Logotipo

Descripción generada automáticamente con confianza mediaLogotipo

Descripción generada automáticamenteINSTITUTO POLITECNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

ANÁLISIS Y DISEÑO DE ALGORITMOS

PRÁCTICA 05

Rodrigo García Mayorga

2012630554

# Introducción

Esta práctica implementa una interesante combinación de conceptos matemáticos y algorítmicos, trabajando con la secuencia de Fibonacci y números primos, utilizando un enfoque voraz (greedy) para resolver un problema de suma.

**Secuencia de Fibonacci**

La secuencia de Fibonacci es una serie de números donde cada número es la suma de los dos anteriores:

* F(0) = 0
* F(1) = 1
* F(n) = F(n-1) + F(n-2) para n > 1

Por ejemplo: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

**Números Primos**

Un número primo es aquel que solo es divisible por 1 y por sí mismo. En esta práctica, se utiliza una función esPrimo() para identificar números primos mediante un algoritmo de verificación optimizado.

**Algoritmo Voraz (Greedy)**

El programa utiliza un enfoque voraz para encontrar una combinación de números de Fibonacci que sumen un valor K dado. Las características principales del algoritmo son:

1. Genera una secuencia de Fibonacci filtrada (excluyendo posiciones que son números primos)
2. Utiliza la estrategia de seleccionar siempre el número más grande posible
3. Va restando estos números de K hasta llegar a 0

# Desarrollo

**Particularidades del Código**

El programa:

* Utiliza la fecha de nacimiento para generar el valor K
* Implementa un array para almacenar la secuencia de Fibonacci
* Maneja números grandes con el tipo long long
* Muestra los términos utilizados y el total de términos necesarios para la suma

Esta implementación demuestra cómo los algoritmos voraces pueden proporcionar soluciones eficientes para problemas de optimización combinatoria.

**Estructura del Programa**

El programa está compuesto por tres componentes principales:

1. **Función esPrimo()**
   * Verifica si un número es primo
   * Utiliza un algoritmo optimizado verificando hasta la raíz cuadrada del número
   * Retorna true si el número es primo, false en caso contrario
2. **Función encontrarTerminosFibonacci()**
   * Recibe un parámetro K que representa el número objetivo
   * Genera una secuencia de Fibonacci filtrada
   * Implementa un algoritmo voraz para encontrar la suma
   * Características principales:
     + Usa un array fib[] para almacenar la secuencia
     + Excluye posiciones que corresponden a números primos
     + Almacena números usando long long para manejar valores grandes
3. **Función main()**
   * Inicializa variables con la fecha de nacimiento:

dia = 17

mes = 01

año = 93

* + Calcula K usando la fórmula: K = dia \* 100 + mes \* 10 + año
  + Llama a la función encontrarTerminosFibonacci()

**Algoritmo de Solución**

1. **Generación de Secuencia**
   * Inicia con los valores base [0, 1, 1]
   * En cada iteración:
     + Verifica si la posición actual es prima
     + Si no es prima, genera el siguiente número de Fibonacci
     + Continúa hasta superar el valor de K
2. **Búsqueda de Solución**
   * Utiliza un enfoque voraz (greedy)
   * Recorre la secuencia de mayor a menor
   * Selecciona números que pueden restarse de K
   * Cuenta el total de términos utilizados

**Salida del Programa**

El programa muestra:

* El valor de K calculado
* Los términos de Fibonacci utilizados en la suma
* El total de términos necesarios

**Complejidad**

* Verificación de primos: O(√n)
* Generación de secuencia: O(K)
* Búsqueda de solución: O(log K)

Este programa demuestra una implementación eficiente que combina conceptos de teoría de números con algoritmos voraces para resolver un problema de suma específico.

**Texto

El contenido generado por IA puede ser incorrecto.**

## Enlace del código (github):

<https://github.com/LowisN/ADA_Practica5>