URL VIDEOS

Explicación enunciado del problema:https://youtu.be/LN6OqNs6CZo
Explicación código: https://youtu.be/C2i2wyRmq8Y
Análisis de complejidad: https://youtu.be/rfOCgVPXoYw

ENUNCIADO DEL PROBLEMA

¡A Pasha le encantan los números primos! Una vez más, en sus intentos por encontrar una nueva forma de generar números primos, se interesó por un algoritmo que encontró en Internet:
Para obtener un nuevo número y, repite k veces la representación decimal del número x (sin ceros a la izquierda).
Por ejemplo, para x=52 y k=3, obtenemos y=525252, y para x=6 y k=7, obtenemos y=666666666.
Pasha realmente quiere que el número resultante y sea primo, pero aún no sabe cómo comprobar la primalidad de los números generados por este algoritmo. ¡Ayuda a Pasha y dile si y es primo!

**Un número entero x se considera primo si tiene exactamente 2 divisores distintos: 1 y x. Por ejemplo, 13 es primo porque sólo tiene 2 divisores: 1 y 13. Tenga en cuenta que el número 1 no es primo, ya que sólo tiene un divisor.

INPUT
La primera y única línea de cada conjunto de datos contiene dos enteros: x y k (1≤x≤10^9, 1≤k≤7).

OUTPUT

Para cada conjunto de datos de entrada, indique «SÍ» (sin comillas) si el número resultante y será primo, y «NO» en caso contrario.

URL DE GITLAB

https://gitlab.com/ejercicios-proyecto-ada/ejercicio-7-simple-repetition

MÉTODO CÁNDIDO

```java
public boolean metodoCandido(int x, int k) {
    boolean primo = true;
    StringBuilder numeroStr = new StringBuilder();
    for (int i = 0; i < k; i++) {
        numeroStr.append(x);
    }
    long numUnido = Long.parseLong(numeroStr.toString());

    if (numUnido <= 1) return false;

    for(long i = 2; i<numUnido; i++) {
        if (numUnido%i == 0) {
            primo = false;
        }
    }

    return primo;
}
```

Eficacia	El método es eficaz, puesto a que tiene una solución para todas las entradas válidas ingresadas.
	Instancia 1: x: 1 k: 2 RESULTADO MÉTODO CÁNDIDO: YES
	Instancia 2: x: 9 k: 7 RESULTADO MÉTODO CÁNDIDO: NO
	Instancia 3: x: 999999937 k: 1 RESULTADO MÉTODO CÁNDIDO: YES
Eficiencia	El método no es el más eficiente ya que primero concatena k veces el número x y luego verifica los divisores yendo hasta el número unido - 1, por lo que en casos extremos tomaría mucho tiempo en verificar si es o no primo.
	Instancia 1:

	x: 1 k: 2 RESULTADO MÉTODO CÁNDIDO: YES **Tiempo del proceso: 82 microsegundos**
	Instancia 2: x: 9 k: 7 RESULTADO MÉTODO CÁNDIDO: NO **Tiempo del proceso: 74840 microsegundos**
	Instancia 3: x: 999999937 k: 1 RESULTADO MÉTODO CÁNDIDO: YES **Tiempo del proceso: 7309205 microsegundos**
Correctitud	El algoritmo es correcto, ya que da un resultado esperado para todas las entradas posibles
	Instancia 1: x: 1

	k: 2 RESPUESTA ESPERADA: YES RESULTADO MÉTODO CÁNDIDO: YES
	Instancia 2: x: 9 k: 7 RESPUESTA ESPERADA: NO RESULTADO MÉTODO CÁNDIDO: NO
	Instancia 3: x: 999999937 k: 1 RESULTADO ESPERADO: YES RESULTADO MÉTODO CÁNDIDO: YES
Completitud	El método es completo, ya que la lógica usada para verificar si el número es primo o no es correcta, aunque es lento asegura siempre un resultado correcto, si es que este existe
	Instancia 1: x: 1

	k: 2 RESPUESTA ESPERADA: YES RESULTADO MÉTODO CÁNDIDO: YES
	Instancia 2: x: 9 k: 7 RESPUESTA ESPERADA: NO RESULTADO MÉTODO CÁNDIDO: NO
	Instancia 3: x: 999999937 k: 1 RESULTADO ESPERADO: YES RESULTADO MÉTODO CÁNDIDO: YES

MÉTODO 2

```java
public boolean metodoOptimo(int x, int k) {
    if (x==1){
        return k == 2;
    }

    if (k>1) {
        return false;
    } else {
        for(int i = 2; i*i<= x; i++) {
            if (x%i == 0) {
                return false;
            }
        }
        return true;
    }
}
```

Eficacia	El método es eficaz, puesto a que tiene una solución para todas las entradas válidas ingresadas
	Instancia 1:

	x: 1 k: 2 RESULTADO MÉTODO ÓPTIMO: YES
	Instancia 2: x: 9 k: 7 RESULTADO MÉTODO ÓPTIMO: NO
	Instancia 3: x: 999999937 k: 1 RESULTADO MÉTODO ÓPTIMO: YES
Eficiencia	El método es eficiente ya que no tiene que concatenar el número y verifica los divisores hasta la raíz del número, por lo que es muy eficiente.
	Instancia 1: x: 1 k: 2

	RESULTADO MÉTODO ÓPTIMO: YES **Tiempo del proceso: 40 microsegundo**
	Instancia 2: x: 9 k: 7 RESULTADO MÉTODO ÓPTIMO: NO **Tiempo del proceso: 17 microsegundos**
	Instancia 3: x: 999999937 k: 1 RESULTADO MÉTODO ÓPTIMO: YES **Tiempo del proceso: 796 microsegundos**
Correctitud	El algoritmo es correcto, ya que da un resultado esperado para todas las entradas posibles
	Instancia 1: x: 1 k: 2 RESULTADO ESPERADO: YES

	RESULTADO MÉTODO ÓPTIMO: YES
	Instancia 2: x: 9 k: 7 RESULTADO ESPERADO: NO RESULTADO MÉTODO ÓPTIMO: NO
	Instancia 3: x: 999999937 k: 1 RESULTADO ESPERADO: YES RESULTADO MÉTODO ÓPTIMO: YES
Completitud	El método es completo, ya que la lógica usada para verificar si el número es primo o no es correcta, es rápido y asegura siempre un resultado correcto, si es que este existe
	Instancia 1: x: 1 k: 2 RESULTADO ESPERADO: YES

	RESULTADO MÉTODO ÓPTIMO: YES
	Instancia 2: x: 9 k: 7 RESULTADO ESPERADO: NO RESULTADO MÉTODO ÓPTIMO: NO
	Instancia 3: x: 999999937 k: 1 RESULTADO ESPERADO: YES RESULTADO MÉTODO ÓPTIMO: YES