Task 22:

Problem 1:

// you can also use imports, for example:

import java.util.\*;

// you can write to stdout for debugging purposes, e.g.

// System.out.println("this is a debug message");

class Solution {

    public boolean solution(int[] A) {

        int n=A.length;

        if(n%2!=0) return false;

        HashSet<Integer> hs=new HashSet<Integer>();

        for(int i=0;i<n;i++)

        {

            if(hs.contains(A[i]))

            hs.remove(A[i]);

            else hs.add(A[i]);

        }

        if(hs.isEmpty()) return true;

        else return false;

    }

}

Problem 2:

class Solution {

    public int solution(int[] A) {

   int max = 0, count = 0;

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

            if(max < A[i]) max = A[i];

            if((i + 1) == max) count++;

        }

        return count;

}

}

Problem 4:

class Solution {

    public int solution(String S) {

        // write your code in Java SE 8

        int n=S.length();

        int max\_val=-1;

        for(int i=0;i<n-1;i++)

        {

            String s1=Character.toString(S.charAt(i));

            String s2=Character.toString(S.charAt(i+1));

            String s3=s1+s2;

            int val=Integer.parseInt(s3);

            if(val>max\_val)

              max\_val=val;

        }

        return max\_val;

    }

}

Problem 5:

// you can also use imports, for example:

import java.util.\*;

// you can write to stdout for debugging purposes, e.g.

// System.out.println("this is a debug message");

class Solution {

    public int solution(String a) {

        ArrayList<String> a1 = new ArrayList<>();

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

    {

        for(int j = i + 1; j <= a.length(); j++)

        {

            if (i != j)

                a1.add(a.substring(i, j));

        }

    }

    TreeMap<String, Integer> a2 = new TreeMap<>();

    for(String s : a1)

        a2.put(s, a2.getOrDefault(s, 0) + 1);

    ArrayList<String> freshlist = new ArrayList<>();

    for(String s : a2.keySet())

    {

         if (a2.get(s) == 1)

            freshlist.add(s);

    }

 TreeMap<String, Integer> dictionary = new TreeMap<>();

    for(String s : freshlist)

    {

         dictionary.put(s, s.length());

    }

    ArrayList<Integer> newlist = new ArrayList<>();

    for(String s : dictionary.keySet())

        newlist.add(dictionary.get(s));

    int ans = Integer.MAX\_VALUE;

    for(int i : newlist)

        ans = Math.min(ans, i);

    return ans == Integer.MAX\_VALUE ? 0 : ans;

    }

}

Task 21:

Problem 1:

public String solution(String s) {

 int n=S.length();

        int i=0;

        for(;i<S.length()-1;i++)

        {

            if(S.charAt(i)>S.charAt(i+1))

            break;

        }

        return S.substring(0,i)+S.substring(i+1,n);

    }

Problem 2:

int n = A.length;

        int m = B.length;;

        Arrays.sort(A);

        Arrays.sort(B);

        int i = 0;

        for (int k = 0; k < n; k++) {

            if (i < m - 1 && B[i] < A[k]){

                i += 1;

                k--; continue;}

            if (A[k] == B[i])

                return A[k];

        }

        return -1;

Problem 3:

public String solution(String S) {

        // write your code in Java SE 8

        int n=S.length();

        String ans="";

        for(int i=0;i<n;i++)

        {

            if(S.charAt(i)=='m' && i+4<n )

              if(S.substring(i,i+5).equals("minus"))

               ans=ans+"-";

            if(S.charAt(i)=='p' && i+3<n )

              if(S.substring(i,i+4).equals("plus"))

               ans=ans+"+";

        }

        return ans;

    }

Problem 4:

Map<Long, Long> mp = new HashMap<Long, Long>();

        if(visits == null) return mp;

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

            if(visits[i] == null) {}

            else

            for(Map.Entry<String, UserStats> entry: visits[i].entrySet()) {

                try {

                long key = Long.parseLong(entry.getKey());

                Optional<Long> value = entry.getValue().getVisitCount();

                if(value.isPresent()) {

                    long pv = mp.getOrDefault(key, 0L);

                    mp.put(key, pv + value.get());

                }

                }catch(Exception ex) {

                    // ignore

                }

            }

        }

        return mp;

Group 6:

Problem 1:

 public int solution(int[] A) {

       int n=A.length;

      int max\_num=-1;

      Map<Integer,Integer> hm=new HashMap<Integer,Integer>();

      for(int i=0;i<n;i++)

      {

           if(hm.containsKey(A[i]))

           hm.replace(A[i],hm.get(A[i])+1);

          else hm.put(A[i],1);

          max\_num=Math.max(max\_num,A[i]);

      }

      //System.out.println("max num"+max\_num);

      int count=0;

      for(Map.Entry<Integer,Integer> entry : hm.entrySet())

      {

          int key=entry.getKey();

          int val=entry.getValue();

          if(key==max\_num ) count++;

          else if(val==1)

           count++;

         else if(val>=2) count=count+2;

         //System.out.println(key+" "+val);

         //System.out.println(count);

      }

      return count;

      }

Problem 2:

   int n=A.length;

        int m=A[0].length;

         int[] row = new int[n];

        int[] col = new int[m];

        int equi\_Row\_count = 0;

        int equi\_Col\_count = 0;

        int sum = 0;

        for (int i = 0; i < n; i++) {

            for (int j = 0; j < m; j++) sum += A[i][j];

            row[i] = sum;

            sum = 0;

        }

        for (int i = 0; i < m; i++) {

            for (int j = 0; j < n; j++) sum += A[j][i];

            col[i] = sum;

            sum = 0;

        }

        for (int i = 0; i < n; i++) {

            int left = 0;

            int right = 0;

            for (int j = 0; j < i; j++) left += row[j];

            for (int k = i + 1; k < n; k++) right += row[k];

            if (left == right) equi\_Row\_count++;

        }

        for (int i = 0; i < m; i++) {

            int left = 0;

            int right = 0;

            for (int j = 0; j < i; j++) left+= col[j];

            for (int k = i + 1; k < m; k++) right += col[k];

            if (left == right) equi\_Col\_count++;

        }

        return equi\_Row\_count \* equi\_Col\_count;

Group 7:

Problem 1:

if(N>100000000) return -1;

        String S=String.valueOf(N);

        int l=S.length();

        int flag[]=new int[10];

        for(int i=0;i<l;i++)

        {

            int k=Character.getNumericValue(S.charAt(i));

            flag[k]++;

        }

        String ans="";

        for(int i=9;i>=0;i--)

        {

            if(flag[i]!=0)

            {

                int a0=flag[i];

                while(a0>0)

                {

                 ans=ans+String.valueOf(i);

                 a0--;

                }

            }

        }

        return Integer.parseInt(ans);

problem 2:

int enable\_print = N % 10;

        while (N > 0) {

           if (N % 10 != 0) {

                enable\_print = 1;

            }

            if (enable\_print ==1) {

                System.out.print(N % 10);

            }

            N = N / 10;

            enable\_print=(N+1)%10;

problem 3:

long sum=0;

        int i=0;

        while(sum<N)

        {

         sum=i\*(i+1)/2;

         if(sum>=N)

         {

             return i-1;

         }

         i++;

        }

        return i-1;

task 19 group 12

problem 1:

int n=s.length();

        int countB=0,countA=0,countL=0,countO=0,countN=0;

        for(int i=0;i<n;i++)

        {

            if(s.charAt(i)=='B')

             countB++;

            else if(s.charAt(i)=='A') countA++;

            else if(s.charAt(i)=='L') countL++;

            else if(s.charAt(i)=='O') countO++;

            else if(s.charAt(i)=='N') countN++;

        }

        countL=countL/2;

        countO=countO/2;

        int min\_val= Math.min(Math.min(countA,countB),countN);

        min\_val=Math.min(Math.min(countO,countL),min\_val);

        return min\_val;

problem 2:

 int n=S.length();

    int num\_Bs = 0, min\_dels = 0;

    for(int i=0;i<n;i++) {

        char c=S.charAt(i);

        if (c=='A') {

            min\_dels = Math.min(num\_Bs, min\_dels + 1);

        }

        else {

            num\_Bs++;

            // there is no need to exclude the last B at the end of

            // the string, the min\_dels does not change

        }

    }

    return min\_dels;

problem 3:

int[] occurrences = new int[27];

for (char ch : S.toCharArray()) {

occurrences[ch -'a']++;

}

char best\_char = 'a';

int best\_res = 0;

for (int i = 0; i < 26; i++) {

if (occurrences[i] > best\_res) {

best\_char = (char)((int)'a' + i);

best\_res = occurrences[i];

}

}

return Character.toString(best\_char);

}

Problem 4:

String k[]=S.split("[.]");

        int n=k.length;

        int max\_len=0;

        for(int i=0;i<n;i++)

        {

            k[i]=k[i].trim();

            String m[]=k[i].split(" ");

            int l=m.length,count=0;

            for(int j=0;j<l;j++)

            {

               if(m[j].trim().isEmpty())

                count++;

            }

            l=l-count;

            if(max\_len<l) max\_len=l;

        }

        return max\_len;

Group 8:

Problem 1:

   int n=A.length;

        int m=A[0].length;

         int[] row = new int[n];

        int[] col = new int[m];

        int equi\_Row\_count = 0;

        int equi\_Col\_count = 0;

        int sum = 0;

        for (int i = 0; i < n; i++) {

            for (int j = 0; j < m; j++) sum += A[i][j];

            row[i] = sum;

            sum = 0;

        }

        for (int i = 0; i < m; i++) {

            for (int j = 0; j < n; j++) sum += A[j][i];

            col[i] = sum;

            sum = 0;

        }

        for (int i = 0; i < n; i++) {

            int left = 0;

            int right = 0;

            for (int j = 0; j < i; j++) left += row[j];

            for (int k = i + 1; k < n; k++) right += row[k];

            if (left == right) equi\_Row\_count++;

        }

        for (int i = 0; i < m; i++) {

            int left = 0;

            int right = 0;

            for (int j = 0; j < i; j++) left+= col[j];

            for (int k = i + 1; k < m; k++) right += col[k];

            if (left == right) equi\_Col\_count++;

        }

        return equi\_Row\_count \* equi\_Col\_count;

Problem 2:

 lic int solution(int[] A) {

        int n=A.length;

        int tp = 0;

        for (int i = 0; i < n; i++) {

    for (int c = 0; i + 2 < n && A[i + 1] - A[i] ==A[i + 2] - A[i + 1]; i++)

    {

                c++;

                tp += c;

    }

     }

     if(tp>1000000000) return -1;

     return tp;

Group 4:

Problem 1:

int n=A.length;

int sum=0,j,count=0,neg\_sum=0;

ArrayList<Integer> list=new ArrayList<Integer>();

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

{

if(A[i]<0)

{

j=i;

while(A[j]<0)

{

list.add(A[j]);

neg\_sum+=A[j];

j++;

}

Collections.sort(list);

i=j;

if(sum+neg\_sum<0){

for(int k=0;k<list.size();k++)

{

int m=neg\_sum-list.get(k);

if((sum+m)>0)

{

sum=sum+m;

count++;

//System.out.println(sum+"if");

break;

}

else {count++;

//System.out.println("else");

neg\_sum=m;

}

}}//if

neg\_sum=0;

}//if

sum=sum+A[i];

}//for

return count;

Problem 2:

int n=A.length,max\_len=1,len=1,index=0;

        for(int i=1;i<n;i++)

        {

             if(A[i]>A[i-1]){

              len++;

              if(max\_len<len) {

                  max\_len=len;

                  index=i-max\_len+1;

              }

             }

             else {len=1;

             }

        }

        return index;

Problem 3:

 int max\_dist=-1;

        int k=((A-C)\*(A-C)+(B-D)\*(B-D));

        int l=((B-C)\*(B-C)+(A-D)\*(A-D));

        int m=((A-B)\*(A-B)+(C-D)\*(C-D));

        if(k>max\_dist) max\_dist=k;

        if(l>max\_dist) max\_dist=l;

        if(m>max\_dist) max\_dist=m;

        return max\_dist;

Group 9:

Problem 1:

int n=A.length;

        int m=A[0].length;

        int count=0;

        HashMap<Integer,Integer> hm=new HashMap<Integer,Integer>();

        for(int i=0;i<n;i++)

        {

            for(int j=0;j<m;j++)

            {

                int val=A[i][j];

                if(hm.containsKey(val)){

                    int k=hm.get(val);

                 if(k!=-1 && k!=i) {count++;

                 hm.replace(val,-1);

                 }

                }

                else {

                    hm.put(val,i);

                }

            }

        }

        return count;

problem 2:

int n=message.length();

        if(n<=K) return message.trim();

        if(message.charAt(K-1)==' ') return message.trim();

        int index=0;

        for(int i=K;i>=0;i--)

        {

            if(message.charAt(i)==' '){

             index=i;

             break;}

        }

        return message.substring(0,index);

problem 3:

for(int i=0;i<n;i++)

        {

           //tm.put(S[i],P[i]);

           p+=P[i];

        }

        Arrays.sort(S);

        for(int i=n-1;i>0;i--)

        {

            if(p>0)

            {

                p=p-S[i];

                count++;

            }

        }

        return count;

task 20: group 13

problem 1:

int n=D.length;

        int count=1;

       int index=0;

        for(int i=1;i<n;i++)

        {

           for(int j=i-1;j>=index;j--)

            {

               if(Math.abs(D[i]-D[j])>X){

               count++;

               index=i;

                break;}

            }

        }

        return count;

problem 2:

 int N = A.length;

        int result = -1;

        int nX = 0;

        int nY = 0;

        for (int i = 0; i < N; i++) {

            if (A[i] == X)

                nX += 1;

            else if (A[i] == Y)

                nY += 1;

            if (nX == nY && nX!=0)

                result = i;

        }

        return result;

problem 3:

 int n=S.length();

        String ans="";

        int flag[]=new int[26];

        for(int i=0;i<n;i++)

        {

             int val=S.charAt(i)-97;

             //flag[val]++;

             if(flag[val]==1) {

                 ans=String.valueOf(S.charAt(i));

                 break;

             }

             flag[val]++;

        }

        return ans;

problem 4:

 assertEquals(Inverter.invert(null),"");

        assertEquals(Inverter.invert("a"),"a");

        assertEquals(Inverter.invert("abcd"),"dcba");

        assertEquals(Inverter.invert(""),"");