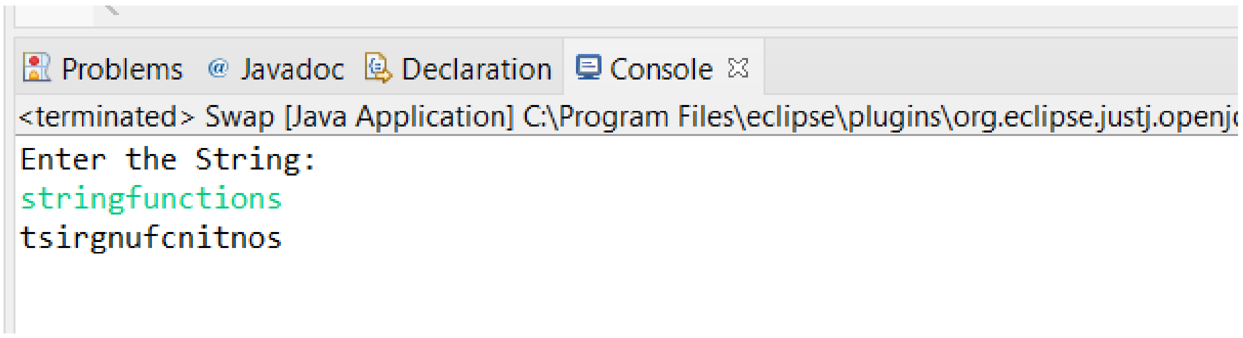
1. **Write a program to accept the string from user and rearrange the string in such a way that every pair of characters is exchanged. Print the final string**
   Example:
   "Stringfunctions" → "tSirgnufcnitnos"
   " java10.0"→ "ajav010."

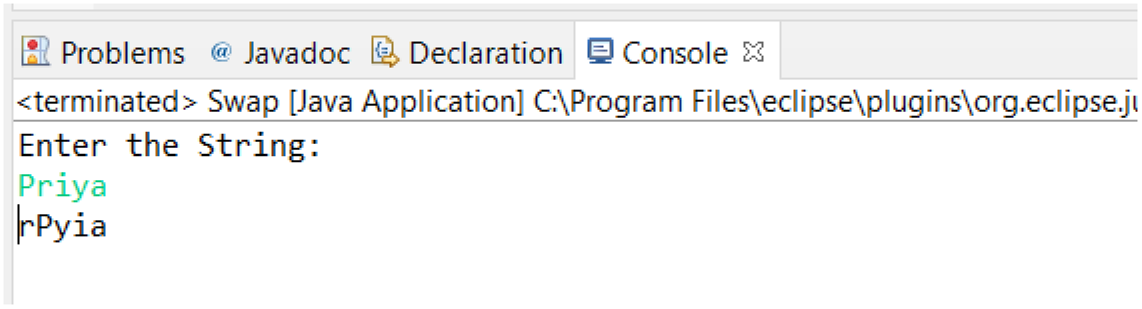
Code

```
import java.util.Scanner;

public class Swap {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		int len=0,i=0;
		char c=0;

	       System.out.println("Enter the String: ");
	       Scanner sc=new Scanner(System.in);
	       String str=sc.next();
	       len=str.length();
	       char[] ch = new char[len];
	       for(i=0;i<len;i++)
	       {
	          ch[i]=str.charAt(i);

	       }

	       for(i=1;i<len;i+=2)
	       {
	          c=ch[i];
	          ch[i]=ch[i-1];
	          ch[i-1]=c;
	       }

	       for(i=0;i<len;i++)
	           System.out.print(ch[i]);
	}

	}
```

OUTPUT



```
<terminated> Swap [Java Application] C:\Program Files\eclipse\plugins\org.eclipse.justj.openj
Enter the String:
stringfunctions
tsirgnufcnitnos
```



```
<terminated> Swap [Java Application] C:\Program Files\eclipse\plugins\org.eclipse.j
Enter the String:
Priya
rPyia
```

2. A hall comprised of NX N seating arrangements. Their aptitude level is mentioned as rating 1 to 5. (represented in NXN matrix). We have to identify the winning possibilities by analyzing their neighbo'sr aptitude level :

1. **If number of people with greater aptitude measure is greater than number of people with the lower aptitude measure then winning possibility is " -1"**
2. **If number of people with greater aptitude measure is lesser than number of people with the lower aptitude measure then winning possibility is " 1"**
3. **If number of people with greater aptitude measure is equal to the number of people with the lower aptitude measure then winning possibility is " 0"**

Print the winning possibility matrix.
Print how many are in winning category (1) , losing category (-1) , neutral(0)

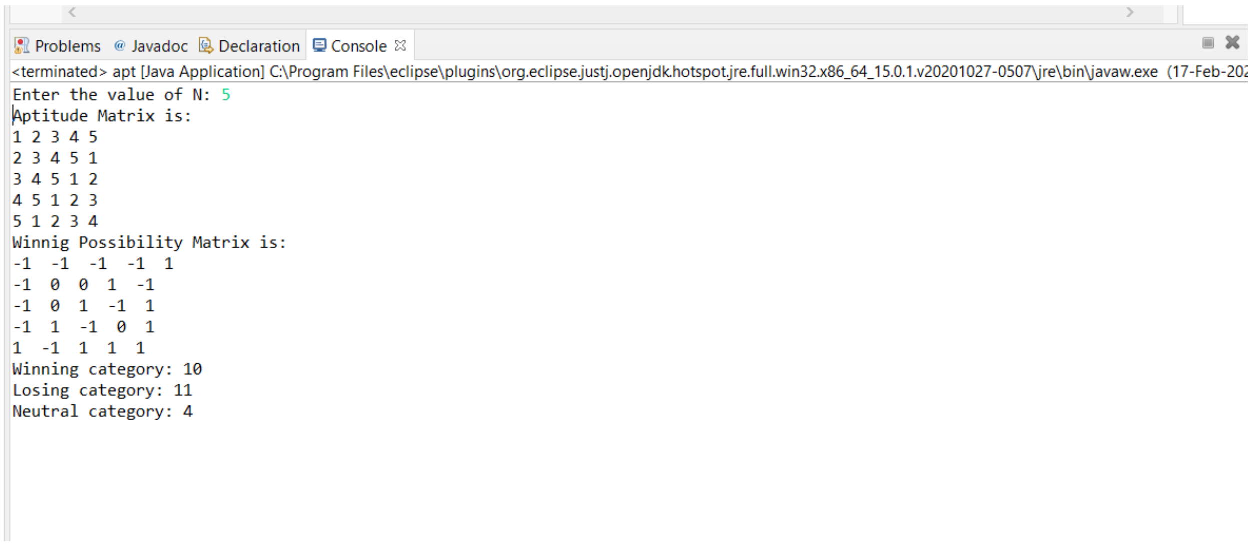
CODE

```
import java.util.*;
import java.io.*;
public class apt {
```

```
public static void main(String[] args) {
      // TODO Auto-generated method stub
      Scanner sc=new Scanner(System.in);
 System.out.print("Enter the value of N: ");
 int n=sc.nextInt();
 int aptitude[][]=new int[n][n];
 int prob[][]=new int[n][n];
 for(int i=0;i<n;i++)
 {
     int apt_filler_helper=i%5;
     for(int j=0;j<n;j++)
     {
         aptitude[i][j]=(apt_filler_helper)%5+1;
         apt_filler_helper++;
     }
 }
 System.out.println("Aptitude Matrix is: ");
 for(int i=0;i<n;i++)
 {
     for(int j=0;j<n;j++)
     {
         System.out.print(aptitude[i][j]+" ");
     }
     System.out.println();
 }
 int count0=0;
 int count1=0;
 int count1_minus=0;
 for(int i=0;i<n;i++)
 {
     for(int j=0;j<n;j++)
     {
         int greater_counter=0;
         int lower_counter=0;
         //top
         if(i>0)
         {
             if(aptitude[i][j]>aptitude[i-1][j])
                 lower_counter++;
             else greater_counter++;
         }
         //right
         if(j<n-1)
         {
             if(aptitude[i][j]>aptitude[i][j+1])
                 lower_counter++;
             else greater_counter++;
         }
         //bottom
         if(i<n-1)
         {
             if(aptitude[i][j]>aptitude[i+1][j])
                 lower_counter++;
             else greater_counter++;
         }
```

```
                //left
                if(j>0)
                {
                    if(aptitude[i][j]>aptitude[i][j-1])
                        lower_counter++;
                    else greater_counter++;
                }
                if(greater_counter>lower_counter)
                {
                    prob[i][j]=-1;
                    count1_minus++;
                }

                else if(greater_counter<lower_counter)
                {
                    prob[i][j]=1;
                    count1++;
                }
                else
                {
                    prob[i][j]=0;
                    count0++;
                }
            }
        }
        System.out.println("Winnig Possibility Matrix is: ");
        for(int i=0;i<n;i++)
        {
            for(int j=0;j<n;j++)
            {
                System.out.print(prob[i][j]+"  ");
            }
            System.out.println();
        }
        System.out.println("Winning category: "+count1);
        System.out.println("Losing category: "+count1_minus);
        System.out.println("Neutral category: "+count0);

       }

}
```

OUTPUT



```
<terminated> apt [Java Application] C:\Program Files\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.1.v20201027-0507\jre\bin\javaw.exe  (17-Feb-202
Enter the value of N: 5
Aptitude Matrix is:
1 2 3 4 5
2 3 4 5 1
3 4 5 1 2
4 5 1 2 3
5 1 2 3 4
Winnig Possibility Matrix is:
-1  -1  -1  -1  1
-1  0  0  1  -1
-1  0  1  -1  1
-1  1  -1  0  1
1  -1  1  1  1
Winning category: 10
Losing category: 11
Neutral category: 4
```

3. **Accept a string from user. Check whether it can be a palindrome if we change a single character. If it is so print " Can be palindrome" and the character need to be changed.**

   Else print "Not possible"

   Eg : abcdcbg→ reverse is "gbcdcba" If you are changing g to a then it became a palindrome

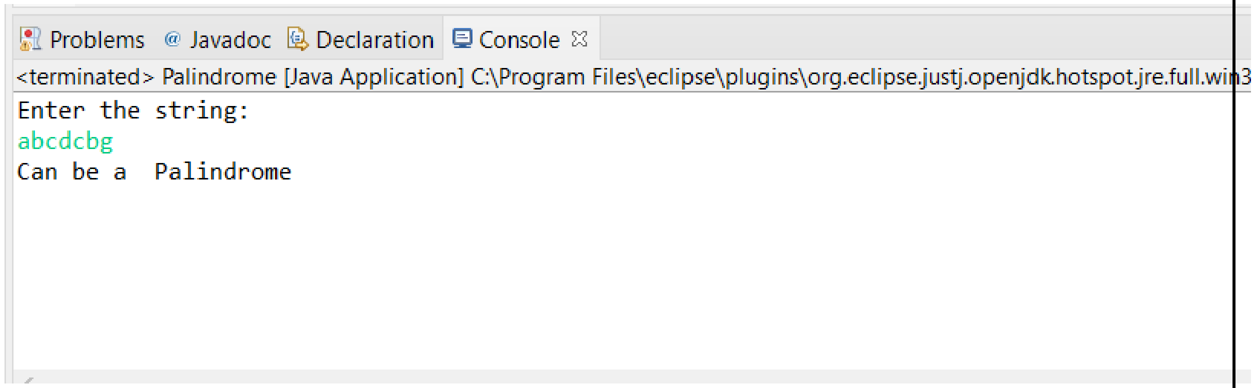
   abcdaaa→ " Not possible"

CODE

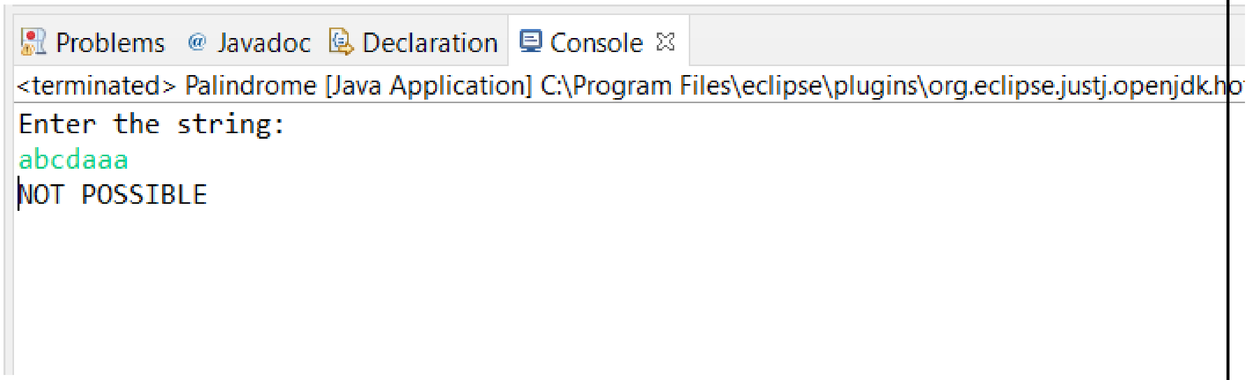
```
import java.util.*;
public class Palindrome {



     static int[] isPalindrome(String str)
    {
        int result[]=new int[2];
        result[0]=1;
        result[1]=0;

      int i=0,j=str.length()-1;
      while(i<j)
      {
             if(str.charAt(i)!=str.charAt(j))
                    {
                    result[0]=0;
                    result[1]++;
```

```
                }
            i++;
            j--;
        }
        return result;
    }
        public static void main(String[] args) {
            System.out.println("Enter the string: ");
            Scanner sc=new Scanner(System.in);
                String str=sc.next();
                int checker[]=isPalindrome(str);
            if(checker[0]==1 ||(checker[0]==0 && checker[1]==1))
                System.out.print("Can be a  Palindrome");
            else
                System.out.println("NOT POSSIBLE");

        }

    }
```

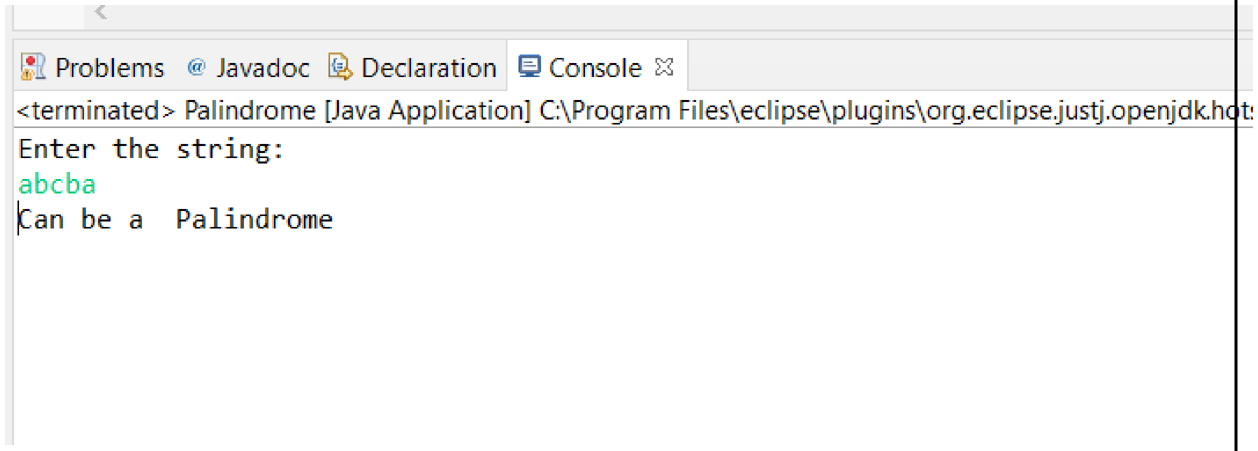
OUTPUT



```
<terminated> Palindrome [Java Application] C:\Program Files\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win3
Enter the string:
abcdcbg
Can be a  Palindrome
```



```
<terminated> Palindrome [Java Application] C:\Program Files\eclipse\plugins\org.eclipse.justj.openjdk.ho
Enter the string:
abcdaaa
NOT POSSIBLE
```


Problems @ Javadoc Declaration Console

```
<terminated> Palindrome [Java Application] C:\Program Files\eclipse\plugins\org.eclipse.justj.openjdk.hot
Enter the string:
abcba
Can be a  Palindrome
```

4. **Accept the two dimensional array from the user. Rearrange them in such a way that all single digit number in the first column, number starting with '1' in the second column and number starting with the 2 in the third column and so on. Print the resultant matrix in the matrix format.**

Eg

```
3   86  12  34
45  5   11  24
14  67  34  41
```

Output

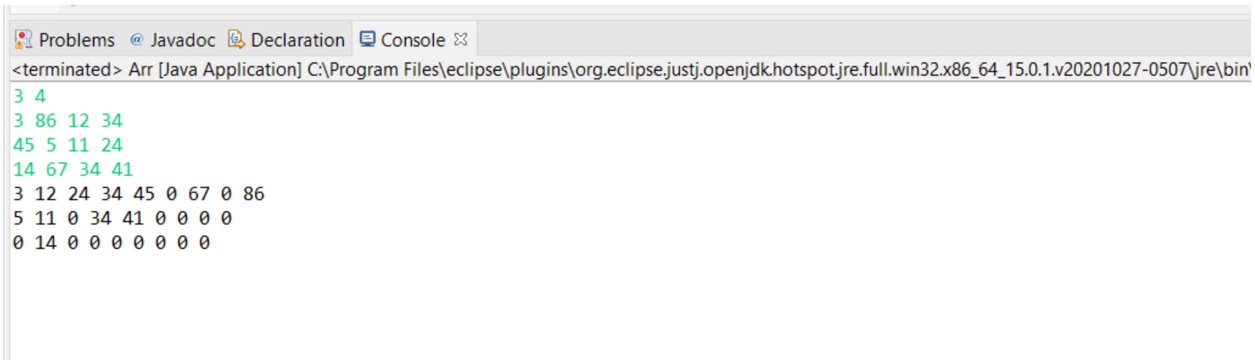
```
3  12  24   34   45  0    67  0  86
5   11   0   34   41  0   0    0  0
0   14   0   0    0    0   0    0  0
```

The output matrix have maximum 10 columns and row size is the maximum column size. In the above example lengthiest column is second column and size is 3 . So 3 is the row size for all columns . If any of the cell does not have value mark it as zero.

CODE

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


public class Arr {
        static int countDigit(int n)
    {
        int count = 0;
        while (n != 0)
        {
            n = n / 10;
            ++count;
        }
        return count;
    }
    static int firstDigit(int n)
    {
        while (n >= 10)
            n /= 10;
        return n;
    }

        public static void main(String[] args) {
              // TODO Auto-generated method stub
        Scanner sc = new Scanner(System.in);
        int n1,n2,r,c;
        n1=sc.nextInt();
        n2=sc.nextInt();
        r=10;c=n1*n2;
        int arr[][]=new int[n1][n2];
        int newarr[][]=new int[r][c];
        Integer colptr[]=new Integer[r];
        for(int i=0;i<r;i++)
            colptr[i]=0;
        for(int i=0;i<n1;i++)
            for(int j=0;j<n2;j++)
                arr[i][j]=sc.nextInt();
        int init=r+1,last=0;
        for(int i=0;i<n1;i++)
        {
            for(int j=0;j<n2;j++)
            {
                if(countDigit(arr[i][j])==1)
                {
                    newarr[0][colptr[0]++]=arr[i][j];
                    init=0;
                }
                else
                {
                    int idx=firstDigit(arr[i][j]);
                    newarr[idx][colptr[idx]++]=arr[i][j];
                    init=init>idx ? idx:init;
                    last=last<idx ? idx:last;
                }
            }
        }
        int maxNumData=Collections.max(Arrays.asList(colptr));
        for(int j=0;j<maxNumData;j++)
        {
            for(int i=init;i<=last;i++)
                System.out.print(newarr[i][j]+" ");
            System.out.println();
        }
    }


     }
```

OUTPUT


Problems Javadoc Declaration Console

```
<terminated> Arr [Java Application] C:\Program Files\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_15.0.1.v20201027-0507\jre\bin\
3 4
3 86 12 34
45 5 11 24
14 67 34 41
3 12 24 34 45 0 67 0 86
5 11 0 34 41 0 0 0 0
0 14 0 0 0 0 0 0 0
```