**Sort a string according to the order defined by another string**

**Given two strings (of lowercase letters), a pattern, and a string. The task is to sort strings according to the order defined by the pattern. It may be assumed that the pattern has all characters of the string and all characters in the pattern appear only once.**

**Examples:**

**Input : pat = "bca", str = "abc"**

**Output : str = "bca"**

**Input : pat = "bxyzca", str = "abcabcabc"**

**Output : str = "bbbcccaaa"**

**Input : pat = "wcyuogmlrdfphitxjakqvzbnes", str = "jcdokai"**

**Output : str = "codijak"**

**Program:**

public class Main

{

static String stringRea(String s1,String s2)

{

StringBuilder ans=new StringBuilder();

int[] apl=new int[26];

for(int i=0;i<s2.length();i++)

{

apl[s2.charAt(i)-'a']++;

}

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

{

for(int j=0;j<apl[s1.charAt(i)-'a'];j++)

{

ans.append(s1.charAt(i));

}

}

return ans.toString();

}

public static void main(String[] args) {

String s1="bca";

String s2="abck";

String pat = "bxyzca", str = "abcabcabch";

String pat1 = "wcyuogmlrdfphitxjakqvzbnes", str2 = "jcdokai";

System.out.println(stringRea(s1,s2));

System.out.println(stringRea(pat,str));

System.out.println(stringRea(pat1,str2));

}

}

**Find the string among given strings represented using given encryption pattern**

**Given an array of strings arr[] of size N and an encrypted string str, the task is to find the correct string from the given array of strings whose encryption will give str where str is encrypted using the following rules:**

**The starting characters form an integer representing the number of uppercase symbols In the decrypted string.**

**The next 3 characters are the last 3 characters of the decrypted string in reverse order.**

**The last few characters also form an integer representing the sum of all digits in the password.**

**The length of each string in the array is at least 3 and if there is more than one correct answer, print among them.**

**Examples:**

**Input: arr[] = {“P@sswORD1”, “PASS123word”}, str = “4dro6”**

**Output: PASS123word**

**Explanation: The decrypted string representing str = “4dro6” should have**

**4 upper case letters, sum of all digits in it as 6 and ends with “ord”.**

**The output string satisfies all the following properties.**

**Input: arr[] = {“Geeks”, “code”, “Day&Night”}, str = “1thg10”**

**Output: -1**

**Explanation: No such string exists which satisfies the encryption.**

**Program:**

import java.util.\*;

public class Main

{

static boolean noOfChapL(String s,int n)

{

int c=0;

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

{

char ch=s.charAt(i);

if(ch>='A'&&ch<='Z')

{

c++;

}

else

{

break;

}

}

if(c==n)

{

return true;

}

return false;

}

static boolean lastChar(String s,String last)

{

int len=last.length();

StringBuilder s1=new StringBuilder();

for(int i=s.length()-len;i<s.length();i++)

{

char ch=s.charAt(i);

s1.append(ch);

}

if(last.equals(s1.toString()))

{

return true;

}

return false;

}

static boolean suminString(String s,int sumN)

{

int c=0;

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

{

char ch=s.charAt(i);

if(ch>='0'&&ch<='9')

{

c=c+(ch-'0');

}

}

if(sumN==c)

{

return true;

}

return false;

}

static String FindTheDecode(String[] words,String str)

{

int nCap=str.charAt(0)-'0';

StringBuilder s2=new StringBuilder();

StringBuilder sumN=new StringBuilder();

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

{

char ch=str.charAt(i);

if(Character.isAlphabetic(ch))

{

s2.append(ch);

}

if(Character.isDigit(ch))

{

sumN.append(ch);

}

}

String s3=s2.reverse().toString();

int sumNu=Integer.parseInt(sumN.toString());

for(String ch:words)

{

if(noOfChapL(ch,nCap) && suminString(ch,sumNu) && lastChar(ch,s3))

{

return ch;

}

}

return "-1";

}

public static void main(String[] args) {

String[] wor={"P@sswORD1", "PASS123word"};

String str="4dro6";

String[] wor1={"Geeks", "code", "Day&Night"};

String str1="1thg10";

System.out.println(FindTheDecode(wor,str));

System.out.println(FindTheDecode(wor1,str1));

}

}

**concatenating uncommon characters of given strings**

**Given two strings S1 and S2. The task is to concatenate uncommon characters of the S2 to S1 and return the resultant string S1 .**

**Examples:**

**Input: S1 = “aacdb”, S2 = “gafd”**

**Output: “cbgf”**

**Input: S1 = “abcs”, S2 = “cxzca”;**

**Output: “bsxz”**

**Program:**

import java.util.\*;

public class Main

{

static String uncommonChar(String s1,String s2)

{

StringBuilder ans=new StringBuilder();

LinkedHashMap<Character,Integer> hm=new LinkedHashMap<>();

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

{

char ch=s1.charAt(i);

hm.put(ch,hm.getOrDefault(ch,0)+1);

}

for(int i=0;i<s2.length();i++)

{

char ch=s2.charAt(i);

hm.put(ch,hm.getOrDefault(ch,0)+1);

}

for(Character ch:hm.keySet())

{

if(hm.get(ch)==1)

{

ans.append(ch);

}

}

return ans.toString();

}

public static void main(String[] args) {

String S1 = "aacdb", S2 = "gafd";

String S11 = "abcs", S21 = "cxzca";

System.out.println(uncommonChar(S1,S2));

System.out.println(uncommonChar(S11,S21));

}

}