Comenzado el	viernes, 11 de noviembre de 2022, 12:05
Estado	Finalizado
Finalizado en	viernes, 11 de noviembre de 2022, 12:15
Tiempo empleado	9 minutos 27 segundos
Calificación	**7,67** de 10,00 (**77**%)

Pregunta **1**

Correcta

Se puntúa 1,00 sobre 1,00

Se suele utilizar programación dinámica en lugar de recursión sin memoria cuando hay una gran cantidad de subproblemas repetidos.

Seleccione una:

- a. Verdadero ✔
- b. Falso

Verdadero. En algunos algoritmos recursivos se generan muchas llamadas repetidas, lo que redunda en un coste elevado. La programación dinámica utiliza una tabla para almacenar los valores de los subproblemas ya calculados, de forma que en caso de requerirse varias veces no sea necesario volver a calcularlos sino solamente recuperarlos de la tabla.

La respuesta correcta es: Verdadero

Pregunta **2**

Incorrecta

Se puntúa -0,33 sobre 1,00

El algoritmo de programación dinámica ascendente en el que se optimiza la cantidad de memoria utilizada que calcula un número combinatorio acaba de rellenar el vector que corresponde a los números combinatorios $\binom{10}{0}$ $\binom{10}{1}$ $\ldots$ $\binom{10}{10}$ y queda de la siguiente forma:

0	1	2	3	4	5	6	7	8	9	10
1	10	45	120	210	252	210	120	45	10	1

Indica qué valores de este vector suma el algoritmo para calcular $\binom{11}{3}$

Seleccione una:

- a. Con este vector aún no se puede calcular
- b. 10 + 45 ✖
- c. 10 + 120
- d. 45 + 120

Para calcular $\binom{11}{3}$ son necesarios los valores $\binom{10}{2}$ y $\binom{10}{3}$, situados en las posiciones 2 y 3 respectivamente por lo que la respuesta es $45 + 120$.

La respuesta correcta es: 45 + 120

Pregunta **3**

Correcta

Se puntúa 1,00 sobre 1,00

Al calcular el número combinatorio $\binom{n}{r}$ mediante programación dinámica descendente, el número de veces que se repite cada subproblema está en $\Theta(1)$.

Seleccione una:

- a. Verdadero
- b. Falso

Verdadero. Cada subproblema $\binom{i}{j}$ se calcula una sola vez a partir de los valores de sus subproblemas. Una vez calculado su valor se guarda y se utilizará a lo sumo para calcular $\binom{i+1}{j}$ y $\binom{i+1}{j+1}$. Por tanto el número de veces que se repite cada subproblema está en $\Theta(1)$.

La respuesta correcta es: Verdadero

Pregunta **4**

Correcta

Se puntúa 1,00 sobre 1,00

Al calcular el número combinatorio $\binom{n}{r}$ mediante programación dinámica ascendente, el número de subproblemas distintos que se resuelven es $(n+1)*(r+1)$.

Seleccione una:

- a. Verdadero
- b. Falso

Verdadero. Se calculan todos los números combinatorios $\binom{i}{j}$ con $0 \leq i \leq n$ y $0 \leq j \leq r$, por tanto $(n+1)*(r+1)$ subproblemas distintos.

La respuesta correcta es: Verdadero

Pregunta **5**

Correcta

Se puntúa 1,00 sobre 1,00

Al calcular el número combinatorio $\binom{n}{r}$ utilizando recursión sin memoria, el número total de llamadas recursivas realizadas está en $\Theta(n^2)$.

Seleccione una:

- a. Verdadero
- b. Falso

Falso. El número de veces que se repiten los términos $\binom{i}{j}$ que hacen falta para calcular $\binom{n}{r}$ está representado por un número combinatorio. Por ejemplo, el término $\binom{2}{1}$ se calcula $\binom{n-2}{r-1}$ veces. Se puede demostrar que $\binom{n}{n/2} \in \Theta(2^n)$.

La respuesta correcta es: Falso

Pregunta **6**

Correcta

Se puntúa 1,00 sobre 1,00

En el problema del cambio de monedas con un sistema monetario de valores {1, 4, 7}, ¿de cuántas formas posibles se puede pagar la cantidad 11 con el menor número de monedas?

Seleccione una:

- a. 3
- b. 0
- c. 4
- d. 1 ✔

El mínimo número de monedas para pagar dicha cantidad es 2.

A medida que construimos la tabla de programación dinámica ascendente, aquí mostrada

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
0	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	0	1	2	3	4	5	6	7	8	9	10	11
2	0	1	2	3	1	2	3	4	2	3	4	5
3	0	1	2	3	1	2	3	1	2	3	4	2

podemos rellenar una matriz que en cada posición (i, j) guarda el número de formas de obtener la cantidad j con el menor número de monedas usando los tipos 1 a i.

La columna 0 la rellenamos con unos y el resto de la fila 0 con ceros.

Para el resto de casos si $monedas[i-1][j]$ coincide con $monedas[i-1][j-M[i-1]]+1$, entonces el número de formas para pagar j (usando el menor número de monedas) con las monedas de tipos 1 a i es el número de formas para pagar j sin usar la moneda i más el número de formas para pagar $j-M[i-1]$ usando los tipos 1 a i de nuevo. En este caso obtenemos la siguiente tabla:

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
0	1	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	2	2	2	1

Y por tanto la solución es: 1

La respuesta correcta es: 1

Pregunta **7**

Correcta

Se puntúa 1,00 sobre 1,00

Supongamos un sistema monetario con cantidad ilimitada de monedas de valores {2, 6, 7}. Se desea pagar la cantidad 13 con el menor número de monedas. Escribe separados por un espacio y ordenados de menor a mayor los índices de la última fila de la matriz rellenada por el algoritmo *que contienen un infinito*.

Respuesta: 1 3 5 ✔

Las filas se rellenan de arriba abajo y de izquierda a derecha obteniendo la siguiente tabla:

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
0	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	0	∞	1	∞	2	∞	3	∞	4	∞	5	∞	6	∞
2	0	∞	1	∞	2	∞	1	∞	2	∞	3	∞	2	∞
3	0	∞	1	∞	2	∞	1	1	2	2	3	3	2	2

Por tanto la solución es: 1 3 5

La respuesta correcta es: 1 3 5

Pregunta **8**

Sin contestar

Puntúa como 1,00

Supongamos un sistema monetario con cantidad ilimitada de monedas de valores {1, 2, 4}. Se desea pagar la cantidad 9 con el menor número de monedas. Escribe separados por un espacio los valores de la última fila de la matriz rellenada por el algoritmo de programación dinámica que resuelve el problema.

Respuesta: ✖

Las filas se rellenan de arriba abajo y de izquierda a derecha obteniendo la siguiente tabla:

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	0	∞	∞	∞	∞	∞	∞	∞	∞	∞
1	0	1	2	3	4	5	6	7	8	9
2	0	1	1	2	2	3	3	4	4	5
3	0	1	1	2	1	2	2	3	2	3

Por tanto la solución es: 0 1 1 2 1 2 2 3 2 3

La respuesta correcta es: 0 1 1 2 1 2 2 3 2 3

Pregunta **9**

Correcta

Se puntúa 1,00 sobre 1,00

El problema del cambio de monedas de una cantidad C con un número ilimitado de monedas de n tipos, resuelto por programación dinámica, puede tener un coste en espacio adicional en $\Theta(C)$ incluso cuando queremos saber cuántas monedas de cada tipo utilizar.

Seleccione una:

- a. Verdadero ✔
- b. Falso

Verdadero. En el problema del cambio de monedas, al contrario que en otros problemas, toda la información necesaria para reconstruir la solución se encuentra en la última fila debido a que el número de monedas de cada tipo es ilimitado. Por ello, se puede reconstruir la solución incluso cuando optimizamos el espacio usando del orden de $\Theta(C)$ memoria adicional.

La respuesta correcta es: Verdadero

Pregunta **10**

Correcta

Se puntúa 1,00 sobre 1,00

Supongamos una función recursiva que está definida de la siguiente manera:

$$f(1) = e_1$$
$$f(i) = max(e_i, f(i-1) + e_i) \text{ para } i > 1$$

siendo $e_1, \ldots, e_n$ expresiones numéricas y que queremos implementar un algoritmo de programación dinámica ascendente que calcule $max_{i=1}^{n} f(i)$. ¿Qué cantidad de memoria adicional óptima necesita el algoritmo?

Seleccione una:

- a. $\Theta(n)$
- b. $\Theta(n \log n)$
- c. $\Theta(1)$ ✔
- d. $\Theta(n^2)$

Basta con utilizar una variable para guardar el valor de $f(i)$ que vamos calculando (desde 1 hasta n por ser ascendente), y otra para guardar el máximo de los valores de f calculados hasta el momento. Luego la respuesta correcta es $\Theta(1)$.

La respuesta correcta es: $\Theta(1)$

Ir a...

Cuestionario - Árboles AVL ►