CÃ³digo escrito para Kotlin con el objetivo de Encontrar NÃºmeros Primos

DescripciÃ³n

Este ejemplo utiliza el **Algoritmo de la Criba de EratÃ³stenes**, que es un mÃ©todo eficiente para encontrar todos los nÃºmeros primos menores o iguales a un nÃºmero dado.

default watermark

```
fun encontrarNumerosPrimos(limite: Int): List<Int> {
    // Crear una lista booleana para marcar los nÃºmeros primos
    val esPrimo = BooleanArray(limite + 1) { true }

    // Eliminar los nÃºmeros que no son primos
    for (i in 2..Math.sqrt(limite.toDouble()).toInt()) {
        if (esPrimo[i]) {
            for (j in i * i..limite step i) {
                esPrimo[j] = false
            }
        }
    }

    // Crear una lista con los nÃºmeros primos encontrados
    val numerosPrimos = mutableListOf<Int>()
    for (i in 2..limite) {
        if (esPrimo[i]) {
            numerosPrimos.add(i)
        }
    }

    return numerosPrimos
}

fun main() {
    val limite = 50 // Define el lÃmite superior para la bÃºsqueda de nÃºmeros
    val primos = encontrarNumerosPrimos(limite)
```

```
    println("NÃºmeros primos hasta $limite: $primos")
}
```

ExplicaciÃ³n del CÃ³digo

1. **InicializaciÃ³n del Array `esPrimo`**:
   - Se crea un array booleano llamado `esPrimo` de tamaÃ±o `limite + 1`, donde cada elemento se inicializa en `true`. Este array se utiliza para marcar si un nÃºmero es primo o no.
2. **Criba de EratÃ³stenes**:
   - El bucle `for` externo recorre los nÃºmeros desde 2 hasta la raÃz cuadrada del lÃmite (`sqrt(limite)`), ya que cualquier nÃºmero mayor que su raÃz cuadrada ya habrÃa sido marcado si no es primo.
   - Si `esPrimo[i]` es `true`, significa que `i` es un nÃºmero primo.
   - Luego, otro bucle interno marca todos los mÃºltiplos de `i` como `false` (no primos) empezando desde `i * i` hasta el lÃmite (`limite`).
3. **Crear la Lista de NÃºmeros Primos**:
   - Se recorre nuevamente el array `esPrimo` desde 2 hasta el `limite`. Todos los Ãndices marcados como `true` son nÃºmeros primos, y se aÃ±aden a la lista `numerosPrimos`.
4. **FunciÃ³n `main`**:
   - En la funciÃ³n `main`, se define el lÃmite hasta donde queremos buscar nÃºmeros primos (`limite = 50` en este caso).
   - Llama a la funciÃ³n `encontrarNumerosPrimos` y muestra la lista de nÃºmeros primos encontrados.

default watermark

Salida del CÃ³digo

Para un lÃmite de 50, la salida serÃa:

less

Copiar cÃ³digo

```
NÃºmeros primos hasta 50: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
```

Este cÃ³digo te proporciona una forma eficiente y sencilla de encontrar todos los nÃºmeros primos dentro de un rango especÃfico utilizando Kotlin. Puedes modificar el valor de `limite` en la funciÃ³n `main` para buscar nÃºmeros primos en diferentes rangos.

CategorÃa

1. Kotlin

Etiquetas

1. kotlin

Fecha de creaciÃ³n
01/09/2024
Autor
admin

default watermark