Answer 1

```
import java.util.*;

class Solution {

   public int numJewelsInStones(String jewels, String stones) {

      int result = 0;

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

         if (jewels.contains(String.valueOf(stones.charAt(i)))) {

            result++;

         }

      }

      return result;

   }



public static void main(String args[])

{

   Scanner sc=new Scanner(System.in);

   String s,j;

   int a=0;

   System.out.println("Stones =");

   s=sc.nextLine();

   System.out.println("Jewels =");

   j=sc.nextLine();

   Solution xx = new Solution();

   if((j.length())>= 1 && (s.length())<= 50 )
```

```
        {
          a=xx.numJewelsInStones(j,s);
          System.out.println(a);
        }
    else
           System.out.println("enter value in range");
}
}
```

Answer 2

```
import java.util.*;
class Solution {
    public String mergeAlternately(String word1, String word2) {
       StringBuilder sb = new StringBuilder();
        int len1 = word1.length();
        int len2 = word2.length();

        for (int i = 0; i < Math.max(len1, len2) ; i++) {
           if (i < len1) {
                sb.append(word1.charAt(i));
            }
            if (i < len2) {
                sb.append(word2.charAt(i));
            }
        }
        return sb.toString();
    }

public static void main(String args[])
{
    Scanner sc=new Scanner(System.in);
    String i,j,a="";
    System.out.println("ENTER 1ST WORD");
    i=sc.nextLine();
    System.out.println("ENTER 2ND WORD");
    j=sc.nextLine();
    Solution xx = new Solution();
          a=xx.mergeAlternately(i,j);
          System.out.println(a);
 }
}
```

Answer 3

```
import java.util.*;
class Solution {
    public int minSteps(String s, String t) {
        int[] arr = new int[26];
        int l = s.length(), sum = 0;
        for(int i = 0; i < l; i++){
            ++arr[s.charAt(i) - 'a'];
            --arr[t.charAt(i) - 'a'];
        }
        for(int i = 0; i < 26; i++) sum += Math.abs(arr[i]);
        return sum/2;
    }
public static void main(String args[])
{
    Scanner sc=new Scanner(System.in);
    String i,j;
    int x=0;
    System.out.println("ENTER 1ST STRING");
    i=sc.nextLine();
    System.out.println("ENTER 2ND STRING");
    j=sc.nextLine();
    Solution xx = new Solution();
          x=xx.minSteps(i,j);
          System.out.println(x);
 }
}
```

Answer 4

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

```
class Solution {

public List<Integer> spiralOrder(int[][] matrix) {

   int m = matrix.length;

   int n = matrix[0].length;

   int sz = m * n;

   List<Integer> res = new ArrayList<>();

   int r = 0, c = 0, R = m-1, C = n-1;

   int i = r, j = c;

   while (res.size() < sz) { // main loop

      while (j <= C && matrix[i][j] != 200) { // check visited and traverse right

         res.add(matrix[i][j]);

         matrix[i][j] = 200; // mark visited

         j++;

      }

                              // adjust i & j and remove row already traversed during right traversal

      j--;

      i++;

      r++;

      while (i <= R && matrix[i][j] != 200) { // check visited and traverse down

         res.add(matrix[i][j]);

         matrix[i][j] = 200;

         i++;

      }

                              // adjust i & j and remove column already traversed during down traversal

      i--;
```

```
          j--;
          C--;
          while (j >= c && matrix[i][j] != 200) { // check visited and traverse left
            res.add(matrix[i][j]);
            matrix[i][j] = 200;
            j--;
          }
                                    // adjust i & j and remove last column already traversed during left traversal
          j++;
          i--;
          c++;
          while (i >= r && matrix[i][j] != 200) { // check visited and traverse up
            res.add(matrix[i][j]);
            matrix[i][j] = 200;
            i--;
          }
                                    // adjust i & j and remove last row already traversed during up traversal
          i++;
          j++;
          R--;
        }
        return res;
      }
    }
```

Answer 5

```
import java.util.*;

class Solution {

public int[] sortArrayByParity(int[] nums) {

        int c=0;

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

            if(nums[j]%2==0){

                int temp=nums[j];

                nums[j]=nums[c];

                nums[c]=temp;

                c++;

             }

        }

return nums;

}

public static void main(String args[])

{

    Scanner sc=new Scanner(System.in);

    int i,j;


    System.out.println("ENTER LENGTH");

    i=sc.nextInt();

    int a[]=new int[i];
    int x[]=new int[i];

    System.out.println("ENTER VALUES");

    for(j=0;j<i;j++)

    {

      a[j]=sc.nextInt();

    }

      Solution xx = new Solution();

      for(int p=0;p<i;p++)

    {

      x[p]=xx.sortArrayByParity(a);

      System.out.println(x[p]);

    }

 }

}
```

Answer 6

```
import java.util.*;

class Solution {

    public int maxProfit(int[] prices) {

        int min=Integer.MAX_VALUE;

        int n=prices.length;

        int profit =0;

        int maxprofit=0;

        for(int i=0;i<n;i++){
            if(prices[i]<min){
                min=prices[i];
            }
            else if(prices[i]>min){
                profit=prices[i]-min;
                if(profit>maxprofit){
                    maxprofit=profit;
                }
            }

        }
        return maxprofit;
    }



public static void main(String args[])
{
    Scanner sc=new Scanner(System.in);
    int i,j;

    System.out.println("ENTER PRICE LENGTH");
    i=sc.nextInt();
    int a[]=new int[i];
    int x[]=new int[i];
    System.out.println("ENTER VALUES");
    for(j=0;j<i;j++)

    {

      a[j]=sc.nextInt();

    }

      Solution xx = new Solution();

      x=xx.maxProfit(a);

      System.out.println(x);

  }

}
```

Answer 7

```
import java.util.*;

    class Solution {

    public int maxProfit(int[] prices) {




        int[] buy = new int[prices.length];

        int[] sell = new int[prices.length];


        buy[0] = -prices[0];


        for(int i = 1; i < prices.length; i++) {


            buy[i] = Math.max(sell[i-1] - prices[i], buy[i-1]);
            sell[i] = Math.max(sell[i-1], buy[i-1] + prices[i]);

        }


        return Math.max(buy[prices.length - 1], sell[prices.length - 1]);

    }


public static void main(String args[])

{

    Scanner sc=new Scanner(System.in);

    int i,j;


    System.out.println("ENTER PRICE LENGTH");

    i=sc.nextInt();

    int a[]=new int[i];

    int x[]=new int[i];

    System.out.println("ENTER VALUES");

    for(j=0;j<i;j++)

    {

      a[j]=sc.nextInt();

    }

      Solution xx = new Solution();

      x=xx.maxProfit(a);

      System.out.println(x);

  }

}
```