→ DSA →

Assignment No. 4

PAGE NO.
DATE:

Que: 1. Answer :-

```
import java.util.ArrayList;
import java.util.ArrayList;

public class CommonElements {
    public static int[] findCommonElement
            (int[] arr1, int[] arr2, int
                                    int arr3)
{
    List<Integer> commonElement = new
                                ArrayList<>();
    int i = 0, j = 0, k = 0;

    while (i < arr1.length && j < arr2.length && k <
                                arr3.length) {

    if (arr1[i] == arr2[j] && arr2[j] == arr3[k]) {
    {
        commonElements.add (arr[i]);
        i++;
        j++;
        k++;
    } else if (arr1[i] < arr2[j]) {

        i++;
    } else if (arr2[j] < arr3[k]) {

        j++;
    } else {
        k++;

    }
}
```

```
int[] result = new int[commonElement.size()];
for (int index = 0; index < commonElement.size
                                    (); index++) {
    result[index] = commonElements.get(index);
}
return result;
}
public static void main (String[] args) {
    int[] arr1 = {1, 2, 3, 4, 5};

    int[] arr2 = {1, 2, 5, 7, 9};

    int[] arr3 = {1, 3, 4, 5, 8};

int[] commonElements = findCommonElements
                        (arr1, arr2, arr3);

System.out.println ("common elements :");

for (int element : commonElements) {

    System.out.print(element + " ");

}
}
```

Que 2. Answer :-

```
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

public class ArrayDifference {
public static List<List<Integer>> findArray
        Difference(int[] nums1, int
                [] nums2)
{
    List<List<Integer>> answer = new ArrayList
                                    <>();
    Set<Integer> set1 = new HashSet<>();
    Set<Integer> set2 = new HashSet<>();

    for (int num : nums1) {
        set1.add(num);
    }

    for (int num: nums2) {
        set.add(num);
    {
    List<Integer> diff1 = new ArrayList<>();
    List<Integer> diff2 = new ArrayList<>();

    for (int num : nums1) {
        if (!set2.contains(num)) {
            diff1.add(num);
        {
    }

    for (int num : num2) {
        if (!set1.contains(num)) {
            diff2.add(num);
        {
}
```

```
answer.add(diff1);
answer.add(diff2);
return answer;
}
public static void main(String[] aegr) {
    int[] nums1 = {1, 2, 3};
    int[] nums2 = {2, 4, 6};

List<List<Integer>> result = findArrayDifference
                    (nums1, nums2);
System.out.println(result);
}
}
```

Que 3 :- Answer :-

```
public static void main(String[] aegr) {
    int[][] matrix = {{1,2,3},{4,5,6},
                      {7, 8, 9}};
int[][] transposeMatrix = transpose(Matrix);

for(int i=0; i<transposeMatrix.length; i++) {
for(int j=0; i<transposeMatrix.length; j++) {

}
System.out.println();
{
{
```

1 2 3 → 1 4 7

4 5 6 → 2 5 8

7 8 9 → 3 6 9

Que: 4: Answer:

```
import java.util.Arrays;
public class ArrayPairSum {
public static int arrayPairsum(int[] nums) {
 Arrays.sort(nums);
 int sum = 0;
 for(int i = 0; i < nums.length; i += 2) {
    sum += nums[i];
 }
 return sum;
}
public static void main(String[] args) {
      int[] nums = {1, 4, 3, 2};
    int maxSum = arrayPairSum(nums);
 System.out.println("Maximated Sum:"
                          + maxSum);
 }
}
```

Que: 5 Answer:

```
public class Staircase {
public static void int countCompleteRows
                          (int n) {
 int rowCount = 0;
 int coins = 0;
 while (coins <= n) {
    rowCount++;
    coins += rowCount;
 }
 return rowCount - 1;
}
public static void main(String[] args) {
 int n = 5;
 int completeRows = countCompleteRows(n);
 System.out.println("Number of complete rows
                          :" + completeRows);
 }
}
```

Que: 6 :- Answer :-

```
import java.util.*;
public class SortedSquares {
 public static int[] sortedSquares(int[] nums)
{
 int[] result = new int[nums.length];
 for(int i = 0; i < nums.length; i++) {
    result[i] = nums[i] * nums[i];
 }
 Arrays.sort(result);
 return result;
}
public static void main(String[] args) {
 int[] nums = {-4, -1, 0, 3, 10};
 int[] square = sortedSquares(nums);
 System.out.println("Sorted Squared:" + Arrays.
                          toString(Squared));
 }
}
```

Que 7 :- Answer :-

```
public class MaxIntegers {
public static int maxCount(int m, int n, int[][] opr) {
{
    if (opr == null || opr.length == 0) {
        return m * n;
    }
    int minX = m;
    int minY = n;

    for (int[] op : opr) {
        minX = Math.min(minX, op[0]);
        minY = Math.min(minY, op[1]);
    }
    return minX * minY;
}

public static void main(String[] args) {
    int m = 3;
    int n = 3;
    int[][] opr = {{2,2},{3,3}};

int maxIntegers = maxCount(m, n, opr);
System.out.println("Number of maximum
                    integers: " + maxIntegers);
{
}
```

Output :

Number of maximum integers: 4

Que 8 :- Answer :-

```
public class ShuffleArray {
public static int[] shuffle(int[] nums, int n) {
    int[] result = new int[2 * n];

    int index = 0;

    for (int i = 0; i < n; i++) {

        result[index++] = nums[i];
        result[index++] = nums[n + i];

    }  result return result;
}

public static void main(String[] args) {
    int[] nums = {2, 5, 1, 3, 4, 7};

    int n = 3;
int[] shuffledArray = shuffle(nums, n);
System.out.println("Shuffled array:");

for (int i = 0; i < shuffledArray.length; i++)
{
    System.out.print(shuffledArray[i]);

    if (i != shuffledArray.length - 1) {

        System.out.println(",");
    }

}
System.out.println("]");

{
}
```