BairesDev

Prueba Técnica - Razonamiento lógico

En **BairesDev**, reconocemos la importancia del razonamiento lógico y la habilidad para resolver desafíos en el éxito de nuestro equipo. Para evaluar tus aptitudes en la resolución de problemas, hemos diseñado esta prueba de ingreso. Es esencial que comprendas y sigas las siguientes reglas para llevar a cabo esta prueba de manera adecuada:

Compromiso durante la Prueba: Una vez que comiences la prueba, te pedimos que permanezcas en tu puesto. Si te levantas de tu lugar, consideraremos que la prueba ha finalizado. Esta regla se establece para mantener un ambiente de igualdad y equidad entre los participantes.

Independencia en la Resolución: Durante la prueba, no está permitido preguntar a tus compañeros. Valoramos tu capacidad para abordar los desafíos de manera individual y confiamos en tu habilidad para encontrar soluciones por tu cuenta. Esta regla busca evaluar tu capacidad de resolución autónoma y tu destreza en situaciones desafiantes.

Ten en cuenta que esta prueba es una oportunidad para demostrar tus capacidades y habilidades en la resolución de problemas, que son cualidades fundamentales para tener éxito en **BairesDev**. ¡Te deseamos mucha suerte y estamos emocionados por conocer tus habilidades!

Reto 1: Fibonacci

Los números de Fibonacci F_K son una sucesión de números naturales definidos de la siguiente manera:

$$
\begin{aligned}
F_0 &= 0, \\
F_1 &= 1, \\
F_k &= F_{k-1} + F_{k-2}, \qquad \text{cuando } k \geq 2.
\end{aligned}
$$

En palabras simples, la sucesión de Fibonacci comienza con 0 y 1, y los siguientes términos siempre son la suma de los dos anteriores.

En la siguiente tabla, podemos ver los números de Fibonacci desde el 0-ésimo hasta el duodécimo.

n	0	1	2	3	4	5	6	7	8	9	10	11	12	...
F_n	0	1	1	2	3	5	8	13	21	34	55	89	144	...

1. Escriba un programa que reciba como entrada un número entero *n,* y entregue como salida el *n-ésimo* número de Fibonacci:

```
Ingrese n: 11
F11 = 89
```

2. Escriba un programa que reciba como entrada un número entero e indique si es o no un número de Fibonacci:

```
Ingrese un numero: 34
34 es numero de Fibonacci
```

```
Ingrese un numero: 78
78 no es numero de Fibonacci
```

3. Escriba un programa que muestres los *m* primeros números de Fibonacci, donde *m* es un número ingresado por el usuario:

```
Ingrese m: 7
Los 7 primeros numeros de Fibonacci son:
0 1 1 2 3 5 8
```

Reto 2: Multiplicación Rusa

El método de multiplicación rusa consiste en multiplicar sucesivamente por 2 el multiplicando y dividir por 2 el multiplicador hasta que el multiplicador tome el valor 1.

Luego, se suman todos los multiplicandos correspondientes a los multiplicadores impares.

Dicha suma es el producto de los dos números. La siguiente tabla muestra el cálculo realizado para multiplicar 37 por 12, cuyo resultado final es 12 + 48 + 384 = 444.

Multiplicador	Multiplicando	Multiplicador impar	Suma
37	12	sí	12
18	24	no	
9	48	sí	60
4	96	no	
2	192	no	
1	384	sí	444

Desarrolle un programa que reciba como entrada el multiplicador y el multiplicando, y entregue como resultado el producto de ambos, calculado mediante el método de multiplicación rusa.

```
Ingrese multiplicador: 37
Ingrese multiplicando: 12
Resultado: 444
```

BairesDev

Reto 3: Números amistosos

Un par de números m y n son llamados amistosos (o se conocen como un par amigable), si la suma de todos los divisores de m (excluyendo a m) es igual al número n, y la suma de todos los divisores del número n (excluyendo a n) es igual a m (con m ≠ n).

Por ejemplo, los números 220 y 284 son un par amigable porque los únicos números que dividen de forma exacta 220 son 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 y 110, y 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110 = 284

Por lo tanto, 220 es un número amistoso. Los únicos números que dividen exactamente 284 son 1, 2, 4, 71 y 142 y 1 + 2 + 4 + 71 + 142 = 220

Por lo tanto, 284 es un número amistoso.

Muchos pares de números amigables son conocidos; sin embargo, sólo uno de los pares (220, 284) tiene valores menores que 1000. El siguiente par está en el rango [1000, 1500].

Desarrolle un programa que permita encontrar dicho par.

Reto 4: Votaciones de la CONFECH

La CONFECH, en su afán de agilizar el proceso de recuento de las votaciones, le ha encargado el desarrollo de un programa de registro de votación por universidades.

Primero, el programa debe solicitar al usuario ingresar la cantidad de universidades que participan en el proceso.

Luego, para cada una de las universidades, el usuario debe ingresar el nombre de la universidad y los votos de sus alumnos, que pueden ser: aceptar (A), rechazar (R), nulo (N) o blanco (B). El término de la votación se indica ingresando una X, tras lo cual se debe mostrar los totales de votos de la universidad, con el formato que se muestra en el ejemplo.

Finalmente, el programa debe mostrar el resultado de la votación, indicando la cantidad de universidades que aceptan, que rechazan y en las que hubo empate entre estas dos opciones.

```
Numero de universidades: 3

Universidad: USM
Voto: A
Voto: R
Voto: A
Voto: N
Voto: X
USM: 2 aceptan, 1 rechazan, 0 blancos, 1 nulos.

Universidad: UChile
Voto: A
Voto: B
Voto: A
Voto: X
UChile: 2 aceptan, 0 rechazan, 1 blancos, 0 nulos.

Universidad: PUC
Voto: A
Voto: R
Voto: R
Voto: A
Voto: X
PUC: 2 aceptan, 2 rechazan, 0 blancos, 0 nulos.

Universidades que aceptan: 2
Universidades que rechazan: 0
Universidades con empate: 1
```

Reto 5: Question Description

Sam and Kelly are programming buddies. Kelly resolves to practice more as Sam isa head initially. They each solve a number of problems daily. Find the mínimum number or days for Kelly to have solved more problems than Sam. If Kelly cannot surpass retum -1.

Example

SamDaily = 3

KellyDaily = 5

Difference = 5

Initially, Sam has solved difference problems more than Kelly. Each day, they solve *samDaily* and *kellyDaily* problems each.

Day 1: *samSolved = difference + samDaily* = 5 + 3 = 8

kellySolved = kellyDaily = 5

Day 2: *samSolved* = 8 + 3 = 11

k*ellySolved = 5 + 5 = 10*

Day 3: *samSolved = 11 + 3 = 14*

kellySolved = 10 + 5 = 15

Sam is 5 problems ahead of Kelly and they solve 3 and 5 problems a day. Sam will be ahead by only 3 after the first day, 1 after the second, and Kelly will pass Sam on day 3.

Function Description

Complete the function *minNum* in the editor below.

MinNum has the following parameter(s):

SamDaily: Number of problems Sam solves in a day

KellyDaily: Number of problems Kelly solves in a day

Difference: Number of problems Sam isa head to begin

Return

Int: the minimum number of days needed by Kelly to exceed Samm, or -1 if it is imposible

Constraints

- $1 \leq samDaily, kellyDaily \leq 100$
- $0 \leq difference \leq 100$

Input format for Custom Testing

Input from stdin will be processed as follows and passed to the fuction.

The first line contains an integer *samDaily.*

The second line contains an integer *kellyDaily.*

The third line contains an integer *ahead.*

Sample Case 0

Sample Input 0

```
STDIN       Function
-----       --------
3     →     samDaily = 3
5     →     kellyDaily = 5
1     →     difference = 1
```

Sample Output 0

1

Sample Case 1

Sample Input 1

```
STDIN       Function
-----       --------
4     →     samDaily = 4
5     →     kellyDaily = 5
1     →     difference = 1
```

Sample Output 1

2

QUESTION DESCRIPTION

Consider every susequence of an array of integers.

- Sort the subsequence in increasing order.
- Determine the sum of differences of elements in the subsequence.
- Return the length of the longest subsequence where this sum is even.

Example

Given *n = 4* elements and *arr = [2, 4, 1, 7]*, these are some of the subsequences.

Subsequence	Sorted Subsequence	Sum of diff of Adjacent elements	Is Valid	Length
[2, 4, 1]	[1, 2, 4]	1 + 2 = 3 (Odd)	No	3
[2, 1, 7]	[1, 2, 7]	1 + 5 = 6 (Even)	Yes	3
[2, 4, 1, 7]	[1, 2, 4, 7]	1 + 2 + 3 = 6 (Even)	Yes	4
[2, 1]	[1, 2]	1 (Odd)	No	2

We can see that the máximum posible length of a valid subsequence is 4.

BairesDev

Function Description

Complete the function *findLongestSubsequence* in the editor below.

FindLongestSubsequence has the following parameter(s):

Int arr[n]: an array of integers

Returns

Int: the length of the longest subsquence as describe

Constraints

- $3 \leq n \leq 10^5$
- $0 \leq arr[i] \leq 10^9$

Input Format For Custom Testing

The first line contains an integer, *n,* the number of elements in *arr.*

Each line *i* of the *n* subsequent lines ($where\ 0 \leq i < n$) contaisn an integer, *arr[i]*

Sample Case 0

Sample Input For Custom Testing

```
STDIN       FUNCTION
-----       ----------
7       →   arr size [] n = 7
7       →   arr = [7, 5, 6, 2, 3, 2, 4]
```

Sample Case 1

Sample Input For Custom Testing

```
STDIN       FUNCTION
-----       ----------
4       →   arr size[] n = 4
1       →   arr = [1, 3, 5, 7]
3
5
7
```

Sample Output

```
4
```

Explanation

The entire array can be used.

- arrange the subsequence in ascending order, 1,3,5,7
- the adjacent differences are 2, 2, 2
- 2+2+2 = 6

BairesDev