;

        int[] res = new int[alen + blen];

        int i = 0;
        int j = 0;
        int k = 0;

        // when there are elements in both the array
        while (i < alen && j < blen) {
            if (a[i] < b[j]) {
                res[k] = a[i];
                i++;
            } else {
                res[k] = b[j];
                j++;
            }
            k++;
        }

        while (i < alen) {
            res[k] = a[i];
            i++;
            k++;
        }

        while (j < blen) {
            res[k] = b[j];
            j++;
            k++;
        }

        return res;
    }

    public static void print(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }
    }

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
```

```
int[] a = new int[n];
for (int i = 0; i < n; i++) {
    a[i] = scn.nextInt();
}
int m = scn.nextInt();
int[] b = new int[m];
for (int i = 0; i < m; i++) {
    b[i] = scn.nextInt();
}
int[] mergedArray = mergeTwoSortedArrays(a, b);
print(mergedArray);
}

}
```

Code : 5

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class E_MergeSort {
```

```
    public static int[] mergeSort(int[] arr, int lo, int hi) { //1
        if (lo == hi) { //2
            int[] ba = new int[1];
            ba[0] = arr[lo];
            return ba;
        }
        int mid = (lo + hi) / 2; //3
        int[] f = mergeSort(arr, lo, mid); //4
        int[] s = mergeSort(arr, mid + 1, hi); //5
        int[] fin = mergeTwoSortedArrays(f, s); //6
        return fin; //7
    }
```

```
//used for merging two sorted arrays
```

```
public static int[] mergeTwoSortedArrays(int[] a, int[] b) {
    System.out.println("Merging these two arrays ");
    System.out.print("left array -> ");
    print(a);
    System.out.print("right array -> ");
    print(b);
    int i = 0, j = 0, k = 0;
    int[] ans = new int[a.length + b.length];
    while (i < a.length && j < b.length) {
        if (a[i] <= b[j]) {
            ans[k] = a[i];
            i++;
            k++;
        } else {
            ans[k] = b[j];
            j++;
            k++;
        }
    }

    while (i < a.length) {
        ans[k] = a[i];
        k++;
        i++;
    }

    while (j < b.length) {
        ans[k] = b[j];
        k++;
        j++;
    }

    return ans;
}
```



```
public static void print(int[] arr) {  
    for (int i = 0; i < arr.length; i++) {  
        System.out.print(arr[i] + " ");  
    }  
    System.out.println();  
}
```

```
public static void main(String[] args) throws Exception {  
    Scanner scn = new Scanner(System.in);  
    int n = scn.nextInt();  
    int[] arr = new int[n];  
    for (int i = 0; i < n; i++) {  
        arr[i] = scn.nextInt();  
    }  
    int[] sa = mergeSort(arr, 0, arr.length - 1);  
    System.out.print("Sorted Array -> ");  
    print(sa);  
}
```

```
}
```

Code : 6

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class F_PartitionAnArray {
```

```
    public static void partition(int[] arr, int pivot) {  
        int i = 0;  
        int j = 0;  
        while (i < arr.length) {  
            if (arr[i] > pivot) {  
                i++;  
            } else if (arr[i] <= pivot) {  
                swap(arr, i, j);  
                i++;  
                j++;  
            }  
        }  
    }
```

```
}
```

```
// used for swapping ith and jth elements of array
```

```
public static void swap(int[] arr, int i, int j) {  
    System.out.println("Swapping " + arr[i] + " and " + arr[j]);  
    int temp = arr[i];  
    arr[i] = arr[j];  
    arr[j] = temp;  
}
```

```
public static void print(int[] arr) {  
    for (int i = 0; i < arr.length; i++) {  
        System.out.print(arr[i] + " ");  
    }  
    System.out.println();  
}
```

```
public static void main(String[] args) throws Exception {
```

```
    Scanner scn = new Scanner(System.in);  
    int n = scn.nextInt();  
    int[] arr = new int[n];  
    for (int i = 0; i < n; i++) {  
        arr[i] = scn.nextInt();  
    }  
    int pivot = scn.nextInt();  
    partition(arr, pivot);  
    print(arr);
```

```
}
```

```
}
```

Code : 7

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class G_QuickSort {
```

```
    public static void quickSort(int[] arr, int lo, int hi) {
```

```
        if (lo > hi) { // lo>=hi will work same
            return;
        }
```

```
        int pivot = arr[hi];
        int pi = partition(arr, pivot, lo, hi);
        quickSort(arr, lo, pi - 1);
        quickSort(arr, pi + 1, hi);
```

```
    }
```

```
    public static int partition(int[] arr, int pivot, int lo, int hi) {
```

```
        System.out.println("pivot -> " + pivot);
        int i = lo, j = hi;
        while (i <= hi) {
            if (arr[i] <= pivot) {
                swap(arr, i, j);
                i++;
                j++;
            } else {
                i++;
            }
        }
        System.out.println("pivot index -> " + (j - 1));
        return (j - 1);
    }
```

```
    // used for swapping ith and jth elements of array
```

```
    public static void swap(int[] arr, int i, int j) {
        System.out.println("Swapping " + arr[i] + " and " + arr[j]);
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
    }
```

```
    public static void print(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }
```

```
    public static void main(String[] args) throws Exception {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
```

```
int[] arr = new int[n];
for (int i = 0; i < n; i++) {
    arr[i] = scn.nextInt();
}
quickSort(arr, 0, arr.length - 1);
print(arr);
}

}
```

Code : 8

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class H_QuickSelect {
```

```
    public static int quickSelect(int[] arr, int lo, int hi, int k) {
        int pivot = arr[hi];
        int pidx = partition(arr, pivot, lo, hi);
        if (k == pidx) {
            return pivot;
        } else if (k > pidx) {
            return quickSelect(arr, pidx + 1, hi, k);
        } else {
            return quickSelect(arr, lo, pidx - 1, k);
        }
    }
```

```
    public static int partition(int[] arr, int pivot, int lo, int hi) {
        System.out.println("pivot -> " + pivot);
        int i = lo, j = hi;
        while (i <= hi) {
            if (arr[i] <= pivot) {
                swap(arr, i, j);
                i++;
                j++;
            } else {
                i++;
            }
        }
        System.out.println("pivot index -> " + (j - 1));
        return (j - 1);
        // j is first among (larger than pivot) numbers
        // pivot lives at j - 1, and it is at it's correct sorted position
    }
```

```
    // used for swapping ith and jth elements of array
```

```
    public static void swap(int[] arr, int i, int j) {
        System.out.println("Swapping " + arr[i] + " and " + arr[j]);
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
    }
```

```
    public static void print(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }
```

```
    public static void main(String[] args) throws Exception {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
```

```
int[] arr = new int[n];
for (int i = 0; i < n; i++) {
    arr[i] = scn.nextInt();
}
int k = scn.nextInt();
System.out.println(quickSelect(arr, 0, arr.length - 1, k - 1));
}

}
```

Code : 9

```
package Topic_12_TimeAndSpaceComplexity;

import java.io.*;

import java.util.*;

public class I_CountSort {

    public static void countSort(int[] arr, int min, int max) {
        int range = max - min + 1;
        int[] ans = new int[arr.length];
        //make frequency arr
        int[] farr = new int[range];
        for (int i = 0; i < arr.length; i++) {
            farr[arr[i] - min]++;
        }
        //convert it into prefix sum array
        for (int i = 1; i < farr.length; i++) {
            farr[i] += farr[i - 1];
        }
        //stable sorting(filling ans array)
        for (int i = arr.length - 1; i >= 0; i--) {
            int pos = farr[arr[i] - min] - 1;
            ans[pos] = arr[i];
            farr[arr[i] - min]--;
        }
        //filling original array with the help of ans array
        for (int i = 0; i < arr.length; i++) {
            arr[i] = ans[i];
        }
    }

    public static void print(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }
    }

    public static void main(String[] args) throws Exception {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[] arr = new int[n];
        int max = Integer.MIN_VALUE;
        int min = Integer.MAX_VALUE;
        for (int i = 0; i < n; i++) {
            arr[i] = scn.nextInt();
            max = Math.max(max, arr[i]);
            min = Math.min(min, arr[i]);
        }
        countSort(arr, min, max);
        print(arr);
    }
}
```

Code : 10

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class J_RadixSort {
```

```
    public static void radixSort(int[] arr) {  
        int max = Integer.MIN_VALUE;  
        for (int i = 0; i < arr.length; i++) {  
            max = Math.max(max, arr[i]);  
        }  
        //call countSort for every digit from right to left  
        for (int exp = 1; max / exp >= 1; exp *= 10)  
            countSort(arr, exp);  
    }
```

```
    public static void countSort(int[] arr, int exp) {
```

```
        int[] ans = new int[arr.length];  
        // make frequency arr  
        int[] farr = new int[10];  
        for (int i = 0; i < arr.length; i++) {  
            farr[(arr[i] / exp) % 10]++;  
        }  
        // convert it into prefix sum array  
        for (int i = 1; i < farr.length; i++) {  
            farr[i] += farr[i - 1];  
        }  
        // stable sorting(filling ans array)  
        for (int i = arr.length - 1; i >= 0; i--) {  
            int pos = farr[(arr[i] / exp) % 10] - 1;  
            ans[pos] = arr[i];  
            farr[(arr[i] / exp) % 10]--;  
        }  
        // filling original array with the help of ans array  
        for (int i = 0; i < arr.length; i++) {  
            arr[i] = ans[i];  
        }  
        System.out.print("After sorting on " + exp + " place -> ");  
        print(arr);  
    }
```

```
    public static void print(int[] arr) {  
        for (int i = 0; i < arr.length; i++) {  
            System.out.print(arr[i] + " ");  
        }  
        System.out.println();  
    }
```

```
    public static void main(String[] args) throws Exception {  
        Scanner scn = new Scanner(System.in);  
        int n = scn.nextInt();  
        int[] arr = new int[n];
```



```
        for (int i = 0; i < n; i++) {  
            arr[i] = scn.nextInt();  
        }  
        radixSort(arr);  
        print(arr);  
    }  
}
```

Code : 11

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.util.*;
```

```
public class K_SortDates {
```

```
    public static void sortDates(String[] arr) {  
        countSort(arr, 1000000, 100, 32);  
        countSort(arr, 10000, 100, 13);  
        countSort(arr, 1, 10000, 2501);  
    }
```

```
    public static void countSort(String[] arr, int div, int mod, int range) {  
        String[] ans = new String[arr.length];  
        // make frequency arr  
        int[] farr = new int[range];  
        for (int i = 0; i < arr.length; i++) {  
            farr[Integer.parseInt(arr[i], 10) / div % mod]++;  
        }  
        // convert it into prefix sum array  
        for (int i = 1; i < farr.length; i++) {  
            farr[i] += farr[i - 1];  
        }  
        // stable sorting(filling ans array)  
        for (int i = arr.length - 1; i >= 0; i--) {  
            int pos = farr[Integer.parseInt(arr[i], 10) / div % mod] - 1;  
            ans[pos] = arr[i];  
            farr[Integer.parseInt(arr[i], 10) / div % mod]--;  
        }  
        // filling original array with the help of ans array  
        for (int i = 0; i < arr.length; i++) {  
            arr[i] = ans[i];  
        }  
    }
```

```
    public static void print(String[] arr) {  
        for (int i = 0; i < arr.length; i++) {  
            System.out.println(arr[i]);  
        }  
    }
```

```
    public static void main(String[] args) throws Exception {  
        Scanner scn = new Scanner(System.in);  
        int n = scn.nextInt();  
        String[] arr = new String[n];  
        for (int i = 0; i < n; i++) {  
            String str = scn.next();  
            arr[i] = str;  
        }  
        sortDates(arr);  
        print(arr);  
    }
```

```
}
```



Code : 12

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class L_Sort01 {
```

```
    public static void sort01(int[] arr) {  
        // 0 to j-1 -> All Zeroes //  
        // j to i-1 -> All One's //  
        /* i to arr.length-1 -> All unknowns */
```

```
        int i = 0, j = 0;  
        while (i < arr.length) {  
            if (arr[i] == 0) {  
                swap(arr, i, j);  
                i++;  
                j++;  
            } else {  
                i++;  
            }  
        }  
    }
```

```
    // used for swapping ith and jth elements of array
```

```
    public static void swap(int[] arr, int i, int j) {  
        System.out.println("Swapping index " + i + " and index " + j);  
        int temp = arr[i];  
        arr[i] = arr[j];  
        arr[j] = temp;  
    }
```

```
    public static void print(int[] arr) {  
        for (int i = 0; i < arr.length; i++) {  
            System.out.println(arr[i]);  
        }  
    }
```

```
    public static void main(String[] args) throws Exception {  
        Scanner scn = new Scanner(System.in);  
        int n = scn.nextInt();  
        int[] arr = new int[n];  
        for (int i = 0; i < n; i++) {  
            arr[i] = scn.nextInt();  
        }  
        sort01(arr);  
        print(arr);  
    }
```

```
}
```

Code : 13

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.util.*;
```

```
public class M_Sort012 {
```

```
    public static void sort012(int[] arr) {  
        // 0 to j-1 -> All Zeroes  
        // j to i-1 -> All One's  
        // i to k - 1 -> All unknowns  
        // k to last -> All Two's
```

```
        int i = 0, j = 0, k = arr.length - 1;
```

```
        while (i <= k) {  
            if (arr[i] == 0) {  
                swap(arr, i, j);  
                i++;  
                j++;  
            } else if (arr[i] == 1) {  
                i++;  
            } else {  
                swap(arr, i, k);  
                k--;  
            }  
        }  
    }
```

```
}
```

```
// used for swapping ith and jth elements of array
```

```
public static void swap(int[] arr, int i, int j) {  
    System.out.println("Swapping index " + i + " and index " + j);  
    int temp = arr[i];  
    arr[i] = arr[j];  
    arr[j] = temp;  
}
```

```
public static void print(int[] arr) {  
    for (int i = 0; i < arr.length; i++) {  
        System.out.println(arr[i]);  
    }  
}
```

```
public static void main(String[] args) throws Exception {  
    Scanner scn = new Scanner(System.in);  
    int n = scn.nextInt();  
    int[] arr = new int[n];  
    for (int i = 0; i < n; i++) {  
        arr[i] = scn.nextInt();  
    }  
    sort012(arr);  
    print(arr);  
}
```

```
}
```

Code : 14

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class N_TargetSumPair1 {
```

```
    public static void targetSumPair(int[] arr, int target) {  
        Arrays.sort(arr);
```

```
        int left = 0, right = arr.length - 1;  
        while (left < right) {  
            if (arr[left] + arr[right] == target) {  
                System.out.println(arr[left] + ", " + arr[right]);  
                left++;  
                right--;  
            } else if (arr[left] + arr[right] > target) {  
                right--;  
            } else {  
                left++;  
            }  
        }  
    }
```

```
    public static void main(String[] args) throws Exception {  
        Scanner scn = new Scanner(System.in);  
        int n = scn.nextInt();  
        int[] arr = new int[n];  
        for (int i = 0; i < n; i++) {  
            arr[i] = scn.nextInt();  
        }  
        int target = scn.nextInt();  
        targetSumPair(arr, target);  
    }
```

```
}
```

Code : 15

```
package Topic_12_TimeAndSpaceComplexity;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class O_PivotOfSortedRotatedArray {
```

```
    public static int findPivot(int[] arr) {  
        int lo = 0, hi = arr.length - 1;  
        while (lo < hi) {  
            int mid = (lo + hi) / 2;  
            if (arr[mid] > arr[hi]) {  
                lo = mid + 1;  
            } else {  
                hi = mid;  
            }  
        }  
        return arr[lo];  
    }  
}
```

```
    public static void main(String[] args) throws Exception {  
        Scanner scn = new Scanner(System.in);  
        int n = scn.nextInt();  
        int[] arr = new int[n];  
        for (int i = 0; i < n; i++) {  
            arr[i] = scn.nextInt();  
        }  
        int pivot = findPivot(arr);  
        System.out.println(pivot);  
    }  
}
```

