

ALGORITMOS: AYUDANTÍA 1

Ayudante: Yerko Ortiz

Objetivo: entender mejor conceptos iniciales del curso y soltar la mano con Java.

1. Introducción

1.1. Algunos Conceptos

Defina los siguientes conceptos con sus propias palabras:

- Algoritmo:
- Estructura de Datos:
- Recursividad:
- Iteración:
- Diseño:
- Implementación:
- Complejidad Computacional:

2. Análisis

Usando un lápiz y papel como pantalla y su imaginación como unidad de procesamiento, ejecute a mano los siguientes algoritmos con las entradas que se indican en cada uno.

2.1. Selection Sort

```
void sort(int arr[])
{
    int n = arr.length;
    for (int i = 0; i < n-1; i++) {
        int min_idx = i;
        for (int j = i+1; j < n; j++)
            if (arr[j] < arr[min_idx]) min_idx = j;
        int temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
    }
}
```

Input: 2, 3, 1, 5, 6, 7

Output:

2.2. Máximo Común Divisor

```
static int gcd(int a, int b)
{
    if (b == 0) return a;
    return gcd(b, a % b);
}
```

Input: 123, 51

Output:

3. Diseño e Implementación

Diseñe e implemente algoritmos en Java para los siguientes problemas:

3.1. Conjetura de Collatz

Considere un algoritmo que recibe como entrada un número entero positivo n ; si n es par entonces el algoritmo divide n por dos, para el caso complementario (que n sea impar) el algoritmo multiplica n por tres y luego le suma uno. El algoritmo repite este proceso hasta que n sea 1.

Por ejemplo la secuencia con $n = 3$ es:

$3 \rightarrow 10 \rightarrow 5 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$

Diseñe e implemente un algoritmo que pueda simular el algoritmo descrito.

Input 1: 3

Output 1: 3 10 5 16 8 4 2 1

3.2. Traducción Decimal a Binario

En el sistema numérico decimal representamos una cantidad haciendo uso de 10 símbolos $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, por ejemplo el valor tres lo representamos con el dígito 3; el sistema binario por otro lado hace uso de solo dos dígitos $\{0, 1\}$, donde el valor tres se representa con el bitstring 10.

La codificación del sistema binario representa una suma de potencias de dos multiplicadas por el dígito que le corresponde:

$$10 = 1 \times 2^1 + 0 \times 2^0$$

$$111 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$1010 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

Es posible generalizar y decir que cualquier entero n positivo de base 10 se puede escribir de la forma:

$$\sum_{i=0}^L a_i 2^i = n$$

Donde L es el largo del bitstring y a_i es el símbolo que corresponde a la i -ésima posición del bitstring leído de derecha a izquierda.

Diseñe e implemente un algoritmo que transforme un entero positivo n en su representación binaria.

Input 1: $n = 12$

Output 1: 1100

Input 2: $n = 4$

Output 2: 100

3.3. Gray Codes

Se empieza con esta secuencia de enteros representados en binario:

0

1

—

Luego la secuencia continua así:

00

01

11

10

—

Posteriormente continua así:

000
001
011
010
110
111
101
100

Estas secuencias llevan el nombre de Reflected Gray Codes para 1 bit, 2 bits y 3 bits respectivamente. Es posible observar que en cada secuencia cambia uno y solo uno de los bits en el bitstring, también es importante notar que una secuencia arbitraria de n bits tendrá 2^n elementos en total.

Diseñe e implemente un algoritmo que dado un entero n positivo de base 10, imprima su posición en una secuencia de Reflected Gray Codes.

Ejemplo: en la primera secuencia se puede notar que $n = 1$ (1) está en la posición 2, en la segunda secuencia es posible notar que $n = 3$ (10) se encuentra en la posición 4, en la tercera secuencia es posible notar que el $n = 7$ (111) se encuentra en la posición 6.

Input 1: 3

Output 1: 4

Input 2: 7

Output 2: 6

Input 3: 12

Output 3: 9

3.4. Hints para el curso

- D. Knuth en su libro The Art Of Computer Programming menciona que la mejor forma de aprender un algoritmo nuevo es ejecutarlo con lápiz y papel; Es decir que antes de implementar y usar un algoritmo que desconoce, haga el ejercicio al menos una vez de ejecutarlo a mano, de esa forma lo podrá interiorizar.
- Resolver un problema nuevo la mayoría de las veces es difícil, no busque soluciones de otras personas hasta que no le haya dedicado un tiempo razonable!
- La solución a un problema de algoritmos se puede diseñar e implementar de diversas formas, tómese el tiempo de discutir su solución con la de sus compañeros; Esto le otorgará nuevas perspectivas del problema.
- Antes de implementar una solución es importante que la diseñe bien, no programe nada hasta que este seguro de que el diseño es correcto; Posteriormente no se quede solo con el diseño, implemente su solución y verifique que funciona experimentalmente.

4. Edsger Dijkstra



"Computer Science is no more about computers than astronomy is about telescopes." - Edsger W. Dijkstra