

Fidel Morales Briones A01198630

Tarea 3 programación dinámica

Pseudocódigo:

```
Unset
findCoinCount(array coins, int amount):
    dp = [] # arreglo para almacenar el valor mínimo de monedas por cada
cantidad hasta amount
    coinCount = [[]] # arreglo para almacenar la cantidad de cada moneda
por cada cantidad hasta amount

    // iterar desde la cantidad de 1 hasta amount para encontrar la mínima
cantidad de monedas para cada valor y las monedas que se utilizarán para
cada valor
    for (i = 1 until amount) {
        // revisa cada moneda
        for (j = 0 until coins size) {
            // condición para verificar que la resta no dará un número
negativo
            if (i >= coins[j] && dp[i - coins[j]] != INF) {
                // revisar si la suma con esta moneda da una menor cantidad de
monedas utilizadas
                if (dp[i - coins[j]] + 1 < dp[i]) {
                    dp[i] = dp[i - coins[j]] + 1
                    coinCount[i] = coinCount[i - coins[j]]
                    coinCount[i][j]++
                }
            }
        }
    }

    // regresar un arreglo vacío si no hay una respuesta
    if (dp[amount] == INF) return []

    // regresar el arreglo de la cantidad de cada moneda
    return coinCount[amount]
```

Análisis de complejidad con código:

C/C++

```
vector<int> findCoinCount(vector<int> coins, int amount) {  
    vector<int> dp(amount + 1, INT_MAX); 0(1)  
    vector<vector<int>> coinCount(amount + 1, vector<int>(coins.size(),  
0)); 0(1)  
    dp[0] = 0; 0(1)  
  
    for (int i = 1; i <= amount; i++) { 0(m)  
  
        for (int j = 0; j < coins.size(); j++) { 0(n * m)  
            if(i >= coins[j] && dp[i - coins[j]] != INT_MAX) { 0(1)  
                if (dp[i - coins[j]] + 1 < dp[i]) { 0(1)  
                    dp[i] = dp[i - coins[j]] + 1; 0(1)  
                    coinCount[i] = coinCount[i - coins[j]]; 0(1)  
                    coinCount[i][j]++; 0(1)  
                }  
            }  
        }  
    }  
  
    if (dp[amount] == INT_MAX) { 0(1)  
        return {}; 0(1)  
    }  
  
    return coinCount[amount]; 0(1)  
}
```

Prueba:

```
PS C:\Users\fidel\Desktop\tec uni\5to\algoritmos\dynamic programming> ./tarea_tres.exe 10 17 input.txt  
1  
1  
0  
0  
PS C:\Users\fidel\Desktop\tec uni\5to\algoritmos\dynamic programming>
```

Bibliografía:

NeetCode. (2021, 6 enero). *Coin Change - Dynamic Programming Bottom Up - LeetCode*

322 [Vídeo]. YouTube. <https://www.youtube.com/watch?v=H9bfqozjoqs>