Resolução de Problemas - Arrays e Matrizes

Exercicio 1 - Missing Number

Solução em Python

```
def missibgNumber():
    x = input()
    arr = input().split(" ") # Ao usar o split, a entrada vira array

    for i, elm in enumerate(arr): #Transforma os elmentos do array para inteiro
        arr[i] = int(elm)

    arr.sort() #Ordena os elementos
    last = arr[0] - 1 #Variavel para verificar o elemento anteriror no loop

    #Tratando casos especiais
    if x == '2': #Caso o array tenha apenas 1 elemento
        a = 1 if arr[0] == 2 else 2
        print(a)
        return 0

    for elm in arr:
        if elm-1 != last: #Verifica se o elemento atual quebra a sequencia
            print(elm-1) #Mostra o elemento faltante
```

```
            return 0
        last = elm

    print(last+1) #Se nenhum elemento quebra  a sequencia, o faltante é o ultimo
    return 0
```

Solução em Java

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        int cont, soma = 0, soma_aux = 0;
        // captura o tamanho do array
        int tamanho = input.nextInt();
        // captura o 'enter' do usuário, para evitar erros
        input.nextLine();
        // cria um vetor com o tamanho inserido pelo usuário
        int[] vetor = new int[tamanho];
        // loop para preencher o array
        // variavel 'soma' para somar os valores dos números inseridos
        for (cont = 0; cont < tamanho-1; cont++){
            vetor[cont] = input.nextInt();
            soma += vetor[cont];
        }
        // loop para realizar a soma correta caso todos os números estejam pres
entes
        for (cont = 1; cont <= tamanho; cont++){
            soma_aux += cont;
        }
        // imprime na tela a diferença entre a 'soma correta' e a soma obtida do u
```

```
suário, que resulta no valor que esta faltando
        System.out.println(soma_aux - soma);
    }
}
```

Exercicio 2 - Incereasing Array

Solução em Python

```
def increasingArray():
    x = input()
    arr = input().split(" ") # Ao usar o split, a entrada vira array

    maxim = int(arr[0]) #Definindo o 1o elemento como o maior
    resposta = 0

    for i in range(1, len(arr)):
        if int(arr[i]) < maxim: #Apenas vamos a alterar elementos menores que o
maior
            qtd = maxim-int(arr[i]) #qtd é a diferença entre o maior elemento e o at
ual
            resposta += qtd

        maxim = max(maxim, int(arr[i])) #Atualizando o maior valor
    print(resposta)
```

Solução em Java

```
import java.util.Scanner;
```

```
public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        // captura o tamanho do array
        int tamanho = input.nextInt();
        int cont;
        long qtde_mov = 0, maior;
        input.nextLine();
        // cria um array com o tamanho indicado pelo usuario
        long[] vetor = new long[tamanho];
        boolean status;
        // captura os números do array
        for (cont = 0; cont < tamanho; cont++){
            vetor[cont] = input.nextInt();
        }
        // atribui a variavel 'maior' o primeiro elemento do array
        maior = vetor[0];
        // percorre o array e verifica se o elemento atual eh maior do que o maior elemento
        // caso seja maior, atribuo a variavel 'qtde_mov' a diferenca entre o maior valor e o valor atual, que corresponde justamente a quantidade de incrementos que tenho que realizar
        // minha posicao atual recebe o valor do maior elemento
        // caso nao seja maior, ou ele eh igual ou menor que o proximo elemento. Ou seja, ja esta em ordem crescente, eu apenas atualizo a variavel de maior valor.
        for (cont = 0; cont < tamanho; cont++){
            if(vetor[cont] < maior){
                qtde_mov += (maior - vetor[cont]);
                vetor[cont] = maior;
            } else maior = vetor[cont];
        }
        System.out.println(qtde_mov);
    }
}
```

Exercicio 3 - Chessboard and Queens

Solução em Python

```
def isPossible(i, j, col, lin, diag1, diag2):
    #A rainha ataca na vertical, horizontal e diagonal
    # Todas as casas com a mesma linha tem o mesmo i
    # Todas as casas com a mesma coluna tem o mesmo j
    # todas as casas da mesma diagonal1 tem o mesmo (i+j)
    # todas as casas da mesma diagonal2 tem o mesmo (i-j)
    if (i not in lin) and (j not in col ) and ((int(i)+int(j)) not in diag1) and ((int(i)-int
(j)) not in diag2):
        return True
    return False


def queens():
    mtx = []
    for i in range(8): #Coletando o input e transformando em uma matriz 8×8
        arr = list(input())
        mtx.append(arr)

    possibles = 0 #Variavel que armazena a resposta final

    #Arrays usados para salvar colunas, linhas e diagonais proibidas
    col = [] #Pela forma padrão de percorrer matrizes, decidimos mapear as rai
nhas colocadas pela coluna
    lin = []
    diag1 = []
    diag2 = []

    last = 0 #Caso um caminho der errado, é necessário voltar a ultima rainha c
```

```
olocada
    i = 0

    while True: #A condição de parada é um break
        flag = False #Usada para saber se na linha atual alguma rainha foi posicio
nada

        for j in range(last, 8): #Loop percorre a partir da ultima rainha tentada na l
inha atual
            if isPossible(i, j, col, lin, diag1, diag2) and mtx[i][j] == '.': #Verifica se po
de posicionar uma rainha na posição atual
                last = 0 #A proxima linha não teve nenhuma tentativa de posição de
rainha
                #Atualiza proibições
                lin.append(i)
                col.append(j)
                diag1.append(int(i)+int(j))
                diag2.append(int(i)-int(j))
                flag = True #Indica que uma rainha foi posicionada na linha
                break #Sai do loop interno
            elif j == 7: #Só entra nessa condição se nenhuma rainha pode ser posi
cionada na linha
                flag = False



        if flag: #Se teve rainha posicionada
            if i == 7: # Se estamos na ultima linha do tabuleiro
                possibles = possibles+1 #Posicionamos todas as rainhas de uma no
va forma
                #Volta uma interação para verificar outras possibilidades
                lin.pop(-1)
                last = col.pop(-1) #A ultima rainha verificada esta na ultima coluna a
dicionada
                last+=1 #A próxima coluna a verificar é a ultima +1
                diag1.pop(-1)
                diag2.pop(-1)
```

```
            else: #Se não estamos na ultima linha, passa para a próxima
                i+=1
        else:#Se não teve rainha posicionada
            if i==0: #Se verificamos todas as possibilidades, finaliza o loop
                break
            #Se ainda tem mais possibilidades, remove a ultima rainha colocada
            i -= 1 #Volta uma linha
            #Volta uma interação para verificar outras possibilidades
            lin.pop(-1)
            last = col.pop(-1) #A ultima rainha verificada esta na ultima coluna adic
ionada
            last+=1 #A próxima coluna a verificar é a ultima +1
            diag1.pop(-1)
            diag2.pop(-1)


    print(possibles) #Mostra a resposta
```

Exercicio 4 - Decent Arrays

Solução em Python

```
def decent_array():
    #Coletando as entradas
    x = input()
    arr = input().split(" ") # Ao usar o split, a entrada vira array
    last = 0

    for i in range(len(arr)):
        if int(arr[i]) < last: # No momento que algum elemento for menor que o ult
imo, retorna false
            print("No")
```

```
            return 0
        last = int(arr[i]) #Atualiza o ultimo

    print("Yes") #Se sair do loop, todos os elementos são maiores que o anterio
r
    return 0
```

Solução em Java

```
import java.util.Scanner;

public class main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        int cont;
        // leitura do tamanho do vetor
        int tamanho_vetor = input.nextInt();
        input.nextLine();
        // inicialização do vetor
        int[] vetor = new int[tamanho_vetor];
        // leitura dos dados do vetor
        for (cont = 0; cont < tamanho_vetor; cont++){
            vetor[cont] = input.nextInt();
        }
        // verifica se tiver uma ocorrencia no minimo de que o vetor não seja cre
scente, para a verificação e marca "No"
        for (cont = 0; cont < (tamanho_vetor-1); cont++){
            if(vetor[cont] > vetor[cont+1]) {
                System.out.print("No");
                return;
            }
        }
```

```
        System.out.print("Yes");


    }
}
```

Exercicio 5 - 2D Array - DS

Solução em Python

```
def sumHourglass(mtx, i, j): #Função retorna a soma dos elementos no hourgl
ass
    x = int(mtx[i][j]) + int(mtx[i][j+1]) + int(mtx[i][j+2])
    x += int(mtx[i+1][j+1]) + int(mtx[i+2][j]) + int(mtx[i+2][j+1]) + int(mtx[i+2][j+
2])
    return x

def hourglass():
    mtx = []

    for _ in range(6):
        mtx.append(list(map(int, input().rstrip().split()))) #Coleta o input e format
a para matriz

    maximum = -63 #O valor maximo inicial é a menor soma possivel segundo
as constraints

    #Como o array é de tamanho fixo (6×6) percorremos até que não possa ser
formada a figura do hourglass
    for i in range(4):
        for j in range(4):
```

```
            my_sum = sumHourglass(mtx, i, j) #Realiza a soma dos elementos
            maximum = max(my_sum, maximum) # Verifica o maior entre eles

    print(maximum)
```

Solução em Java

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

class Solution {

    public static int hourglassSum(List<List<Integer>> arr) {
        //variavel para acumular a soma da ampulheta atual
        int currentGlassSum;
        //variavel que carregara o valor da maior soma de ampulheta
        //ela eh inicializada com 0 para caso hajam somas negativas e nao de err
o na primeira comparacao
        int greatestGlassSum = 0;
        //variavel para referenciar a linha atual que estou verificando
        int line = 0;
        //laco para percorrer as colunas da matriz, em que qeuremos ir ate a posi
cao 3
        for (int column = 0; column <= 4; column++) {
            //incializamos a soma da ampulheta atual como 0
            currentGlassSum = 0;
            //se a variavel da coluna for igual a 4, quer dizer que todas as ampulhe
tas que iniciam na primeira linha foram analisadas e podemos partir pra prox li
nha
            if(column == 4) {
                line++;
                column = 0;
```

```
            }
            //somo os 3 valore da primeira linha da ampulheta
            currentGlassSum += arr.get(line).get(column);
            currentGlassSum += arr.get(line).get(column+1);
            currentGlassSum += arr.get(line).get(column+2);

            //somo o valor do meio da segunda linha da ampulheta
            currentGlassSum += arr.get(line+1).get(column+1);

            //somo os 3 valores da terceira linha da ampulheta
            currentGlassSum += arr.get(line+2).get(column);
            currentGlassSum += arr.get(line+2).get(column+1);
            currentGlassSum += arr.get(line+2).get(column+2);

            //se for a primeira ampulheta, eu seto como o maior valor a propria so
ma
            if(column == 0 && line == 0) {
                greatestGlassSum = currentGlassSum;
            }
            //senao, comparo a soma atual com a maior ate entao e sempre vou gu
ardando a maior
            greatestGlassSum = Math.max(currentGlassSum, greatestGlassSum);

            //condicao de parada do laco, pois o valor 4 da iteracao eh usado para
verificar se devemos passar para a proxima linha
            if (line == 3 && column == 3) break;
        }

        return greatestGlassSum;
    }

    public static void main(String[] args) {
        //crio a matriz 6×6 para iterar sobre e colocar os valores do input do usu
ario
        List<List<Integer>> matrix = new ArrayList<>();
        Scanner scanner = new Scanner(System.in);
```

```
        for (int i = 0; i < 6; i++) {
            matrix.add(new ArrayList<>());
            for (int j = 0; j < 6; j++) {
                matrix.get(i).add(j, scanner.nextInt());
            }
        }
        //printo o resultado da funcao com a matriz como parametro
        System.out.println(hourglassSum(matrix));
    }
}
```

Exercicio 6 - Building a List

Solução em Python

```
def buildingList():
    cases = int(input())

    for i in range(cases): #Loop para realizar todos os casos
        length = int(input())
        list_chars = list(input())
        list_chars.sort() #Ordena os caracteres da entrada (ex: Para a entrada [b,
a,c] gera [a,b,c])



        words = [] #Array que armazena as respostas possiveis

        #1)A adição de palavras é feita de forma crescente de tamanho, depois el
es serão ordenados
        #ex: pra uma entrada abc primeiro adicionaremos 'a', 'b', 'c' depois 'ab',
'ac' ...
        for i in range(length): #Esse for indica o tamanho da string que será adici
```

```
onada
        aux = ''
        maxlen = i+1 #O tamanho maximo de uma stringa para cada interação é i+1  (ex: Quando i=0, só adicionamos strings de tamanho 1)
        j=0 #Variavel que percorre o list_chars

        verify = i*(-1) if i != 0 else -1 #O verify é usado para saber se todas as possibilidades foram percorridas
        removes = -1 #Esta variável indica quantas remoções devem ser feitas ao atingir o limite de uma string antes de passar para a próxima
        while True:
            if len(aux) >= maxlen: #Sempre que atingir o tamanho maximo, elem entos são removidos baseado na variável removes
                aux = aux.replace(aux[removes:],'')
                removes = -1
            aux += list_chars[j] #Em toda interação uma nova letra é adicionada à string, baseado em j

            #2) O exercício só permite palavras em que todo carcter à esquerda é menor
            #Nesse caso para uma entrada = 'abcd' e maxlen = 3 não existe 'bc a' como resposta
            #Adicionamos o actual para controle de parada
            if len(aux) == 1:
                actual = j

            #3) Sempre que a string aux atinja o tamanho maximo ela vai ser um a resposta válida
            if len(aux) == maxlen:
                words.append(aux) #Adicionamos a palavra à lista de respostas



            #4)Toda vez que o j chega ao final da lista de chars realizamos algu mas verificações
            if j == length-1:
                #4.1)Passamos para a próxima interação se:
```

```
            #4.1.1- O tamanho da lista de chars menos o tamanho maximo da s
tring da interação for igual à posição atual
            #ex: list_chars = [a,b,c,d,e] , maxlen = 4 , actual = 1  passaria no t
este
            #traduzindo: não tem como uma string iniciando em 'c' formar um
a string válida de tamanho 4, pois ele precisa de outros 3 elementos maiores
que ele
            # assim 'cbde' não é válido, a ultima string valida seria 'bcde'

            #4.1.2- o caracter da string na posição de verificação for igual ao l
istchar na posição de verificação
            #ex:  list_chars = [a,b,c,d,e] , aux = 'abce' , verify = -3  compara o
'c' com o 'b'
            #Isso quer dizer que so posso pasar para o proximo tamanho se a
tingi a maior combinação possivel do tamanho atual
            if actual == length-maxlen and aux[verify] == list_chars[verify]:
                break
            #4.2)Se atingimos um array valido que possui a ultima letra do list
_chars mas ele não é o maior possivel
            #Verificamos quantas remoções devem ser feitas na proxima inter
ação para seguir
            #ex1: list_chars = [a,b,c,d,e] , aux = 'abce' , verify = -3  gera remo
ves = -2 pois ainda existe 'abde' válida
            #ex2: list_chars = [a,b,c,d,e] , aux = 'abde' , verify = -3  gera remo
ves = -3 pois ainda existe 'acde' válida
            elif aux[verify] != list_chars[verify]:
                for e in range(-2, (verify-1),-1):
                    if aux[e] != list_chars[e]:
                        removes = e
                        break
                j = list_chars.index(aux[removes]) + 1
            #4.3) Caso contrário, o j inicia a partir do atual mais 1 e zeramos a
string
            #ex: j=0 , actual=0 , j = 1
            # ou seja, Se minha string era 'ac' ela vai ser zerada e na proxima
```

```
interação vai iniciar com 'b'
                else:
                    j = actual + 1
                    aux = ''
            else:#Se o j não for o maior char do list_chars, aumenta o j
                j+=1

    words.sort()#Ordena as palavras coletadas por ordem lexicográfica

    for word in words:#Mostra as palavras
        print(word)
```

Solução em Java

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;

public class Main {
    /*
    nessa solucao foi utilizada uma funcao recursiva que cria um array com as devidas posicoes
    a funcao eh chamada com os seguintes parametros:
    - a string completa que eh passada no input
    - o indice atual do caracter que controla da onde a chamada recursiva parti ra
    - a string que sera formada e adicionada ao array de combinacoes, comeca igual a "" (string vazia)
    - a lista de combinacoes que sera formada com as respostas
    para facilitar o entendimento da recursao devemos pensar que a string sera iterada caracter a caracter.
    Entao, quando a funcao comeca, a string atual esta vazia, assim nada eh ad
```

```
d no array de combinacoes e vai para o laco.
   Em cada um desses lacos, um caracter eh pego e se chama a funcao recur
sivmanete considerando a concatenacao da string atual + char na posicao se
guinte a analisada.
   */
   static void generateCombinations(String completeString, int currentIndex, S
tring currentString, List<String> combinations) {
      //se a string for vazia, nao tem o que adicionar na lista de combinacoes
      if(!currentString.isEmpty()) {
         combinations.add(currentString);
      }

      //laco que chama a recursao para capturar as combinacoes usando conc
atenacao de char
      for(int i = currentIndex; i < completeString.length(); i++) {
         generateCombinations(completeString, i+1, currentString + completeSt
ring.charAt(i), combinations);
      }
   }



   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      //leio a quantidade de strings que serao colocados no input
      int t = scanner.nextInt();
      //laco para iterar sobre a quantidade de strings que serao fornecidas pel
o usuario
      for(int i = 0; i < t; i++) {
         //leio o tamanho da string que sera colocada no input, mas nao guardo
pois na funcao se toma como condicao de parada o tamanho da string
         scanner.nextInt();
         scanner.nextLine();
         //crio a lista de combinacoes vazia para chamada da recursao
         List<String> combinations = new ArrayList<>();
         //leio a string cujas combinacoes serao calculadas
         String completeString = scanner.next();
```

```
        //chamada da funcao recursiva
        generateCombinations(completeString, 0, "", combinations);
        //Depois de pegar as combinacoes, eu ordeno a lista, pois o desafiuo p
ede
        Collections.sort(combinations);
        //laco para output
        for(String combination : combinations) {
            System.out.println(combination);
        }
    }

  }
}
```

Exercicio 7 - Left Rotate the Array

Solução em Python

```
def leftRotateArray():
    length,shifts = input().split(" ")
    length,shifts = int(length),int(shifts)
    arr = input().split(" ")

    aux = []

    #1) A lógica inicial consiste em guardar os elementos que serão alterados e
m outro array
    #ex:  para a entrada arr = [1,2,3,4,5] , shifts = 2   gero o aux = [1,2]
    for i in range(shifts):
        aux.append(arr[i])

    #2) Em seguida, removemos os elementos coletados do array original
```

```
    #ex:  para arr = [1,2,3,4,5] , aux = [1,2] gera arr = [3,4,5]
    for elm in aux:
        arr.remove(elm)

    #3) Concatenar as duas listas
    #ex:  para arr = [3,4,5] , aux = [1,2] gera arr = [3,4,5,1,2]
    arr.extend(aux)

    s = ''
    for el in arr: #Formatando a saida
        s += el + ' '

    print(s.strip())
```

Solução em Java

```
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        //Criacao de um Scanner para fazer input
        Scanner sc = new Scanner(System.in);
        //Leitura dos valores referentes ao tamanho do array e a quantidade de r
otações
        int arrayLength = sc.nextInt();
        int nRotations = sc.nextInt();
        //Criacao do array para que sera fornecido pelo usuario
        int[] userArray = new int[arrayLength];
        //Laço para input dos elementos do array em sequencia separados por e
spaco
        for(int i = 0; i <= arrayLength - 1; i++){
            userArray[i] = sc.nextInt();
```

```
        }
        //criacao de array auxiliar que sera o resultado apos as rotacoes
        int[] auxArray = new int[arrayLength];
        //percorro o array
        for(int i = 0; i < arrayLength; i++){
            //guardo a posicao atual em uma variavel para conseguir reposiciona-l
a
            int aux = i;
            //laco para fazer as rotacoes
            for(int j = 1; j <= nRotations; j++){
                //caso a posicao do elemento atual for a primeira, quer dizer que ela
ira para a ultima posicao do array
                if(aux == 0){
                    aux = arrayLength - 1;
                } else {
                    //senao a posicao do elemento cai uma casa, ou seja, anda pra es
querda
                    aux--;
                }
            }
            //no final eu coloco o elemento na posicao correta
            auxArray[aux] = userArray[i];
        }
        //por fim, faco um laco para printar o array apos as rotacoes
        for(int i = 0; i <= arrayLength - 1; i++){
            System.out.print(auxArray[i]);
            if(i != arrayLength - 1){
                System.out.print(" ");
            }
        }
    }
}
```

Exercicio 8 - Mr. Zero

Solução em Java

```
import java.util.Scanner;

public class main {
   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);
      // lê a quantidade de linhas e colunas
      int linhas = input.nextInt();
      int colunas = input.nextInt();
      int i, j, aux_i, aux_j, soma = 0, cont = 0;
      // inicializa a matriz com a quantidade de linhas e colunas
      int[][] matriz = new int[linhas][colunas];
      // inicializa um vetor para a quantidade de linhas e um vetor para a quanti
dade de colunas
      boolean[] linhaTemUm = new boolean[linhas];
      boolean[] colunaTemUm = new boolean[colunas];
      // le os valores da matriz
      for (i = 0; i < linhas; i++) {
         for (j = 0; j < colunas; j++) {
            matriz[i][j] = input.nextInt();
         }
      }
      // caso apareça um valor diferente de zero esse indice da linha é marcad
o como true
      for (i = 0; i < linhas; i++) {
         for (j = 0; j < colunas; j++) {
            if (matriz[i][j] != 0) linhaTemUm[i] = true;
         }
      }
      // caso apareca um valor diferente de zero esse indice da linha é marcad
o como falso
```

```
        for (j = 0; j < colunas; j++) {
            for (i = 0; i < linhas; i++) {
                if (matriz[i][j] != 0) colunaTemUm[j] = true;
            }
        }
        // caso o indice da linha e da coluna estejam como false um contador é in
cermentado
        for (i = 0; i < linhas; i++) {
            for (j = 0; j < colunas; j++) {
                if (!linhaTemUm[i] && !colunaTemUm[j]){
                    cont++;
                }
            }
        }

        System.out.println(cont);
    }
}
```

Exercicio 9 - The Universe Loves Minimum Steps

Solução em Java

```
import java.util.Scanner;

public class main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        // captura o número de casos
        int num_casos = input.nextInt(), qtde = 1;

        input.nextLine();
```

```
        while (qtde <= num_casos){
            // captura o tamanho do vetor
            int tamanho_vetor = input.nextInt(), cont, qtde_mov = 0, aux = 1;
            input.nextLine();
            // inicializa o vetor
            int[] vetor = new int[tamanho_vetor];
            // preenche o vetor com os dados
            for (cont = 0; cont < tamanho_vetor; cont++){
                vetor[cont] = input.nextInt();
            }

            while (aux != 0) {
                aux = 0;
                // analisa se o número é maior do que zero, caso seja diminuo em u
ma unidade
                for (cont = 0; cont < tamanho_vetor; cont++) {
                    if (vetor[cont] > 0) {
                        vetor[cont]--;
                        aux++;
                    }
                    // analisa se o número é menor do que zero, caso seja aumento e
m uma unidade
                    else if (vetor[cont] < 0) {
                        vetor[cont]++;
                        aux++;
                    }
                }
                // soma a quantidade de movimentos
                if(aux != 0) qtde_mov++;
            }
            System.out.println("Case " + qtde + ": " + qtde_mov);
            qtde_mov = 0;
            qtde++;
        }
```

```
    }
}
```