



UNIVERSIDAD TECMILENIO

Actividad 3



Nombre: Luis Fernando Núñez Díaz

Maestro: Adalberto Emmanuel Rojas Perea

Materia: Estructuras de Datos

Serie de Fibonacci

Diseño del Algoritmo

El programa solicita al usuario un número entero n que representa la cantidad de elementos de la serie de Fibonacci que se desean imprimir. En lugar de utilizar un ciclo `for`, se emplea una función recursiva llamada `mostrar`, que recibe tres parámetros:

- a : el primer número de la serie (inicialmente 0)
- b : el segundo número de la serie (inicialmente 1)
- n : el número de elementos restantes por imprimir

Implementación

La función `mostrar` imprime el valor actual de a y luego se llama a sí misma con los siguientes valores de la serie (b y $a + b$) y con $n - 1$ como nuevo contador. El caso base se alcanza cuando $n == 0$, momento en el cual la recursión se detiene, este enfoque evita el uso de estructuras iterativas.

Funcionamiento del Algoritmo

```
Serie de Fibonacci  
Introduce el número de elementos de la serie: 10  
0, 1, 1, 2, 3, 5, 8, 13, 21, 34,
```

Suma de Subconjuntos

Diseño del Algoritmo

Este programa resuelve el clásico problema de la suma de subconjuntos (Subset Sum), que consiste en determinar si existe un subconjunto dentro de un conjunto de enteros que sume exactamente un valor objetivo.

La función `subsetSum` se implementa de forma recursiva y recibe tres parámetros:

- `nums`: el arreglo de enteros
- `n`: la cantidad de elementos disponibles para considerar
- `valor`: el valor objetivo que se desea alcanzar

Implementación

La función evalúa dos casos base:

- Si `valor == 0`, se ha encontrado un subconjunto válido.
- Si no quedan elementos (`n == 0`) y el valor aún no se ha alcanzado, no existe solución.

Luego, se aplican dos caminos recursivos:

1. Excluir el último número del conjunto.
2. Incluir el último número y restarlo del valor objetivo.

Este enfoque divide el problema en subproblemas más pequeños, explorando todas las combinaciones posibles de inclusión/exclusión.

Funcionamiento del Algoritmo

```
Introduce el valor objetivo: 9
Sí existe un subconjunto con suma 9
```

Sudoku con Backtracking

Diseño del Algoritmo

Este programa resuelve un tablero de Sudoku utilizando el algoritmo de backtracking. El tablero se representa como una matriz de 9x9, donde los ceros indican celdas vacías.

El algoritmo se basa en dos funciones principales:

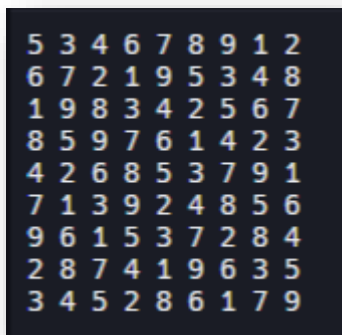
- **esSeguro:** Verifica si un número puede colocarse en una celda sin violar las reglas del Sudoku (sin repetir en fila, columna o subcuadro 3x3).
- **resolverSudoku:** Recorre el tablero celda por celda, intentando colocar números válidos. Si se encuentra un conflicto, retrocede (backtracking) y prueba otra opción.

Implementación

El algoritmo sigue estos pasos:

1. Avanza por el tablero buscando una celda vacía.
2. Intenta colocar un número del 1 al 9 que sea válido.
3. Si encuentra un número válido, lo coloca y continúa con la siguiente celda.
4. Si no hay opciones válidas, retrocede a la celda anterior y prueba otro número.
5. El proceso continúa hasta que el tablero esté completo o se determine que no hay solución.

Funcionamiento del Algoritmo



5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Reflexión

La recursividad y el paradigma de divide y vencerás son fundamentales para resolver problemas complejos de forma eficiente. Permiten descomponer tareas grandes en subproblemas más simples, como se ve en el cálculo de factoriales, el traslado de discos en las torres de Hanoi y la resolución de Sudokus mediante backtracking. Estos enfoques no solo optimizan el proceso, sino que también ofrecen soluciones claras, estructuradas y escalables en programación y sistemas reales.