URL VIDEOS

Explicación enunciado del problema: https://youtu.be/MiLk8VOSxMo?feature=shared
Explicación código: https://youtu.be/0gq2ngIcWjU?feature=shared
Análisis de complejidad: https://youtu.be/owKZcZnTRJM?feature=shared

ENUNCIADO DEL PROBLEMA

Takahashi is standing on a multiplication table with infinitely many rows and columns.

the square (i,j) contains the integer i×j. Initially, Takahashi is standing at (1,1), In one move, he can move from (i,j) to either (i+1,j) or (i,j+1).

Given an integer *N*, find the minimum number of moves needed to reach a square that contains *N*.

Constraints

- 2≤N≤10^12
- *N* is an integer.

Input

Input is given from Standard Input in the following format: N

Output

Print the minimum number of moves needed to reach a square that contains the integer N.

URL DE GITLAB O GITHUB PROYECTO EN FORMA PRIVADA, USUARIO madarme@ufps.edu.co, con rol maintener

https://gitlab.com/ejercicios-proyecto-ada/ejercicio-5-walk-on-a-multiplication-table

MÉTODO 1

```
public long metodoCandido(long n) {
    long minMoves = Long.MAX_VALUE;
    for(int i = 1; i<= n; i++){
        for(int j = 1; j<= n; j++){
            long producto = i*j;
            if(producto == n){
                long moves = (i-1) + (j-1);
                if(moves < minMoves){
                    minMoves = moves;
                }
            }
        }
    }
    return minMoves;
}
```

Eficacia	El método es eficaz para las entradas de números enteros, en entradas mas grandes tipo Long el método no es eficaz
	Instancia 1: Número: 10 Resultado de la invocación: 5

	Instancia 2: Número: 9146 Resultado de la invocación: 301
	Instancia 3: Numero: 10000000019L Resultado de la invocación: El método no devuelve una salida
Eficiencia	El método en entradas pequeñas es eficiente, sin embargo no lo suficiente ya que itera dos veces desde 1 hasta n y en entradas grandes no es eficiente, ya que le toma demasiado tiempo dar una salida.
	Instancia 1: Número: 10 Resultado de la invocación: 5 **Tiempo del proceso: 157 microsegundos**
	Instancia 2: Número: 9146 Resultado de la invocación: 301 **Tiempo del proceso: 58691 microsegundos**
	Instancia 3: Numero: 10000000019L

	Resultado de la invocación: No devuelve salida **Tiempo del proceso: No termina de ejecutarse el método**
Correctitud	El algoritmo no es correcto ya que no da el resultado esperado para todas las entradas posibles.
	Instancia 1: Número: 10 Resultado esperado: 5 Resultado método candido: 5
	Instancia 2: Número: 9146 Resultado esperado: 301 Resultado método candido: 301
	Instancia 3: Numero: 10000000019L Resultado esperado: 10000000018 Resultado método cándido: No da la respuesta
Completitud	El método no es completo ya que aunque la lógica sirve para hallar el resultado, es demasiado lento para producir el resultado esperado.
	Instancia 1:

	Número: 10 Resultado esperado: 5 Resultado método candido: 5
	Instancia 2: Número: 9146 Resultado esperado: 301 Resultado método candido: 301
	Instancia 3: Numero: 10000000019L Resultado esperado: 10000000018 Resultado método cándido: No da la respuesta

MÉTODO 2	
`` public long metodoOptimo(long n) { long mayor = 0; for(long i = 1; i*i <= n; i++){ if(n%i==0){ mayor = i; } } return mayor + ((n/mayor)-2); } ```	
Eficacia	El método es eficaz para todas las entradas posibles.
	Instancia 1: Número: 10 Resultado de la invocación: 5
	Instancia 2:

	Número: 9146 Resultado de la invocación: 301
	Instancia 3: Numero: 10000000019L Resultado de la invocación: 10000000018
Eficiencia	El método es eficiente ya que solo recorre hasta raíz de n para hallar los divisores, lo hace en muy poco tiempo.
	Instancia 1: Número: 10 Resultado de la invocación: 5 **Tiempo del proceso: 43 microsegundos**
	Instancia 2: Número: 9146 Resultado de la invocación: 301 **Tiempo del proceso: 25 microsegundos**
	Instancia 3: Numero: 10000000019L Resultado de la invocación: 10000000018 **Tiempo del proceso: 1510 microsegundos**
Correctitud	El método devuelve salidas esperadas para todas las

	entradas posibles.
	Instancia 1: Número: 10 Resultado esperado: 5 Resultado método óptimo: 5
	Instancia 2: Número: 9146 Resultado esperado: 301 Resultado método óptimo: 301
	Instancia 3: Número: 10000000019L Resultado esperado: 10000000018 Resultado método óptimo: 10000000018
Completitud	El método es completo ya que con la lógica usada se asegura una respuesta correcta si existe.
	Instancia 1: Número: 10 Resultado método óptimo: 5
	Instancia 2: Número: 9146

	Resultado método óptimo: 301
	Instancia 3: Número: 10000000019L Resultado método óptimo: 10000000018