41. Playing with String - II

Write a program to accept a string array as input, convert all the elements into lowercase and sort the string array. Display the sorted array.

Include a class UserMainCode with a static method **sortArray** which accepts the string array. The return type is the string array formed based on requirement.

Create a Class Main which would be used to accept the string array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer which denotes the size of the array followed by the array of strings,
Output consists of a string array.
Refer sample output for formatting specifications.

Sample Input 1:
5
AAA
BB
CCCC
A
ABCDE

Sample Output 1:
a
aaa
abcde
bb
cccc

```
publicclass Main {
publicstaticvoid main(String[] args) throws ParseException {
Scanner sc = new Scanner(System.in);
int n=sc.nextInt();
String[] a= new String[n];
for(int i=0;i<n;i++)
      a[i]=sc.next();
String[] res=User.sortArray(a);
for(int i=0;i<res.length;i++)
      System.out.println(res[i]);
```

```
}
}
```

```
publicclass User {
publicstatic String[] sortArray (String s[]) {
      String[] a=new String[s.length];
for(int i=0;i<s.length;i++)
{
a[i]=s[i].toLowerCase();
}
Arrays.sort(a);
return a;
}
}
```

42. Median Calculation

Write a program to accept an int array as input, and calculate the median of the same.

Median Calculation Procedure:

1. Sort the sequence of numbers.
2. The total number count is odd, Median will be the middle number.

The total number count is even, Median will be the average of two middle numbers, After calculating the average, round the number to nearest integer.

Include a class UserMainCode with a static method **calculateMedian** which accepts the int array. The return type is the integer which would be the median.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer which denotes the size of the array followed by the array of integers.
Output consists of a integer.
Refer sample output for formatting specifications.

Sample Input 1:

7

1

2
1
4
7
1
2

Sample Output 1:

2

Sample Input 2:

6
52
51
81
84
60
88

Sample Output 2:

71

```
publicclass Main {
publicstaticvoid main(String[] args) throws ParseException {
Scanner sc = new Scanner(System.in);
int n=sc.nextInt();
int[] a= newint[n];
for(int i=0;i<n;i++)
      a[i]=sc.nextInt();
System.out.println(User.calculateMedian (a));

}
}



publicclass User {

publicstaticint calculateMedian(int s[]) {
      double med=0;
      double avg=0;
Arrays.sort(s);
int mid=s.length/2;
if(s.length%2!=0)
      med=s[mid];
else
{
```

```
avg=(double)(s[mid]+s[mid-1]\)/2;
     System.out.println(avg);
       med=Math.ceil(avg);
}
return(int)med;
}
}
```

43. Sequence in Array

Write a program to accept an int array as input, and check if [1,2,3] appears somewhere in the same sequence.

Include a class UserMainCode with a static method **searchSequence** which accepts the int array. The return type is a boolean which returns true or false.

Create a Class Main which would be used to accept the integer array and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a an integer which denotes the size of the array followed by the array of integers.
Output should print true or false.
Refer sample output for formatting specifications.

Sample Input 1:
9
11
-2
5
1
2
3
4
5
6

Sample Output 1:
TRUE

Sample Input 2:
6
-2
5
1
3
2
6

Sample Output 2:

FALSE

```
publicclass Main {
publicstaticvoid main(String[] args) throws ParseException {
Scanner sc = new Scanner(System.in);
int n=sc.nextInt();
int[] a= newint[n];
for(int i=0;i<n;i++)
      a[i]=sc.nextInt();
boolean b=User.calculateMedian (a);
System.out.println(b);

}
}

publicclass User {
publicstaticboolean calculateMedian(int s[]) {
      int[] a={1,2,3};
      int n=s.length-(a.length-1);
      boolean b=false;
      for(int i=0;i<n;i++)
      {
            if(s[i]==a[0] )
            {
                  if(s[i+1]==a[1])
                  {
                        if(s[i+2]==a[2])
                        {
                              b=true;
                              break;
                        }
                        else
                              b=false;
            }
                  else
                        b=false;
            }
            else
                  b=false;
      }
      return b;
}
}
```

44. Asterisk & Characters

Write a program to read a string and return true or false based on the below rule:

1. Return true if for every '*' in the string, there are same characters both side immediately before and after the star, else return false.

Include a class UserMainCode with a static method **scanStarNeighbors** which accepts the string. The return type is the boolean TRUE or FALSE based on the rule.

Note: The check is case-insensitive.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.
Output consists of TRUE or FALSE.
Refer sample output for formatting specifications.

Sample Input 1:
Hello*World

Sample Output 1:
FALSE

Sample Input 2:
Welcome*elizabeth

Sample Output 2:
TRUE

```
publicclass Main {
publicstaticvoid main(String[] args) throws ParseException {
Scanner sc = new Scanner(System.in);
String n=sc.next();

boolean b=User.scanStarNeighbors(n);
System.out.println(b);

}
}
```

```
publicclass User {
publicstaticboolean scanStarNeighbors(String s) {
	StringTokenizer st=new StringTokenizer(s,"*");
	boolean b=false;
	while(st.hasMoreTokens())
	{
		String s1=st.nextToken();
		String s2=st.nextToken();
		if(s1.charAt(s1.length()-1)==s2.charAt(0))
		{
			b=true;
		}
	}
	return b;
}
}
```

45. Occurance Count

Write a program to read a string that contains a sentence and read a word. Check the number of occurances of that word in the sentence.

Include a class UserMainCode with a static method **countWords** which accepts the two strings. The return type is the integer giving the count.

Note: The check is case-sensitive.

Create a Class Main which would be used to accept the two strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings.
Output consists of count indicating the number of occurances.
Refer sample output for formatting specifications.

Sample Input 1:
Hello world Java is best programming language in the world
world

Sample Output 1:
2

Sample Input 2:

hello world

World

Sample Output 2:

0

```
publicclass Main {
publicstaticvoid main(String[] args) throws ParseException {
Scanner sc = new Scanner(System.in);
String s1=sc.nextLine();
String s2=sc.next();
intb=User.countWords (s1,s2);
System.out.println(b);

}
}
```

```
publicclass User {
publicstaticint countWords (String s1,String s2) {
	StringTokenizer st=new StringTokenizer(s1," ");
	int c=0;
	while(st.hasMoreTokens())
	{
		String s3=st.nextToken();
	if(s3.equals(s2))
		{
			c++;
		}
	}
	return c;
}
}
```

46.Regular Expressions - III

Write a program to read two strings S1 & S2, compute the number of times that S2 appears in S1.

Include a class UserMainCode with a static method **searchString** which accepts the two strings. The return type is the integer giving the count.

Note: The check is case-insensitive.

Create a Class Main which would be used to accept the two strings and call the static method present in UserMainCode.

Input and Output Format:

Input consists of two strings.

Output consists of count indicating the number of occurances.

Refer sample output for formatting specifications.

Sample Input 1:

Catcowcat

cat

Sample Output 1:

2

Sample Input 2:

Catcowcat

catp

Sample Output 2:

0

```
publicclass User {
	publicstaticint scanStarNeighbors(String s1,String s2) {
		int ls1=s1.length();
		int ls2=s2.length();
		int n=ls1-(ls2-1);
		System.out.println(n);
		int ct=0;
		for(int i=0;i<n;i++)
		{

			String ss=s1.substring(i,i+(ls2));
			if(s2.equals(ss))
			ct++;
			}

		return ct;
	}
	}
```

47. Strings Processing

Write a program to read a string that contains comma separated fruit names and also a number N. Pick the nth fruit and return it. If the total number of elements are less than the number specified in N, then return the last element.

Include a class UserMainCode with a static method **findFruitName** which accepts the the string and the number n. The return type is the string which has the fruit name.

Create a Class Main which would be used to accept the string and integer and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string and integer.
Output consists of a string.
Refer sample output for formatting specifications.

Sample Input 1:
Apple,Banana,Orange
2

Sample Output 1:
Banana

Sample Input 2:
Apple,Banana,Orange
4

Sample Output 2:
Orange

```
import java.text.ParseException;
importjava.util.ArrayList;
importjava.util.HashMap;
importjava.util.Iterator;
import java.util.Scanner;

publicclass Main {
publicstaticvoid main(String[] args) throws ParseException {
Scanner sc = new Scanner(System.in);
```

```
String s1=sc.nextLine();
int n=sc.nextInt();
System.out.println(User.findFruitName(s1,n));
}
}

importjava.util.ArrayList;
importjava.util.Arrays;
importjava.util.LinkedHashSet;
import java.util.StringTokenizer;
publicclass User {
publicstatic String findFruitName(String s1,int n) {
	StringTokenizer st=new StringTokenizer(s1,",");
	intc=0,i=0;
	String ss=null;

	String[] s=new String[st.countTokens()];

	while(st.hasMoreTokens())
	{
		s[i]=st.nextToken();
		i++;
	}
	if(i>n)
	{
	 ss=s[n-1];
	}
	else
	{
		ss=s[i-1];//last element display
	}

	return ss;
}
}
```

48. Proper Case

Write a program to read a string and convert the intial letter of each word to uppercase.

Include a class UserMainCode with a static method **changeCase** which accepts the string. The return type is the modified string.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.
Output consists of a string.
Refer sample output for formatting specifications.

Sample Input 1:
This is cognizant academy

Sample Output 1:
This Is Cognizant Academy

49. Length of same word

Write a program to read a string containing multiple words find the first and last words, if they are same, return the length and if not return the sum of length of the two words.

Include a class UserMainCode with a static method **compareLastWords** which accepts the string. The return type is the length as per problem.

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a string.
Output consists of a integer.
Refer sample output for formatting specifications.

Sample Input 1:
This is Cognizant Academy

Sample Output 1:
11

Sample Input 2:

Hello World Hello

Sample Output 2:

5

```
import java.util.StringTokenizer;
publicclass UserMainCode {
	publicstatic String changeWord(String s)
	{
		StringTokenizer st=new StringTokenizer(s," ");
		StringBuffer sb=new StringBuffer();
		while(st.hasMoreTokens())
		{
			String s1=st.nextToken();
			sb.append(s1.substring(0,1).toUpperCase());
			sb.append(s1.substring(1));
			sb.append(" ");

		}
		return sb.toString();
	}
}
```

50. Perfect Number

Write a program to that takes a positive integer andreturns true if the number is perfect number.

A positive integer is called a perfect number if the sum of all its factors (excluding the number itself, i.e., proper divisor) is equal to its value.

For example, the number 6 is perfect because its proper divisors are 1, 2, and 3, and 6=1+2+3; but the number 10 is not perfect because its proper divisors are 1, 2, and 5, and 1+2+5 is not equal to 10

Include a class UserMainCode with a static method **getPerfection** which accepts the number. The

return type is boolean (true / false).

Create a Class Main which would be used to accept the string and call the static method present in UserMainCode.

Input and Output Format:

Input consists of a integer.
Output consists of TRUE / FALSE.
Refer sample output for formatting specifications.

Sample Input 1:
28

Sample Output 1:
TRUE

```
publicclass User {
	publicstaticboolean scanStarNeighbors(intn) {
		boolean b;

		int sum=0;

			for(int i=1;i<n;i++)
			{
				if(n%i==0)
				{
				 sum=sum+i;
				 System.out.println(sum);
				}

			}
			if(sum==n)
			{
				b=true;
			}
			else
			{
				b=false;
			}


		return b;
	}
```

```
                }
----------------------------------------
```