\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*week-6\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ROLL N0:230701233

1. Given a String input1, which contains many number of words separated by : and each word contains exactly two lower case alphabets, generate an output based upon the

below 2 cases. Note:

1. All the characters in input 1 are lowercase alphabets.
2. input 1 will always contain more than one word separated by :
3. Output should be returned in uppercase. Case 1:

Check whether the two alphabets are same.

If yes, then take one alphabet from it and add it to the output.

Example 1:input1 = ww:ii:pp:rr:oo output = WIPRO

Expanation:

wod1 is ww, both are same hence take w wrd2 is ii, both are same hence take i

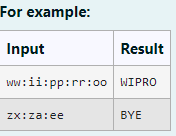
ord3 is pp, both are same hence take pword4 is rr, both are same hence take r word5 is oo, both are same hence take o

Hence the output is WIPRO Case 2:

If the two alphabets are not same, then find the position value of them and find maximum value – minimum value.

Take the alphabet which comes at this (maximum value - minimum value) position in the alphabet series.

Example 2” input1 = zx:za:ee output = BYE



Code:

import java.util.\*; class diff{

char different(char a, char b){ if ((int)a != (int)b)

return (char)((int)'a' + ((int)a-(int)b) - 1); return a;

}

}

public class Main{

public static void main(String[] args){ Scanner scan = new Scanner(System.in); diff z = new diff();

String q = scan.nextLine(); StringBuffer ans = new StringBuffer(); StringBuffer temp = new StringBuffer(); for(int i = 0;i < q.length();i++){

if(q.charAt(i) == ':'){ temp.append(" ");

}

else{

temp.append(Character.toString(q.charAt(i)));

}

}

String h = temp.toString();

for(int i = 0;i < temp.length();i++){ if(i%3 == 0){

ans.append(Character.toString(z.different(h.charAt(i),h.charAt(i+1))));

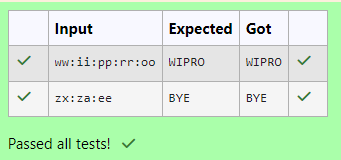
}

}

System.out.print(ans.toString().toUpperCase());

}

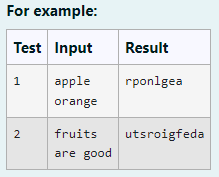
}

Output:

1. Given 2 strings input1 C input2.
   * Concatenate both the strings.
   * Remove duplicate alphabets C white spaces.
   * Arrange the alphabets in descending order. Assumption 1:

There will either be alphabets, white spaces or null in both the inputs. Assumption 2:

Both inputs will be in lower case.



Code:

import java.util.\*;

public class HelloWorld {

public static void main(String[] args) { Scanner scan = new Scanner(System.in); String a = scan.nextLine();

String b = scan.nextLine(); StringBuffer ab = new StringBuffer();

if(a.trim().isEmpty() CC b.trim().isEmpty()){ System.out.print("null");

}

else{

for(int i = 0;i < a.length();i++){ if (a.charAt(i) != ' ') {

ab.append(Character.toString(a.charAt(i)));

}

}

for(int i = 0;i < b.length();i++){ if (b.charAt(i) != ' '){

ab.append(Character.toString(b.charAt(i)));

}

}

char[] d = ab.toString().toCharArray();

Arrays.sort(d);

for(int i = d.length - 1;i >= 1;i--){ if(d[i] != d[i-1])

System.out.print(d[i]);

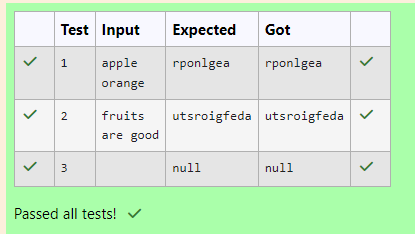
}

System.out.print(d[0]);

}

}

}

Output:

1. You are provided a string of words and a 2-digit number. The two digits of the number represent the two words that are to be processed.

For example:

If the string is "Today is a Nice Day" and the 2-digit number is 41, then you are expected to process the 4th word ("Nice") and the 1st word ("Today").

The processing of each word is to be done as follows:

Extract the Middle-to-Begin part: Starting from the middle of the word, extract the characters till the beginning of the word.

Extract the Middle-to-End part: Starting from the middle of the word, extract the characters till the end of the word.

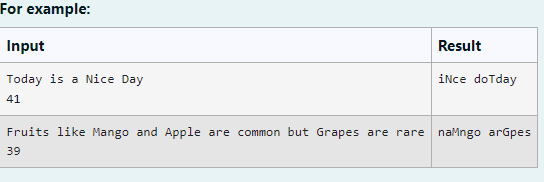
If the word to be processed is "Nice":

Its Middle-to-Begin part will be "iN". Its Middle-to-End part will be "ce".

So, merged together these two parts would form "iNce". Similarly, if the word to be processed is "Today":

Its Middle-to-Begin part will be "doT". Its Middle-to-End part will be "day".

So, merged together these two parts would form "doTday".



Code:

import java.util.\*; public class mix{

public static void main(String[] args){ Scanner scan = new Scanner(System.in); String g = scan.nextLine();

int n = scan.nextInt(),ones,flag = 0; StringBuffer temp = new StringBuffer(); StringBuffer temp1 = new StringBuffer(); int space = 0;

while (n > 0){

ones = (n %10) - 1;

for(int i = 0; i < g.length();i++){ if (g.charAt(i) == ' '){

space = space + 1;

}

else if(space == ones CC flag == 0){ temp.append(Character.toString(g.charAt(i)));

}

else if(space == ones CC flag == 1){

temp1.append(Character.toString(g.charAt(i)));

}

}

space = 0 ; flag = 1;

n = n /10;

}

rew m = new rew();

System.out.println(m.r(temp1.toString()) + " " + m.r(temp.toString()));

}

}

class rew{

String r(String a){

int le = a.length(),n,q;

StringBuffer temp3 = new StringBuffer(); if(le % 2 == 1){

n = ((int)(le/2));

q = ((int)(le/2));

}

else{

n = ((int)(le/2)) - 1;

q = ((int)(le/2));

}

for(int i = n;i >= 0;i--){

temp3.append(Character.toString(a.charAt(i)));

}

for(int i = q;i < le;i++){

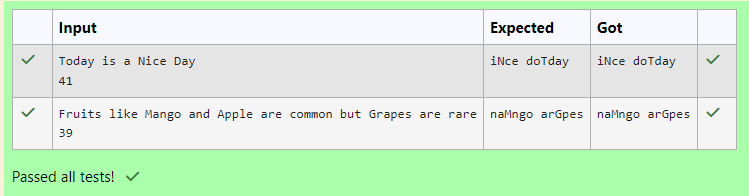
temp3.append(Character.toString(a.charAt(i)));

}

return temp3.toString();

}

}

Output: