/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* return difference between given and input String\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args) {

String strGiven = "abcdefghijklmnopqrstuvwxyz";

String strInput = "OnlinetestwithGSclient";

findAndPrintUncommonChars(strGiven,strInput);

}

//Difference between given and input String

static void findAndPrintUncommonChars(String strGiven,String strInput)

{

// size of the hash table

int MAX\_CHAR = 256;

String str1 = (strGiven.replaceAll("\\s","")).toLowerCase();

String str2 = (strInput.replaceAll("\\s","")).toLowerCase();

System.out.println("strGiven==="+str1);

System.out.println("strInput==="+str2);

// mark presence of each character as 0

// in the hash table 'present[]'

int present[] = new int[MAX\_CHAR];

for (int i = 0; i < MAX\_CHAR; i++)

{

present[i] = 0;

}

int l1 = str1.length();

int l2 = str2.length();

// for each character of str1, mark its

// presence as 1 in 'present[]'

for (int i = 0; i < l1; i++)

{

present[str1.charAt(i) - 'a'] = 1;

}

// for each character of str2

for (int i = 0; i < l2; i++)

{

// if a character of str2 is also present

// in str1, then mark its presence as -1

if (present[str2.charAt(i) - 'a'] == 1

|| present[str2.charAt(i) - 'a'] == -1)

{

present[str2.charAt(i) - 'a'] = -1;

}

// else mark its presence as 2

else

{

present[str2.charAt(i) - 'a'] = 2;

}

}

// print all the uncommon characters

for (int i = 0; i < MAX\_CHAR; i++)

{

if (present[i] == 1 || present[i] == 2)

{

System.out.print((char) (i + 'a') + " ");

}

}

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*first repeating character\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args) {

String strarr[] = { "array", "apple", "rat"};

firstRepeatingChars(strarr);

}

//Difference between given and input String

static void firstRepeatingChars(String strarr[])

{

String strFull = "";

for(int n=0 ; n < strarr.length ; n++)

{

strFull += strarr[n];

}

System.out.println("strFull==="+strFull);

//remove all the spaces

strFull = strFull.replaceAll(" ", "");

Character rptChar = null;

//Will store each character and it's count

HashMap<Character, Integer> map = new HashMap<Character, Integer> ();

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

Character chr = strFull.charAt(i);

if(map.containsKey(chr)){

map.put(chr,map.get(chr)+1);

}else{

map.put(chr, 1);

}

}

//Iterate the string and return the character for which the count is > 1 in map

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

if(map.get(strFull.charAt(i))>1){

rptChar = strFull.charAt(i);

break;

}

}

System.out.println("first repeat character==" + rptChar);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*panagram detector\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args) {

String strPanagram="The quick brown fox jumps over the lazy";

panagramDetector(strPanagram);

}

//To find the string is panagram or not

static void panagramDetector(String str)

{

boolean[] alphaList = new boolean[26];

int index = 0;

int flag = 1;

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

if ( str.charAt(i) >= 'A' && str.charAt(i) <= 'Z') {

index = str.charAt(i) - 'A';

}else if( str.charAt(i) >= 'a' && str.charAt(i) <= 'z') {

index = str.charAt(i) - 'a';

}

alphaList[index] = true;

}

for (int i = 0; i <= 25; i++) {

if (alphaList[i] == false)

flag = 0;

}

System.out.print("String: " + str);

if (flag == 1)

System.out.print("\nThe above string is a pangram.");

else

{

System.out.print("\nThe above string is not a pangram.");

}

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* perfect number \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*perfect number \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args) {

isPerfect();

}

private static void isPerfect() {

int num, sum = 0;

int i;

for (num = 1; num < 100; num++) {

for (int j = 1; j <= num - 1; j++) { // change made here

if (num % j == 0) {

sum = sum + j;

}

}

if (sum == num) {

System.out.println(sum);

}

sum = 0; // change made here

}

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*strong number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Strong number:

class IsAStrongNumberOrNot  
{  
    public static void main(String s[])  
    {  
        int input = 145;  
          
        int temp, mod, sum = 0;  
      
        temp = input;  
      
        while(input > 0)  
        {  
            mod = input % 10;  
      
            sum=sum + fact (mod);  
      
            input  = input /10;  
                      
        }  
        if(temp == sum)  
        {  
            System.out.println(temp +" is a strong number");  
        }  
        else  
        {  
            System.out.println(temp +" is not a strong number");  
        }  
    }  
    public static int fact(int num)  
    {  
        int fact=1;  
      
        for(int i = 1; i <= num; i++)  
        {  
            fact \*= i;  
        }  
        return fact;  
    }  
          
}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Amstrong number\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Armstrong number

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args) {

int n =153;

isAmstrongNumber(153);

}

private static void isAmstrongNumber(int n)

{

int c=0,a,temp;

temp=n;

while(n>0)

{

a=n%10;

n=n/10;

c=c+a\*a\*a;

System.out.println(" c==="+c);

}

if(c==temp)

System.out.println("the number is Amstrong==="+temp);

else

System.out.println("the number is not Amstrong==="+temp);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*numberology\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args) {

String input = "S. KANAPATHY";

numerology(input);

}

static void numerology(String input)

{

String strNumerology = input.replaceAll("\\s", "");

strNumerology =strNumerology.replaceAll("[^a-zA-Z0-9\_-]","");

System.out.println("strNumerology==="+strNumerology);

int sum=0;

int a=0;

int b=0;

int c=0;

int d=0;

int e=0;

int f=0;

int g=0;

int h=0;

for(int n=0; n < strNumerology.length() ; n++)

{

if(strNumerology.charAt(n)==('A')|| strNumerology.charAt(n)==('J') || strNumerology.charAt(n)==('Q') || strNumerology.charAt(n)==('Y') || strNumerology.charAt(n)==('I'))

{

a=1;

//sum=sum+a;

}

else if(strNumerology.charAt(n)==('B')|| strNumerology.charAt(n)==('K') || strNumerology.charAt(n)==('R'))

{

b=2;

//sum=sum+b;

}

else if(strNumerology.charAt(n)==('C')|| strNumerology.charAt(n)==('G') || strNumerology.charAt(n)==('L') || strNumerology.charAt(n)==('S'))

{

c=3;

//sum=sum+c;

}

else if(strNumerology.charAt(n)==('D')|| strNumerology.charAt(n)==('M') || strNumerology.charAt(n)==('T'))

{

d=4;

//sum=sum+d;

}

else if(strNumerology.charAt(n)==('E')|| strNumerology.charAt(n)==('H') || strNumerology.charAt(n)==('X'))

{

e=5;

//sum=sum+e;

}

else if(strNumerology.charAt(n)==('U')|| strNumerology.charAt(n)==('V') || strNumerology.charAt(n)==('W'))

{

f=6;

//sum=sum+f;

}

else if(strNumerology.charAt(n)==('O')|| strNumerology.charAt(n)==('Z') )

{

g=7;

//sum=sum+g;

}

else if(strNumerology.charAt(n)==('B')|| strNumerology.charAt(n)==('F') || strNumerology.charAt(n)==('P'))

{

h=8;

//sum=sum+h;

}

sum= a+b+c+d+e+f+g+h;

}

System.out.println("sum of the input string"+sum);

int z=0;

int num=0;

while(sum>10)

{

z= sum%10;

num = z+num;

}

System.out.println("numerology==="+num);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*combination is stairs\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*prime factors \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// Program to print all prime factors

import java.io.\*;

import java.lang.Math;

class GFG {

    // A function to print all prime factors

    // of a given number n

    public static void primeFactors(int n)

    {

        // Print the number of 2s that divide n

        while (n % 2 == 0) {

            System.out.print(2 + " ");

            n /= 2;

        }

        // n must be odd at this point.  So we can

        // skip one element (Note i = i +2)

        for (int i = 3; i <= Math.sqrt(n); i += 2) {

            // While i divides n, print i and divide n

            while (n % i == 0) {

                System.out.print(i + " ");

                n /= i;

            }

        }

        // This condition is to handle the case whien

        // n is a prime number greater than 2

        if (n > 2)

            System.out.print(n);

    }

    public static void main(String[] args)

    {

        int n = 315;

        primeFactors(n);

    }

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* median of 2 sorted arrays \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args) {

int arr1[] = {2,3,6,7,9};

int arr2[] = {1,4,8,10};

double k= findMedianSortedArrays(arr1,arr2);

System.out.println("median==="+k);

}

static double findMedianSortedArrays(int[] nums1, int[] nums2) {

int total = nums1.length+nums2.length;

if(total%2==0){

return (getKth(nums1, 0, nums1.length-1, nums2, 0, nums2.length-1, total/2)

+ getKth(nums1, 0, nums1.length-1, nums2, 0, nums2.length-1, total/2-1))/2.0;

}else{

return getKth(nums1,0, nums1.length-1, nums2, 0, nums2.length-1, total/2);

}

}

//k is the index starting from 0

static int getKth(int[] nums1, int i1, int j1, int[] nums2, int i2, int j2, int k){

if(j1<i1){

return nums2[i2+k];

}

if(j2<i2){

return nums1[i1+k];

}

if(k==0){

return Math.min(nums1[i1], nums2[i2]);

}

int len1 = j1 - i1 + 1;

int len2 = j2 - i2 + 1;

int m1 = k\*len1/(len1+len2);

int m2 = k - m1 - 1;

m1 += i1;

m2 += i2;

if(nums1[m1]<nums2[m2]){

k = k-(m1-i1+1);

j2 = m2;

i1 = m1+1;

}else{

k = k-(m2-i2+1);

j1 = m1;

i2 = m2+1;

}

return getKth(nums1, i1, j1, nums2, i2, j2, k);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* unique substring

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args)

{

String s = "aab";

int n = 2;

// Function calling

result(s, n);

}

// Function to print the

// unique sub-String of length n

static void result(String s1,int n)

{

// set to store the Strings

HashSet<String> st = new HashSet<String>();

for (int i = 0; i < s1.length(); i++)

{

String ans = "";

for (int j = i; j < s1.length(); j++)

{

ans += s1.charAt(j);

// if the size of the String

// is equal to 1 then insert

if (ans.length()== n)

{

// inserting unique

// sub-String of length L

st.add(ans);

break;

}

}

}

// Printing the set of Strings

for (String it : st)

System.out.print(it + " ");

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* longest occurring character\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args) {

String strlongest = "aabbbbbcc";

longOccurrChar(strlongest);

}

static void longOccurrChar(String str)

{

List<String> parts = Arrays.asList(str.split("(?<=(.))(?!\\1)"));

String max = Collections.max(parts, Comparator.comparing(s -> s.length()));

System.out.println("largest substring: " + max);

int index = str.indexOf(max);

System.out.println("index of largest substring: " + index);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* input 2D array of strings average of strings \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import java.io.\*;

import java.util.\*;

/\*

\* To execute Java, please define "static void main" on a class

\* named Solution.

\*

\* If you need more classes, simply define them inline.

\*/

class Solution {

public static void main(String[] args) {

String student[][] = {{"Rohit", "85"}, {"Rahul", "80"}, {"Amit","85"},{"Rohit", "90"}};

int bestNumber=hishestAverage(student);

System.out.println("bestnumber==="+bestNumber);

}

static int hishestAverage(String[][] scores)

{

if(scores == null || scores.length == 0) {

return -1;

}

int highestAve = 0;

Map<String, List<Integer>> map = new HashMap<>();

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

List<Integer> scoreList = map.get(scores[i][0]);

if(scoreList == null) {

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

currentScore.add(Integer.valueOf(scores[i][1]));

map.put(scores[i][0], currentScore);

} else {

scoreList.add(Integer.valueOf(scores[i][1]));

map.put(scores[i][0], scoreList);

}

}

//go through the map. find the largest ave

for(Map.Entry<String, List<Integer>> entry : map.entrySet()) {

int currentAveScore = aveCalculate(entry.getValue());

highestAve = Math.max(highestAve, currentAveScore);

}

return highestAve;

}

static int aveCalculate(List<Integer> scores) {

int len = scores.size();

int sum = 0;

for(int score : scores) {

sum += score;

}

float ave = sum / len;

return (int) ave;

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/