**Duração da execução**

Faça com que a função RunLength( **str** ) pegue o parâmetro **str** sendo passado e retorne uma versão compactada da string usando o algoritmo de codificação Run-length. Esse algoritmo funciona pegando a ocorrência de cada caractere repetido e emitindo esse número junto com um único caractere da sequência repetida. Por exemplo: "wwwggopp" retornaria **3w2g1o2p** . A string não conterá números, pontuação ou símbolos.

**Exemplos**

Entrada: "aabbcde"  
Saída: 2a2b1c1d1eEntrada: "wwwbbbw"  
Saída: 3w3b1w

import java.util.\*;

import java.io.\*;

class Main {

  public static String RunLength(String str) {

    StringBuffer bf = new StringBuffer();

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

      int count=1;

      while(i<str.length() -1(&& str.charAt(i))==str.charAt(i+1)){

        count++;

        i++;

      }

      bf.append(str.charAt(i);

      bf.append(count);

    }

    return bf.toString();

  }

  public static void main (String[] args) {

    Scanner s = new Scanner(System.in);

    System.out.println(doEncode( str. "aabccab"));

  }

}

import java.util.\*;

import java.io.\*;

class Main {

  public static String RunLength(String str) {

    StringBuilder sb = new StringBuilder();

    Pattern pattern = Pattern.compile("(([a-zA-Z])\\2\*)");

    pattern.matcher(text).result.group(1).length();

    sb.append(result.group(2)).append(repetitions);

  });

    return sb.toString();

  }

  public static void main (String[] args) {

    // keep this function call here

    Scanner s = new Scanner(System.in);

    System.out.print(RunLength(s.nextLine()));

  }

}

**Quatro maiores**

Faça com que a função LargestFour( **arr** ) pegue a matriz de inteiros armazenados em **arr** e encontre os quatro maiores elementos e retorne sua soma. Por exemplo: se **arr** é [4, 5, -2, 3, 1, 2, 6, 6] então os quatro maiores elementos nesta matriz são 6, 6, 4 e 5 e a soma total desses números é **21** , então seu programa deve retornar **21** . Se houver menos de quatro números na matriz, seu programa deve retornar a soma de todos os números da matriz.

**Exemplos**

Entrada: new int[] {1, 1, 1, -5}  
Saída: -2Entrada: new int[] {0, 0, 2, 3, 7, 1}  
Saída: 13

import java.util.\*;

import java.io.\*;

class Main {

  public static int LargestFour(int[] arr) {

    // code goes here

    return arr[0];

  }

  public static void main (String[] args) {

    // keep this function call here

    Scanner s = new Scanner(System.in);

    System.out.print(LargestFour(s.nextLine()));

  }

}

**Subcadeia Palindrômica**

Faça com que a função PalindromicSubstring( **str** ) pegue o parâmetro **str** que está sendo passado e encontre a substring palindrômica mais longa, o que significa a substring mais longa que é lida da mesma forma para frente e para trás. Por exemplo: se **str** for "abracecars", seu programa deve retornar a string **racecar** porque é o palíndromo mais longo dentro da string de entrada.  
  
A entrada conterá apenas caracteres alfabéticos minúsculos. A substring palindrômica mais longa sempre será exclusiva, mas se não houver nenhuma com mais de 2 caracteres, retorne a string **none** .

**Exemplos**

Entrada: "hellosannasmith"  
Saída: sannasEntrada: "abcdefgg"  
Saída: nenhuma

import java.util.\*;

import java.io.\*;

class Main {

  public static String PalindromicSubstring(String str) {

    // code goes here

    return str;

  }

  public static void main (String[] args) {

    // keep this function call here

    Scanner s = new Scanner(System.in);

    System.out.print(PalindromicSubstring(s.nextLine()));

  }

}

Imagine que você é o(a) arquiteto(a) responsável pelo desenho de um sistema de logins para um site extremamente concorrido de entradas de um mega show de Rock em Rio (got it?). Dado que o número de entradas é realmente limitado e muito inferior à quantidade de acesso, você precisa garantir que o site só irá finalizar a venda para as pessoas que vão realmente entrar, ou seja, você não pode deixar uma pessoa comprar uma pessoa ingresso sem que haja mais disponível. Além disso, um cliente com internet mais lenta não ficaria feliz de não conseguir comprar seu ingresso pois uma pessoa com internet mais veloz passou sua frente.

Desenhe uma estrutura de software que sustente esse serviço, pode ser um desenho simples, da maneira que você preferir, desde que o mesmo transmita uma ideia da arquitetura que você teve para quem o lê. Colocamos aqui um exemplo de desenho de uma arquitetura aleatória para servir de inspiração.

<http://help.market.com.br/csharp/hmfile_hash_6c8b0b18.png>

**Para nos enviar seu desenho (posta desse exercício), suba a imagem do mesmo GitHub em um link aquires na caixa de respostas.**

Boa sorte!