SEÇÃO 3 - DESENVOLVIMENTO DE SOFTWARE

(Para quem selecionou as áreas de Frontend e/ou Backend e/ou Mobile)

Questão 10 - Transferência de Strings

Os dados, na forma de uma string binária, devem ser enviados através de dois servidores no HackerLand. Contudo, de acordo com um novo protocolo de controle de rede, os dados só podem ser enviados na forma de strings binárias que não possuem dois caracteres adjacentes iguais. Essas strings binárias sem dois caracteres adjacentes iguais são chamadas de strings especiais. Os dados a serem transmitidos são primeiramente divididos em uma/numerosas subsequências, que são sequências especiais; em seguida, cada sequência especial é enviada como um pacote de dados através dos servidores conectados.
Dada uma string binária que deve ser enviada através de dois servidores, encontrar a quantidade mínima de pacotes de dados em que ela será dividida.

Observação: Uma subsequência de uma string é obtida excluindo alguns caracteres da string ao mesmo tempo em que a ordem é mantida. Por exemplo, “011” é uma subsequência de “0101”, enquanto “100” não.

Exemplo

Considerar input_str = "00100”

A string fornecida pode ser dividida em três subsequências especiais, como indicado abaixo: "0", "010", e "0". Trata-se também da quantidade mínima de subsequências especiais nas quais a string pode ser dividida. Portanto, a saída é 3.

Descrição da Função

Completar a função getMinSubsequences no editor abaixo.

A função getMinSubsequences possui o seguinte parâmetro:
strinput_str: uma string binária

Retornos

int: Quantidade mínima de subsequências nas quais a string pode ser dividida.

Limitações

• $1 \leq$ comprimento de input_str $\leq 10^5$

▼ Formato de Entrada para Testes Personalizados

A linha única contém uma string, input_str.

▼ Exemplo de Caso 0

Exemplo de Entrada para Testes Personalizados

```
STDIN         FUNCTION
-----         --------
1101     →    input_str = "1101"
```

Exemplo de Saída

```
2
```

Explicação

A string pode ser dividida em duas subsequências válidas: "1" e "101".

▼ Exemplo de Caso 1

Exemplo de Entrada para Testes Personalizados

```
STDIN         FUNCTION
-----         --------
11001010  →   input_str = "11001010"
```

Exemplo de Saída

```
2
```

Explicação

As duas subsequências são "101010" e "10".

Questão 11 - Anulação de Array

Dados dois arrays, change e arr, que consistem em números inteiros n e m respectivamente, em uma única operação, qualquer elemento de arr[i] pode ser decrementado em 1.

Na i-ésima operação, o elemento no índice change[i] de arr, ou seja, arr[change[i]] pode ser alterado. Considerar que a indexação começa em 1.

Independentemente do valor de change[i], na I-ésima operação, qualquer elemento pode ser decrementado.
Se change[i] > 0: Se arr[change[i]] = 0, isso pode ser alterado para NULL.

Observe que não é necessário alterar um elemento para NULL caso seja mais ideal fazer isso posteriormente. Você pode optar por não fazer nada ou decrementar qualquer elemento.

Encontrar a quantidade mínima de operações necessárias para alterar todos os elementos do array para NULL ou reportar -1 se não for possível.

Exemplo:
Considerar n = 4 e m = 2
change[] = [0, 1, 0, 2], arr[] = [1, 1]
1. Na primeira operação, arr[1] pode ser decrementado. O array se torna [0, 1].
2. Na segunda operação, visto que change[2] = 1 e arr[1] = 0, o primeiro elemento pode ser alterado para NULL. O array se torna [NULL, 1].
3. Na terceira operação, arr[2] pode ser decrementado. O array se torna [NULL, 0].
4. Na quarta operação, visto que change[4] = 2 e arr[2] = 0, o segundo elemento pode ser alterado para NULL. O array se torna [NULL, NULL].

Trata-se de um ótimo caminho. Retornar a quantidade de operações, 4.

Descrição da Função
Completar a função getMinOperations no editor abaixo.
A função getMinOperations possui o(s) seguinte(s) parâmetro(s):
int change[n]: um array de números inteiros
int arr[m]: um array de números inteiros

Retornos:
int: quantidade mínima de operações necessárias para alterar todos os elementos para NULL

Limitações

• $1 \leq n \leq 10^5$

• $1 \leq m \leq 10^5$

• $0 \leq change[i] \leq m$

• $1 \leq arr[i] \leq 10^5$

▼ Formato de Entrada para Testes Personalizados

A primeira linha contém um número inteiro, n, que denota a quantidade de elementos em change.

Cada linha i das linhas subsequentes n (onde 0 <i<n) contém um número inteiro que denota change[i]. A linha seguinte contém um número inteiro, m, que denota a quantidade de elementos em arr.

Cada linha i das linhas subsequentes m (onde 0 <i<m) contém um número inteiro que denota arr[i].

▼ Exemplo de Caso 0

Exemplo de Entrada para Testes Personalizados

```
STDIN           FUNCTION
-----           --------
10        →     change[] size n = 10
1         →     change = [1, 0, 1, 3, 2, 1, 0, 3, 1, 1]
0
1
3
2
1
0
3
1
1
3         →     arr[] size m = 3
2         →     arr = [2, 1, 2]
1
2
```

Exemplo de Saída

```
8
```

Explicação

Um dos possíveis fluxos de operações pode ser o seguinte:

1. Decrementar arr[1]- [1, 1,2]
2. Decrementar arr[1] - [0, 1, 2]
3. Decrementar arr[2] - [0, 0, 2]
4. Decrementar arr[3] - [0, 0, 1]
5. Alterar arr[2] como change[5] = 2 - [0, NULL, 1]
6. Alterar arr[1] como change[6] = 1 - [NULL, NULL, 1]
7. Decrementar change[3] - [NULL, NULL, 0]
8. Alterar arr[3] como change[8] = 3 - [NULL, NULL, NULL]
9. As operações 8 e 9 não são necessárias.

▼ Exemplo de Caso 1

Exemplo de Entrada para Testes Personalizados

```
STDIN          FUNCTION
-------        ----------
7        →     change[] size n = 7
0        →     change = [0, 0, 0, 2, 1, 3, 2]
0
0
2
1
3
2
3        →     arr[] size m = 3
1        →     arr = [1, 3, 2]
3
2
```

Exemplo de Saída

```
-1
```

Explicação

Não é possível alterar todos os elementos para NULL respeitando as condições fornecidas.

Questão 12 - Nadando pelo Rio

O percurso do rio é representado pela matriz n*m. O rio começa no ponto (0, 0) e tem uma tendência natural de fluir para baixo e para a direita. Existem alguns obstáculos no percurso do rio. As células vazias são representadas por '.' e o obstáculo é representado por '#'. Você está nadando no rio. Movendo-se na direção do fluxo do rio, ou seja, para a direita ou para baixo, não é necessário gastar energia. Contudo, ao mover-se contra o fluxo do rio, ou seja, para a esquerda ou para cima, é necessária 1 unidade de energia.
O objetivo é estar atualmente nas coordenadas iniciais fornecidas e atingir as coordenadas finais com o mínimo gasto de energia possível.
Garante-se que não existem obstáculos nas coordenadas inicial e final.

Determinar a energia mínima possível necessária para chegar ao destino ou determinar se é impossível chegar ao destino.
Retornar -1 caso seja impossível chegar ao destino.

Considerar, por exemplo, n = 4, m = 5, e o rio é retratado por:

.....
...#.
..#.#
.....

Além disso, coordenadas iniciais = (2, 3), coordenadas finais = (1,4) (indexação baseada em 0)
A partir do ponto (2, 3), descer uma unidade, ou seja, até o ponto (3, 3), consumindo 0 unidades de energia.
Depois, mover-se 2 unidades para a esquerda, ou seja, até o ponto (3, 1), consumindo 2 unidades de energia.
Depois, mover-se 3 unidades para cima, ou seja, até o ponto (0, 1), consumindo 3 unidades de energia.
Depois, mover-se 3 unidades para a direita, ou seja, até o ponto (0, 4), consumindo 0 unidades de energia.
Depois, mover-se 1 unidade para baixo, ou seja, até o ponto (1,4, 1), consumindo 0 unidades de energia.
Assim, serão gastas 5 unidades de energia no total.
Pode ser mostrado que a resposta não pode ser inferior a 5. Portanto, a resposta é 5.

Descrição da Função

Completar a função minimumEnergy no editor abaixo. A função deve retornar um número inteiro representando a energia mínima necessária. A função minimumEnergy possui os seguintes parâmetros:
river: um array de strings de comprimento n e cada string tendo um comprimento m, representando o caminho do rio.
initial_x: um número inteiro, representando a coordenada x inicial.
intitial_y: um número inteiro, representando a coordenada y inicial.
final_x: um número inteiro, representando a coordenada x final.
final_y: um número inteiro, representando a coordenada y final.

Limitações

- $1 \leq n \leq 1500$
- $1 \leq m \leq 1500$
- $0 \leq initial\_x, final\_x \leq n$
- $0 \leq initial\_y, final\_y \leq m$

▼ Formato de Entrada para Testes Personalizados

A primeira linha contém um número inteiro, n, que denota a quantidade de linhas em river.
Cada linha i das linhas subsequentes n (onde 0 <i<n) contém uma string (de comprimento m) que descreve river[i].
A linha seguinte contém um número inteiro, initial_x, que denota a coordenada x inicial.
A linha seguinte contém um número inteiro, initial_y, que denota a coordenada y inicial.
A linha seguinte contém um número inteiro, final_x, que denota a coordenada x final.
A última linha contém um número inteiro, final_y, que denota a coordenada y final.

▼ Exemplo Caso 0

Formato de Entrada para Testes Personalizados

```
5
.#..
.##.
....
.#.#
.##.
3
0
0
2
```

Exemplo de Saída

```
4
```

Aqui, n = 3, m = 3 e o rio são retratados por:

.#.

.##

...

Além disso, coordenadas iniciais = (2, 0), coordenadas finais = (0, 2) (indexação baseada em 0)

Pode-se observar, que devido aos obstáculos, é impossível chegar às coordenadas finais. Portanto, a resposta é -1.

FIM DA SEÇÃO DE DESENVOLVIMENTO DE SOFTWARE