



INSTITUTO POLITÉCNICO NACIONAL



ESCUELA SUPERIOR DE CÓMPUTO

INGENIERIA EN SISTEMA COMPUTACIONALES

UNIDAD DE APRENDIZAJE: ANALISIS Y DISEÑO DE ALGORITMOS

PRÁCTICA 4:

PROGRAMACIÓN DINÁMICA

ALUMNOS:

BASTIDA GONZÁLEZ MARÍA GUADALUPE

2021630164

MEJÍA ROMO PABLO EMANUEL

2020350970

PROFESORA: LUZ MARIA SANCHEZ GARCIA

1. Programar en ANSI C, 2 de los siguientes algoritmos por programación dinámica:
 - a. Fibonacci (Top-down y Bottom-up)
 - b. Mochila entera (0,1)
 - c. Coeficientes binomiales
 - d. Subsecuencia más larga
2. Describir brevemente en qué consiste cada uno de los algoritmos por programación dinámica

Fibonacci (Top-down y Bottom-up)

Esta es una secuencia infinita de números, que se define de forma recursiva. Los primeros dos números son 0 y 1 respectivamente. Luego, cada número es igual a la suma de los dos anteriores.

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89...

Queremos escribir un programa que **calcule el n-ésimo número en la sucesión de Fibonacci**.

- Bottom-Up (Iterativo)

La idea importante es calcular los números de fibonacci de menor a mayor, manteniendo siempre la lista de los anteriores. De esta forma, cuando quiero calcular un número, ya tengo resueltos sus dos anteriores, por lo cual solo tengo que sumarlos.

- Top-Down (Recursión + Memoización)

El problema que se tiene al resolver utilizando recursión fue que se repiten cálculos de algunos números de Fibonacci. Sin embargo, se puede evitar repetir los cálculos modificando levemente la función. En lugar de simplemente calcular, cada vez que se llame a la función se debe verificar si el valor que deseamos utilizar ya fue calculado. En caso afirmativo, se devuelve la respuesta que ya se tenía. En caso contrario, se realiza el cálculo.

Mochila entera (0,1)

El problema de la mochila (KP) puede ser definido con un conjunto de n artículos donde cada artículo es identificado por n_x , con un valor entero p_x , y un peso w_x . El problema consiste en elegir un subconjunto de n artículos maximizando el beneficio obtenido considerando el peso total de los artículos seleccionados, sin exceder la capacidad c de la mochila.

El problema denominado 0-1 Knapsack Problem, se tienen un número k clases, donde se puede elegir solo un ítem j , donde el número total de ítems seleccionados para ocupar el contenedor es presentado con la variable N_i , donde $i=1,2, \dots, k$ y se tiene como función objetivo, maximizar el beneficio.

3. Explicar si existen otras formas de resolver cada algoritmo.

El algoritmo de Fibonacci si se puede resolver de otra forma, por fuerza bruta, en este caso únicamente se ejecuta un ciclo n numero de veces, en donde n es el n -ésimo numero de la sucesión.

Al igual que Fibonacci, el algoritmo de la mochila se puede implementar con fuerza bruta, pero al utilizarla los resultados obtenidos no son los óptimos y/o el algoritmo es lento.

En ambos casos, los algoritmos no son exactamente los mismos, es decir, el planteamiento del problema es diferente, pero en esencia se buscan los mismos resultados.

4. Indicar la ecuación de recurrencia y notación de orden (Big O) de cada algoritmo.

Fibonacci (Top-down y Bottom-up)

- Bottom-Up (Iterativo)

$$T(n) = n + 6 \quad O(n)$$

- Top-Down (Recursión + Memoización)

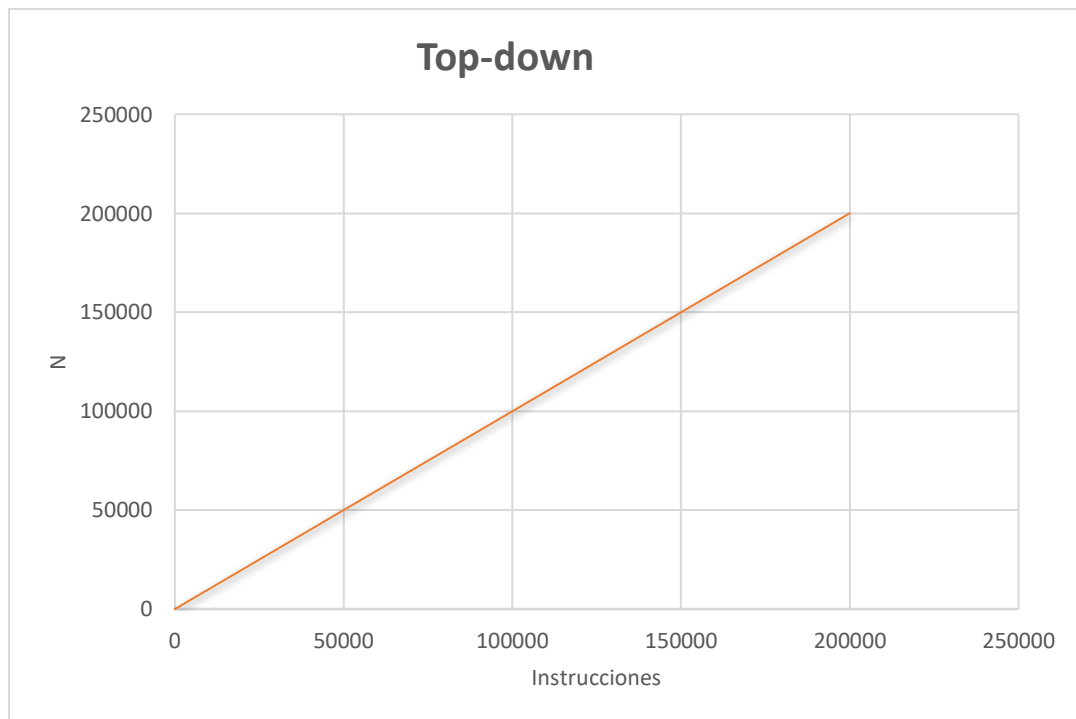
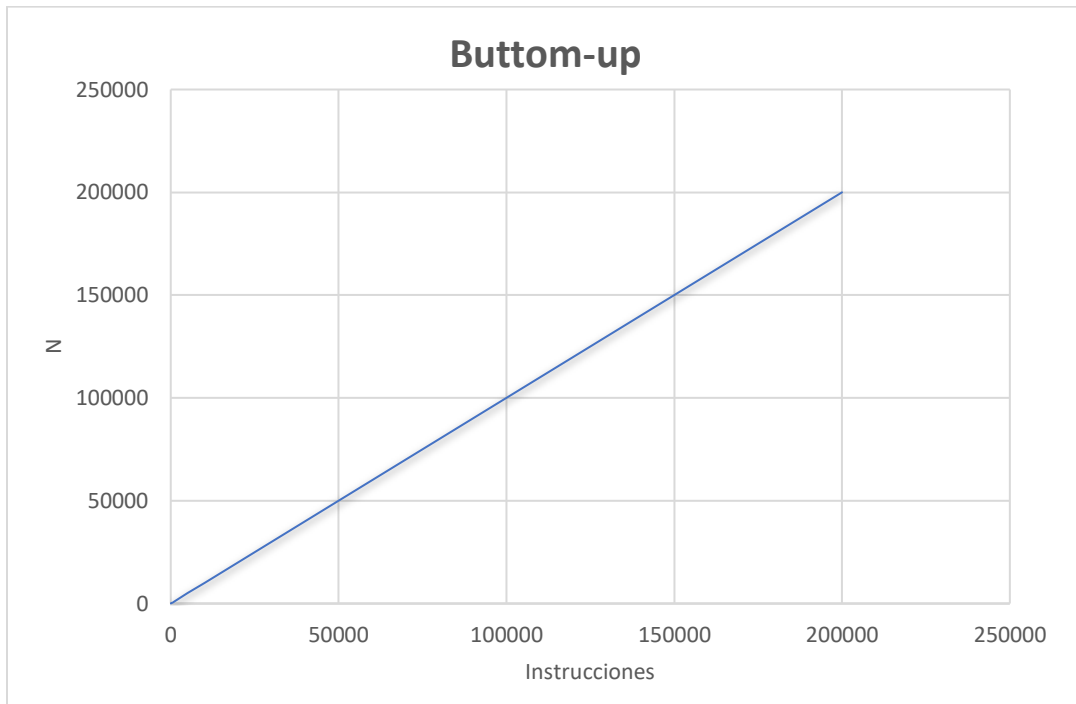
$$T(n) = \begin{cases} 1 & 0 \leq n \leq 1 \\ 1 + T(n-1) + T(n-2) & n > 1 \end{cases} \quad O(n)$$

Mochila entera (0,1)

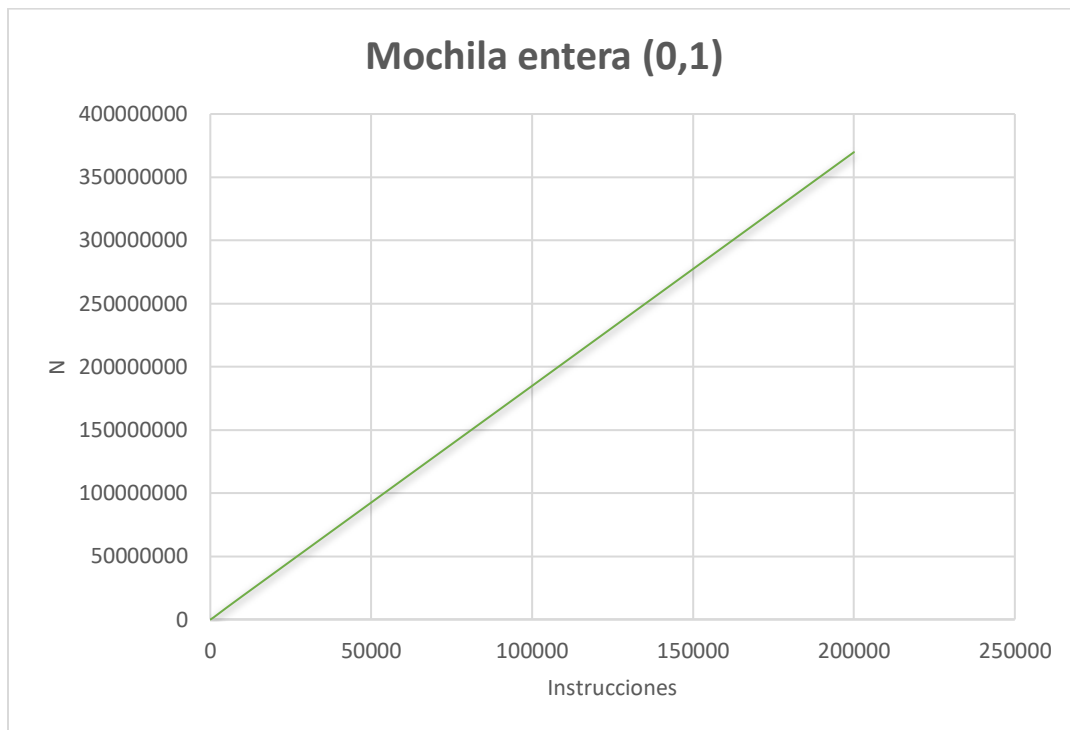
$$T(n) = Awn + Bw + Cn + D \quad O(w * n)$$

5. Graficar el comportamiento temporal de cada uno de los algoritmos (1 gráfica para cada uno de los algoritmos).

Fibonacci (Top-down y Bottom-up)



Mochila entera (0,1)



6. Ejemplifique cada algoritmo con valores de entrada y los resultados óptimos que cada algoritmo arroja para maximizar o minimizar la función objetivo.

Fibonacci (Top-down y Bottom-up)

Dato	Top-down	Bottom-up
5	5	5
10	55	55
15	610	610
20	6765	6765
25	75025	75025
30	832040	832040

Mochila entera (0,1)

i	p	v	J -> (P)								
			0	1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0	0	0	0	0
1	4	10	0	0	0	0	10	10	10	10	10
2	3	40	0	0	0	40	40	40	40	50	50
3	5	30	0	0	0	40	40	40	40	50	70
4	2	20	0	0	20	40	40	60	60	60	70

7. Indique si los datos del punto 6 fueron utilizados para realizar las pruebas al implementar los algoritmos, en caso contrario, indique los valores utilizados en cada uno de ellos.

Los valores anteriores si fueron utilizados en la implementación, y sus resultados son correctos en ambos casos.







8. Realice la conclusión del comportamiento de cada algoritmo de acuerdo con la programación dinámica.

En el caso de Fibonacci, el top-down es una maravilla de implementación ya que se toman en cuenta los casos base, pero no de una manera explícita, sino que sobre la marcha se van presentando automáticamente.

En la mochila entera es curioso cómo es que la matriz necesita la primera columna y fila igualadas en 0, al momento de la implementación no se tomó en cuenta esto y retrasó mucho la solución.

9. Indique el entorno controlado para realizar las pruebas experimentales.

Se utilizó el editor de códigos Visual Studio Code junto con la terminal de comandos que esta incorpora (CMD) para la programación de los códigos en c, el equipo de cómputo donde se ejecutaron dichos códigos cuenta con las siguientes especificaciones:

-  Procesador: Intel Core i7-4600 de 4 núcleos a 2.69 GHz
-  Memoria RAM: DDR3 de 8 GB
-  Unidad de almacenamiento: Unidad de estado sólido de 500 GB
-  Sistema operativo: x64 o 64 bits
-  Compilador: MinGw32-gcc v6.3.0-1
-  Editor de códigos: Visual Studio Code v1.55.2

10. Responda las siguientes preguntas:

- a. Para cada algoritmo ¿existe solución que no sea por programación dinámica y que sea óptima? ¿cuál es?**

No, las otras opciones, si resuelven los problemas, pero no son óptimas en su ejecución y en algunos casos, sus resultados.

- b. ¿Cuál de los 2 algoritmos es más fácil de implementar?**

Fibonacci (Top-down y Bottom-up)

- c. ¿Cuál de los 2 algoritmos es más difícil de implementar?**

Mochila completa (0,1)

- d. ¿El comportamiento experimental de los algoritmos es esperado? ¿Porqué?**

Si, ya que se buscaron simuladores que resolvían estos problemas y los resultados fueron los mismos.

- e. ¿Qué recomendaciones propone para realizar esta práctica?**

Ser muy cuidadosos al momento de la implementación, se debe tomar en cuenta como se comporta el algoritmo, exactamente como se haría a mano.