Assignment No 6

PAGE NO.
DATE:

Que: 1: Answer

```
import java.util.Arraylist;
import java.util.Arraylist;

public clarr PermutationReconstruction {
   public static int[] reconstructPermutation (strings){
    int n = s.length();
    int n perm = new int[n+1];

   for(int i=0; i<=n; i++) {
        perm[i]=i;
    }
   List< Integer> result = new Arraylist<>();
     for (int i=0; i<n; i++);
    {
     if (s.charAt(i)=='I') {
     }
        result.add(perm[i]);
    } else {
        result.add(perm[n-i]);
     }
    }
    result. add(perm [n]);

   for(int i=0; i<=n; i++) {
        perm [i] = result.get(i);
    }
     return perm;
  }
    public static void main (String[] args) {
        String     s = " IDID";
        int[] perm = reconstructPermutation
                                          (s);
        ~~for(int num : perm) {~~
            ~~system.out.~~ print(num+"  ");
       }
    }
```

Que 2:· Answer:-

```
public clarr Matrixsearch {
 public static boolean searchMatrix
                    (int[][] matrix, int target){
    int m = matrix. length;
    int n= matrix[0].length;

    int left = 0 ;
    int right = m * n - 1;

   while ( left <= right) {
       int mid = left + (right - left)/2;
       int midvalue = matrix[mid/n][mid%n];

     if (midValue == target) {
        return true;
     } else if(midvalue < target) {
          left= mid + 1;

     } else{

         right = mid - 1;
     }
  }  return false;
 }
  public static void main(string [] args) {

   int [][] matrix = {
          {1, 3, 5, 7},
          ~~{10, 11, 16, 20}~~,
          { 23, 30, 34, 60}
     };
            int target = 3;
       boolean found = searchMatrix (matrix,target);
     } system. out .println (found);
}
```

Quer: Answer :- 3.

PAGE NO

```
public class MountainArray {
  public static boolean validMountainArray
                                (int[] arr) {
    int n = arr.length;

    if (n<3) {
       return false;
    }
     int i=0;
     while (i<n-1 && arr[i] < arr[i+1]) {
         i++;
     }
       if (i==0 || i == n-1) {
          return false;
     }
     while (i<n-1 && arr[i] > arr[i+1]) {
        i++;
     }
     return i==n-1;
    }
    public static void main (String [] args) {
      int [] arr = {2, 1};
    boolean isValidMountain = ValidMountinArray
                                      (arr);
    system. out. println (isValidMountain);

    }
  }
```

Que: Answer:
4]

```
import java.util.HashMap;
import java.util.Map;
```

PAGE NO.
DATE:

```
public class ContiguousSubarray {
 public static int findMaxLength(int[] nums) {
    int maxLength = 0;
    int count = 0;
 Map<Integer, Integer> countMap = new
                                HashMap<>();
 countMap.put(0, -1);
  for (int i = 0; i < nums.length; i++) {
      if (nums[i] == 0) {
          count--;
     } else {
          count++;
     }
     if (countMap.containsKey(count)) {
      maxLength = Math.max(maxLength,
                     i - countMap.get(count));
     } else {
        countMap.put(count, i);
     }
 }
      return maxLength;
 }
  public static void main(String[] args) {
  int[] nums = {0, 1};
  int maxLength = findMaxLength(nums);
  System.out.println(maxLength);
 }
}
```

Que: 5 Answer:

PAGE NO.
DATE:

```
Import java.util.Arrays;
public class MinimumProductSum {
   pubulic static int minProductsum( Int [] nums1,
                                     int [] nums 2) {
      Arrays. sort(nums1);
      Arrays. sort (nums 2);

      int n = nums1. length;
      int sum = 0;

      for(int i=0; i<n; i++) {
         sum += nums1[i] * nums2[n-i-1];
      }
      return sum;
   }
   public static void main(String [] args) {
      int [] nums1 = {5, 3, 4, 2}
      int [] nums2 = {4, 2, 2, 5};

      int minProduct = minProductSum(nums1, nums2);
      System.out.println(minProduct);
   }
}
```

Que 6 Answer:

PAGE NO
DATE:

```
import java.util.Arrays;
import java.util.HashMap;
import java.util.Map;

public class DoubledArray {
 public static int[] constructOriginal(int[]
                                   changed) {
  Map<Integer, Integer> countMap
                 = new HashMap<>();
   int n = changed.length;

  for(int num : changed) {
      countMap.put(num, countMap.getOr
                        Default(num,0)
                              +1));
  }

  int [] original = new int [n/2];
  int index = 0;
  for(int num : changed) {
   if (countMap.containsKey(num*2) &&
             countMap.get(num*2) > 0)

     original[index] = num;

    countMap.put(num*2, countMap.get
                        (num*2) -1);
     index++;
    }
 }
      if (index == n/2) {

        return original;
      } else {
        return new int[0];
  }}
```

```
public static void main(String[] args){
  int[] changed = {1, 3, 4, 2, 6, 8};
 int[] original = constructOriginal(changed);
 if(original.length > 0){
   ~~System.o~~
   System.out.println(Arrays.toString(original));
 }else{
   System.out.println("No valid original array
                        exist.");
 }
}
```

output
:- [1, 3, 4]

}

Qus: 7 Answer:

```
 public class SpiralMatrix{
public static int[][] generateMatrix(int n){
 int[] matrix = new int[n][n];
 int num = 1;
 int rowStart = 0; rowEnd = n-1;
 int colStart = 0, colEnd = n-1;

while(num <= n * n){

 for(int j = colStart; j <= colEnd; j++){
    matrix[rowStart][j] = num++;
 }
  rowStart++;
 for(int i = rowStart; i <= rowEnd; i++){
    matrix[i][colEnd] = num++;
 }
 colEnd--;
```

```
for (int j = colEnd ; j>= colstart ; ~~i++~~ j--){
  matrix [rowEnd] [j] = num++
}
rowEnd --;
for (int i= rowEnd ; i> rowstart; ~~i++~~ i--)
{ .
  matrix [i] [colEnd] = num++;
}
colEnd -- ~~{~~;

for(int j = colEnd ; j >= colstart ; j--){
   matrix [rowEnd] [j] = num++;
}
rowEnd --;

for (int i= rowEnd; i >= rowstart ; i~~+~~--)
  {
     matrix [i] [colstart] num++;

}
  colstart++;
}
  return matrix ;
}
  public static void void main
              (String [] args) {
 int n = 3;
int [] [] matrix = generateMatrix(n);

for(int[] row: matrix) {

  for(int num : row) {

     System.out.print(num + " ");
  }
    System.out.println();
```

Que 8 : Answer :-

```
public class SparseMatrixMultiplication {
public static int[][] multiply (int[][] mat1,
                    int[][] mat2) {

  int m = mat1.length;
  int k = mat1[0].length;
  int n = mat2[0].length;

  int[][] result = new int[m][n];
  for (int i=0; i<m; i++) {
      for(int j=0; j<n; j++) {
          for(int p=0; p<k; p++) {
             result[i][j] += mat1[i][p] *
                       mat2[p][j];
      }
    }
  }
   return result;
}
 public static void main(String[] args) {

 int[][] mat1 = {{1,0,0},{-1,0,3}};

 int[][] mat2 = {{7,0,0},{0,0,0},{0,0,1}};

 int[][] result = multiply(mat1, mat2);

   for (int[] row: result) {
       for (int num: row) {
          System.out.print(num + " ");
       }
       System.out.println();
   }
 }
}
```