
Nombre:

DNI / NIE:

Análisis y Diseño de Algoritmos
Examen final C4 - 6 de julio de 2022

Instrucciones:

- ANTES DE COMENZAR EL EXAMEN poned vuestros datos en el cuadernillo de preguntas y también en la hoja de respuestas.
- NO OLVIDES poner la modalidad en la hoja de respuestas.
- DEJAD encima de la mesa vuestro DNI o vuestra TIU.
- No podéis consultar ningún material ni hablar con nadie.
- Cada pregunta solo tiene una opción correcta, marcadla como se indica en la hoja de respuestas.
- Cada respuesta incorrecta resta la mitad de una correcta. Las preguntas sin contestar ni suman ni restan puntos.
- Tenéis 90 min. para hacer el examen.

Normativa: (*Reglamento para la evaluación de los aprendizajes*, 27-11-2015)

- Está prohibido acceder al aula del examen con cualquier tipo de dispositivo electrónico.
- Además de dos bolígrafos o lápices, del documento de identificación personal y del material suministrado por el profesorado, NO se permite tener ningún objeto o documento, ni en la mesa ni en sus inmediaciones.
- Si tienes dudas acerca de un objeto o dispositivo concreto, pregunta al profesor antes de que comience la prueba.
- El incumplimiento de esta normativa puede conllevar, entre otras, la expulsión del aula del examen sin posibilidad de realizar la prueba.

Algunas preguntas hacen referencia al problema con el que hemos trabajado en las sesiones de prácticas. Su enunciado común es el siguiente:

Encaminamiento óptimo: *Se deben situar m puertas de acceso en una red de n servidores. Cada servidor i tiene una capacidad de almacenamiento $c_i \geqslant 0$ y d_{i,p_i} es la distancia de i a la puerta de enlace más cercana p_i. El tráfico total de la red es $\sum_{i=1}^{n} c_i d_{i,p_i}$, y debe minimizarse disponiendo de manera óptima las m puertas.*

De este problema hemos distinguido dos versiones según cómo están interconectados los nodos:

- **versión general:** Los nodos y sus interconexiones constituyen un grafo.
- **versión simplificada:** Los nodos se conectan entre sí mediante un único *bus* bidireccional.

Modalidad 1

Preguntas:

1. Estamos trabajando en el problema del encaminamiento óptimo, pero ahora nos dicen que tenemos que tener en cuenta el ancho de banda de las tarjetas de red de los ordenadores, por lo que el tráfico pasará a calcularse como: $\sum_{i=1}^{n} c_i a_i d(i, p_i)$, donde c_i, a_i y p_i son respectivamente, la capacidad, el ancho de banda y la puerta de acceso más cercana al ordenador i. ¿Cuál de las siguientes frases es correcta?
   (a) Al ser a_i un número real, la solución ya no podrá expresarse como un vector. Por tanto, el problema no podrá resolverse por ramificación y poda
   (b) El algoritmo para resolver el problema va a ser prácticamente igual, la complejidad no cambiará
   (c) Esto simplifica el problema y podrá resolverse mediante programación dinámica.
2. ¿Puede resolverse el problema del encaminamiento óptimo (versión simplificada) usando una estrategia de ramificación y poda?
   (a) No
   (b) Sí
   (c) Sí y solo si las capacidades son números enteros positivos
3. Un algoritmo recursivo basado en el esquema divide y vencerás ...
   (a) ... alcanza su máxima eficiencia cuando el problema de tamaño n se divide en a problemas de tamaño n/a.
   (b) ... nunca tendrá un coste temporal asintótico exponencial.
   (c) Las dos anteriores son verdaderas.
4. Una cuadrícula de $m \times n$ cuadrados forma un laberinto. La casilla $(0, 0)$ es la casilla en la que se sitúa el explorador al principio y la casilla $(m-1, n-1)$ es la casilla a la que debe llegar el explorador atravesando el número mínimo de casillas. Las casillas pueden estar o no ocupadas por un bloque que impide que el explorador entre en ella. El explorador solo se puede mover hacia la derecha (es decir, de (i, j) a $(i+1, j)$) o hacia abajo (es decir, de (i, j) a $(i, j+1)$). Los bloqueos están situados de manera que el problema siempre tiene solución. Indica cuál de las siguientes afirmaciones es cierta.
   (a) La solución se puede obtener a través de un algoritmo de programación dinámica iterativa, ya que el problema muestra subestructura óptima, y dicha solución tiene una complejidad temporal $O(mn)$
   (b) El problema carece de subestructura óptima y por tanto la complejidad temporal en el peor caso es $O(2^{m+n})$ ya que el explorador realiza un máximo de $m + n$ movimientos de entre dos posibles (problema de selección discreta).
   (c) Existe siempre una solución voraz al problema que consiste en elegir, desde cada casilla visitada (i, j), aquella casilla de entre las dos casillas vecinas $(i+1, j)$ e $(i, j+1)$ que minimiza en cada momento el número de casillas visitadas hasta el momento.
5. Estamos trabajando en el problema del encaminamiento óptimo. Para calcular el tráfico de una determinada configuración tenemos un vector en el que guardamos, para cada ordenador que no es puerta de acceso, cuál es su puerta de acceso más cercana. Si ahora queremos convertir un ordenador que no era puerta de acceso en puerta de acceso, ¿podemos aprovechar la información de ese vector para ahorrar cálculos?
   (a) Sí, usando una estrategia similar a la usada en el algoritmo de Kruskal
   (b) Sí, usando una estrategia similar a la usada en el algoritmo de Prim
   (c) No

6. Encargamos a un becario que elabore un algoritmo para sumar todos los números de un vector. Al cabo de un rato nos viene con un algoritmo cuya complejidad es $O(\log n)$. ¿Qué hacemos?
   (a) Damos las gracias al becario, ese es el algoritmo obvio.
   (b) Despedimos al becario, eso es imposible.
   (c) Subimos el sueldo al becario, ha encontrado un algoritmo innovador.

7. Un problema de optimización de un determinado indicador por selección discreta de los elementos de un vector de n componentes tiene una solución subóptima voraz que es a menudo una excelente aproximación. ¿Tiene sentido usar el valor obtenido mediante la solución subóptima para establecer la prioridad de visita de los nodos en un algoritmo de ramificación y poda?
   (a) No, porque se trata de una cota pesimista y establecería unas prioridades erróneas que retrasarían la obtención de la solución óptima.
   (b) No hay problema, en principio, en usar una cota pesimista para establecer las prioridades; de hecho, en algún problema podría establecer un orden de visita muy favorable.
   (c) No, porque ello interferiría con la poda basada en cotas optimistas y el mejor valor conocido.

8. ¿Puede resolverse el problema del encaminamiento óptimo (versión general) usando una estrategia de divide y vencerás?
   (a) Sí, aunque aparecerán muchos subproblemas repetidos. Será mucho mejor resolverlo usando programación dinámica
   (b) Sí. Además es muy poco probable que aparezcan subproblemas repetidos y, por tanto, no será efectivo recurrir a la programación dinámica
   (c) No, ya que el problema no tiene una subestructura óptima

9. ¿Podemos saber cuál sería el elemento en posición k cuando ordenáramos un vector de n elementos sin tener que ordenarlo?
   (a) Sí, y el algoritmo es $\Omega(n)$ y $O(n^2)$, aunque la frecuencia de los casos peores disminuye muy rápidamente con n.
   (b) No. Debemos ordenarlo.
   (c) Sí, y el algoritmo es $\Omega(n \log n)$ y $O(n^2)$, aunque la frecuencia de los casos peores disminuye muy rápidamente con n.

10. ¿Cuál sería la complejidad **espacial** mínima de una función, con el siguiente perfil:
```
void add(
   const vector<int>&a,
   const vector<int>&b,
   vector<int>&r
);
```
    que suma los vectores `a` y `b` y devuelve el resultado en el vector `r`?
    (a) $O(0)$
    (b) $O(n)$
    (c) $O(1)$

11. En un problema de minimización resuelto mediante ramificación y poda, una cota pesimista es ...
    (a) ... una cota superior para el valor óptimo, pero que nunca coincide con este.
    (b) ... una cota inferior para el valor óptimo que a veces coincide con este.
    (c) Ninguna de las otras dos opciones es cierta.

12. Tenemos dos algoritmos de ordenación. El algoritmo A pertenece a $\Theta(n \log n)$ y el B pertenece a $\Theta(n^2)$. ¿Es cierto que el algoritmo A va a ser siempre más rápido que el B?
    (a) Sí, ya que $n \log n$ crece mas lento que n^2
    (b) Sí, ya que n^2 crece mas lento que $n \log n$
    (c) No
13. ¿Se puede resolver mediante ramificación y poda un problema de selección discreta en el que cada una de las n decisiones se toman de un conjunto finito diferente?
    (a) Sí, siempre .
    (b) No. Las decisiones deben ser todas de la misma naturaleza.
    (c) Sí, pero sólo si las decisiones son todas binarias.
14. La complejidad temporal del algoritmo Mergesort cuando se aplica a un vector de tamaño n es... (selecciona la más ajustada)
    (a) ... $\Omega(n \log n)$ y $O(n^2)$.
    (b) ... $\Theta(n \log n)$.
    (c) ... $\Omega(n)$ y $O(n^2)$.
15. Se debe implantar fibra óptica en una isla, en la que la población está dispersa en n pequeños núcleos de población, los cuales deben quedar conectados. Los ingenieros tienen a mano una tabla en la que para cada par de localidades (i, j), se indica el coste C_{ij} de tender una línea de fibra entre i y j. Las localidades pueden estar conectadas directamente o indirectamente a través de otras localidades; el coste de los repetidores que se deben instalar para las conexiones indirectas es despreciable comparado con el de las líneas. Indica cuál de las siguientes afirmaciones es correcta en cuanto al tendido óptimo (es decir, de coste mínimo total) de fibra óptica.
    (a) El tendido óptimo es siempre un recorrido en bus (es decir, con $n - 1$ tramos de línea) elegido mediante un algoritmo voraz.
    (b) El tendido óptimo tiene siempre forma de árbol, con $n - 1$ tramos de línea y puede obtenerse mediante un algoritmo voraz.
    (c) El tendido óptimo tiene siempre una estructura radial, y existe un algoritmo muy eficiente para elegir el centro.
16. ¿Puede utilizarse relaciones de recurrencia para analizar la complejidad de un algoritmo de vuelta atrás?
    (a) No, ya que siempre saldría una complejidad exponencial
    (b) No, las relaciones de recurrencia no se pueden aplicar en este caso.
    (c) Sí.
17. ¿Cuál de las siguientes recurrencias expresa la complejidad en el caso más desfavorable del algoritmo (visto en clase) que encuentra el k-ésimo elemento más grande de un vector?
    (a) $f(n) = f(n/2) + n$; $f(1) = 1$
    (b) $f(n) = f(n-1) + n$; $f(1) = 1$
    (c) $f(n) = 2f(n-1) + n$; $f(1) = 1$

18. Queremos formar una suma M con el número mínimo posible de monedas tomadas (con repetición) de un conjunto de m monedas $C = \{c_1, c_2, \ldots c_m\}$. Una solución recursiva posible es la siguiente:

$$n_{\text{opt}}(M) = \begin{cases} 1 + \text{mín}_{1 \leqslant i \leqslant |C|}\, n_{\text{opt}}(M - c_i) & M > 0 \\ 0 & M = 0 \\ \infty & M < 0 \end{cases}$$

¿Sería posible convertir esta solución recursiva en un algoritmo de programación dinámica iterativa con complejidad temporal $O(M|C|)$?

(a) No, no hay manera de evitar la repetición de cálculos y por tanto la complejidad temporal es prohibitiva.

(b) Sí, ya que es posible calcular $n_{\text{opt}}(x)$ para cualquier $x > 0$ usando $n_{\text{opt}}(y)$ calculados para $y < x$.

(c) No, pero sí que se puede añadir memoización al cálculo para evitar la repetición de cálculos.

19. Se pretende resolver la versión simplificada del problema del encaminamiento óptimo mediante la técnica "divide y vencerás". ¿Se podría obtener la mejor disposición de las puertas?

(a) No, ya que no se cumple la propiedad "subestructura óptima".

(b) No, ya que no cumple el teorema de reducción.

(c) Sí, pero a costa de una complejidad temporal prohibitiva.

20. En el esquema de vuelta atrás, los mecanismos de poda basados en la mejor solución hasta el momento...

(a) ... pueden eliminar vectores que representan posibles soluciones factibles.

(b) ... garantizan que no se va a explorar todo el espacio de soluciones posibles.

(c) Las dos anteriores son verdaderas.

21. ¿Cuál es el caso peor del algoritmo de ordenación Quicksort que toma el primer elemento como pivote?

(a) Cuando el elemento pivote queda siempre en uno de los dos extremos.

(b) Cuando el elemento pivote queda siempre en medio.

(c) Cuando debemos hundir el pivote hasta el fondo del montículo.

22. Tenemos un conjunto de n enteros positivos y queremos encontrar el subconjunto de tamaño m de suma mínima.

(a) Lo más adecuado sería usar una técnica de ramificación y poda, aunque en el peor caso el coste temporal asintótico (o complejidad temporal) sería exponencial.

(b) Para encontrar la solución habría que probar con todas las combinaciones posibles de m enteros, con lo que la técnica de ramificación y poda no aporta nada con respecto a vuelta atrás.

(c) Una técnica voraz daría una solución óptima.

23. ¿Para cuál de estos tres algoritmos de ordenación la complejidad temporal en el caso peor es $O(n^2)$?

(a) Mergesort

(b) Quicksort

(c) Heapsort

24. Se pretende resolver el problema del encaminamiento óptimo (versión general) mediante ramificación y poda. ¿Qué podemos asegurar si el subóptimo de partida coincide con la cota optimista del nodo inicial?

    (a) Las otras dos opciones son ambas falsas.

    (b) Que la cota optimista está mal calculada.

    (c) Que la cota pesimista está mal calculada.

25. Sea el vector $v[8] = \{8, 6, 4, 5, 4, 3, 2, 2\}$. Indica cuál de las siguientes opciones es cierta. (se asume la notación del lenguaje C/C++ en la que el primer elemento del vector está en la posición 0, es decir, en $v[0]$).

    (a) El vector v no es un montículo máximo porque el elemento v[3]=5 debe ser "flotado" (desplazado hacia la izquierda).

    (b) El vector v es un montículo máximo.

    (c) El vector v no es un montículo máximo porque el elemento v[2]=4 debe ser "hundido" (desplazado hacia la derecha)..

26. Estamos trabajando el problema del encaminamiento óptimo usando ramificación y poda. Codificamos la solución como un vector de booleanos para indicar si el ordenador en cuestión es una puerta de enlace o no. Empezamos suponiendo que no hay puertas de enlace y vamos añadiendo nuevas. ¿es una buena poda basada en la cota optimista parar la exploración cuando el número de puertas de enlace es superior a m (el número de puertas de enlace requerido)?

    (a) Sí

    (b) No, podría ser que al expandir el nodo el número de puertas disminuya

    (c) No, eso no es una poda basada en la cota optimista

27. Con respecto al parámetro n, ¿Cuál sería la complejidad temporal de la siguiente función si se aplicara memoización?

```
long f( unsigned n ) {
  if( n <= 1 )
    return 1;
  return n * f( n - 2);
}
```

    (a) lineal

    (b) constante

    (c) logarítmica

28. La función $\texttt{met}(m, n)$, con $1 \leqslant m \leqslant n$, se puede calcular recursivamente siguiendo el siguiente esquema:

$$\texttt{met}(m,n) = \begin{cases} \texttt{ogw}(1,n) & \text{if } m = 1 \\ \min_{k=m-1}^{n-1} \texttt{met}(m-1,k) + \texttt{ogw}(k,n) & \text{if } n > m \end{cases}$$

    Estamos interesados en utilizar programación dinámica para acelerar el cálculo pero queremos usar la mínima memoria posible. ¿Qué estructura del almacén sería la más adecuada?

    (a) una tabla (dos índices)

    (b) un vector (un índice)

    (c) una constante (ningún índice)

29. Se pretende resolver el problema del encaminamiento óptimo (versión general) mediante ramificación y poda y para ello se hace uso de una cota que consiste en asumir que los restantes nodos aún no tratados no van a ser puerta de enlace. ¿Que podemos decir de esta cota?

    (a) Que es una cota pesimista solo si ya se han colocado exactamente m puertas.

    (b) Que es una cota optimista.

    (c) Las otras dos opciones son ambas falsas.

30. Sea $f(n) = n^2 + 4$. Dos de las tres afirmaciones siguientes prueban que $f(n) \in O(n^2)$. ¿Cuál es la que no?

    (a) Para todo $n > 2$ se cumple que $n^2 + 4 < 2n^2$

    (b) Para todo $n > \sqrt{\frac{4}{c-1}}$, con $c > 1$, se cumple que $n^2 + 4 > cn^2$.

    (c) Para todo $n < \sqrt{\frac{4}{c-1}}$, con $c > 1$, se cumple que $n^2 + 4 < cn^2$.

31. Queremos imprimir los n enteros positivos que contiene un vector v, ordenados de menor a mayor. Suponed que el entero más grande que puede aparecer es m. Para ello hacemos un algoritmo que hace lo siguiente:

    a) crea un vector de enteros auxiliar (c) de tamaño ($m + 1$) y lo inicializa con ceros.

    b) para cada i desde 0 a $n - 1$ ejecuta `c[v[i]]++`

    c) para cada i desde 0 a m imprime `a[i]` veces el valor i.

    ¿Qué complejidad tiene este algoritmo en función de n?

    (a) $O(n)$, pero este algoritmo no ordena bien en todos los casos

    (b) $O(n \log n)$

    (c) $O(n)$

32. Sea la relación de recurrencia $T(n) = 2T(n/2) + 1; T(1) = 1$. Indica cuál de estas tres expresiones es cierta:

    (a) $T(n) \in \Theta(n^2)$

    (b) $T(n) \in \Theta(n \log n)$

    (c) $T(n) \in \Theta(n)$

33. De las siguientes expresiones, o bien dos son ciertas y una es falsa, o bien al contrario, una es cierta y dos son falsas. Marca la que en este sentido es diferente a las otras dos.

    (a) $\sum_{i=1}^{\log n} \sum_{j=1}^{i} 2^j \in O(n)$

    (b) $\sum_{i=1}^{n} \sum_{j=1}^{\log i} 1 \in O(n)$

    (c) $\sum_{i=1}^{\log n} \sum_{j=1}^{n} 2^j \in O(n \log n)$

34. Si $\lim_{n\to\infty} \frac{f(n)}{n^k} = a$ con $a > 0$, qué podemos afirmar de la función f?

    (a) $f(n) \in \Theta(n^k)$.

    (b) $f(n) = an^k$.

    (c) $f(n) \in O(n)$.

35. ¿Pertenece $n^2 + n + 1$ a $\Omega(n)$?
    (a) Sí
    (b) No
    (c) Si n es pequeño, no

36. Se dispone de un conjunto de n valores numéricos dispuestos en forma de árbol binario y se desea obtener el valor de la suma de todos ellos. ¿Cuál es la complejidad temporal del mejor algoritmo que se puede escribir?
    (a) $O(n)$
    (b) $O(\log n)$
    (c) $O(n \log n)$

37. De las siguientes expresiones, o bien dos son ciertas y una es falsa, o bien al contrario, una es cierta y dos son falsas. Marca la que en este sentido es diferente a las otras dos.
    (a) Si $f(n) \in O(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = 0$
    (b) Si $f(n) \in O(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = \infty$
    (c) Si $f(n) \in \Theta(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = k$

38. ¿Cuál es la complejidad temporal de la función $\mathrm{proc}()$ en función del tamaño n del vector?

$$\begin{aligned}\mathrm{proc}(\mathrm{vec}(a,b)) = \mathrm{ifelse}(&a > b,\\ &\mathrm{concat}(\mathrm{proc}(\mathrm{vec}(a,(a+b)/2)), \mathrm{proc}(\mathrm{vec}((a+b)/2+1,b)),\\ &\mathrm{vec}(a,a)\\ &)\end{aligned}$$

    La notación $\mathrm{vec}(a,b)$ indica el subvector que va de la coordenada a a la coordenada b (tiene $b-a+1$ componentes; cuando $b = a-1$ tiene cero componentes). La función $\mathrm{concat}()$ concatena dos vectores y tiene un coste constante y la función $\mathrm{ifelse}()$ es análoga a una estructura "if…else" en C.
    (a) $\Theta(2^n)$
    (b) $\Theta(n)$
    (c) $\Theta(\log n)$

Nombre:

DNI / NIE:

Análisis y Diseño de Algoritmos
Examen final C4 - 6 de julio de 2022

Instrucciones:

- ANTES DE COMENZAR EL EXAMEN poned vuestros datos en el cuadernillo de preguntas y también en la hoja de respuestas.
- NO OLVIDES poner la modalidad en la hoja de respuestas.
- DEJAD encima de la mesa vuestro DNI o vuestra TIU.
- No podéis consultar ningún material ni hablar con nadie.
- Cada pregunta solo tiene una opción correcta, marcadla como se indica en la hoja de respuestas.
- Cada respuesta incorrecta resta la mitad de una correcta. Las preguntas sin contestar ni suman ni restan puntos.
- Tenéis 90 min. para hacer el examen.

Normativa: (*Reglamento para la evaluación de los aprendizajes*, 27-11-2015)

- Está prohibido acceder al aula del examen con cualquier tipo de dispositivo electrónico.
- Además de dos bolígrafos o lápices, del documento de identificación personal y del material suministrado por el profesorado, NO se permite tener ningún objeto o documento, ni en la mesa ni en sus inmediaciones.
- Si tienes dudas acerca de un objeto o dispositivo concreto, pregunta al profesor antes de que comience la prueba.
- El incumplimiento de esta normativa puede conllevar, entre otras, la expulsión del aula del examen sin posibilidad de realizar la prueba.

Algunas preguntas hacen referencia al problema con el que hemos trabajado en las sesiones de prácticas. Su enunciado común es el siguiente:

Encaminamiento óptimo: *Se deben situar m puertas de acceso en una red de n servidores. Cada servidor i tiene una capacidad de almacenamiento $c_i \geqslant 0$ y d_{i,p_i} es la distancia de i a la puerta de enlace más cercana p_i. El tráfico total de la red es $\sum_{i=1}^{n} c_i d_{i,p_i}$, y debe minimizarse disponiendo de manera óptima las m puertas.*

De este problema hemos distinguido dos versiones según cómo están interconectados los nodos:

- **versión general:** Los nodos y sus interconexiones constituyen un grafo.
- **versión simplificada:** Los nodos se conectan entre sí mediante un único *bus* bidireccional.

Modalidad 2

Preguntas:

1. ¿Puede utilizarse relaciones de recurrencia para analizar la complejidad de un algoritmo de vuelta atrás?
   (a) No, las relaciones de recurrencia no se pueden aplicar en este caso.
   (b) Sí.
   (c) No, ya que siempre saldría una complejidad exponencial

2. Sea $f(n) = n^2 + 4$. Dos de las tres afirmaciones siguientes prueban que $f(n) \in O(n^2)$. ¿Cuál es la que no?
   (a) Para todo $n > \sqrt{\frac{4}{c-1}}$, con $c > 1$, se cumple que $n^2 + 4 > cn^2$.
   (b) Para todo $n > 2$ se cumple que $n^2 + 4 < 2n^2$
   (c) Para todo $n < \sqrt{\frac{4}{c-1}}$, con $c > 1$, se cumple que $n^2 + 4 < cn^2$.

3. Queremos formar una suma M con el número mínimo posible de monedas tomadas (con repetición) de un conjunto de m monedas $C = \{c_1, c_2, \ldots c_m\}$. Una solución recursiva posible es la siguiente:

$$n_{\text{opt}}(M) = \begin{cases} 1 + \text{mín}_{1 \leqslant i \leqslant |C|}\, n_{\text{opt}}(M - c_i) & M > 0 \\ 0 & M = 0 \\ \infty & M < 0 \end{cases}$$

   ¿Sería posible convertir esta solución recursiva en un algoritmo de programación dinámica iterativa con complejidad temporal $O(M|C|)$?
   (a) Sí, ya que es posible calcular $n_{\text{opt}}(x)$ para cualquier $x > 0$ usando $n_{\text{opt}}(y)$ calculados para $y < x$.
   (b) No, pero sí que se puede añadir memoización al cálculo para evitar la repetición de cálculos.
   (c) No, no hay manera de evitar la repetición de cálculos y por tanto la complejidad temporal es prohibitiva.

4. ¿Para cuál de estos tres algoritmos de ordenación la complejidad temporal en el caso peor es $O(n^2)$?
   (a) Quicksort
   (b) Heapsort
   (c) Mergesort

5. Un algoritmo recursivo basado en el esquema divide y vencerás ...
   (a) ... nunca tendrá un coste temporal asintótico exponencial.
   (b) Las dos anteriores son verdaderas.
   (c) ... alcanza su máxima eficiencia cuando el problema de tamaño n se divide en a problemas de tamaño n/a.

6. En el esquema de vuelta atrás, los mecanismos de poda basados en la mejor solución hasta el momento...
   (a) ... garantizan que no se va a explorar todo el espacio de soluciones posibles.
   (b) Las dos anteriores son verdaderas.
   (c) ... pueden eliminar vectores que representan posibles soluciones factibles.

7. Sea la relación de recurrencia $T(n) = 2T(n/2) + 1; T(1) = 1$. Indica cuál de estas tres expresiones es cierta:
   (a) $T(n) \in \Theta(n \log n)$
   (b) $T(n) \in \Theta(n)$
   (c) $T(n) \in \Theta(n^2)$

8. ¿Podemos saber cuál sería el elemento en posición k cuando ordenáramos un vector de n elementos sin tener que ordenarlo?
   (a) No. Debemos ordenarlo.
   (b) Sí, y el algoritmo es $\Omega(n)$ y $O(n^2)$, aunque la frecuencia de los casos peores disminuye muy rápidamente con n.
   (c) Sí, y el algoritmo es $\Omega(n \log n)$ y $O(n^2)$, aunque la frecuencia de los casos peores disminuye muy rápidamente con n.

9. ¿Puede resolverse el problema del encaminamiento óptimo (versión general) usando una estrategia de divide y vencerás?
   (a) Sí. Además es muy poco probable que aparezcan subproblemas repetidos y, por tanto, no será efectivo recurrir a la programación dinámica
   (b) No, ya que el problema no tiene una subestructura óptima
   (c) Sí, aunque aparecerán muchos subproblemas repetidos. Será mucho mejor resolverlo usando programación dinámica

10. ¿Cuál es el caso peor del algoritmo de ordenación Quicksort que toma el primer elemento como pivote?
    (a) Cuando el elemento pivote queda siempre en medio.
    (b) Cuando el elemento pivote queda siempre en uno de los dos extremos.
    (c) Cuando debemos hundir el pivote hasta el fondo del montículo.

11. Se pretende resolver el problema del encaminamiento óptimo (versión general) mediante ramificación y poda. ¿Qué podemos asegurar si el subóptimo de partida coincide con la cota optimista del nodo inicial?
    (a) Que la cota optimista está mal calculada.
    (b) Que la cota pesimista está mal calculada.
    (c) Las otras dos opciones son ambas falsas.

12. Tenemos dos algoritmos de ordenación. El algoritmo A pertenece a $\Theta(n \log n)$ y el B pertenece a $\Theta(n^2)$. ¿Es cierto que el algoritmo A va a ser siempre más rápido que el B?
    (a) Sí, ya que n^2 crece mas lento que $n \log n$
    (b) No
    (c) Sí, ya que $n \log n$ crece mas lento que n^2

13. Tenemos un conjunto de n enteros positivos y queremos encontrar el subconjunto de tamaño m de suma mínima.
    (a) Para encontrar la solución habría que probar con todas las combinaciones posibles de m enteros, con lo que la técnica de ramificación y poda no aporta nada con respecto a vuelta atrás.
    (b) Una técnica voraz daría una solución óptima.
    (c) Lo más adecuado sería usar una técnica de ramificación y poda, aunque en el peor caso el coste temporal asintótico (o complejidad temporal) sería exponencial.

14. ¿Cuál sería la complejidad **espacial** mínima de una función, con el siguiente perfil:

```
void add(
   const vector<int>&a,
   const vector<int>&b,
   vector<int>&r
);
```

que suma los vectores `a` y `b` y devuelve el resultado en el vector `r`?

(a) $O(n)$

(b) $O(1)$

(c) $O(0)$

15. Encargamos a un becario que elabore un algoritmo para sumar todos los números de un vector. Al cabo de un rato nos viene con un algoritmo cuya complejidad es $O(\log n)$. ¿Qué hacemos?

(a) Despedimos al becario, eso es imposible.

(b) Subimos el sueldo al becario, ha encontrado un algoritmo innovador.

(c) Damos las gracias al becario, ese es el algoritmo obvio.

16. Si $\lim_{n\to\infty} \frac{f(n)}{n^k} = a$ con $a > 0$, qué podemos afirmar de la función f?

(a) $f(n) = an^k$.

(b) $f(n) \in \Theta(n^k)$.

(c) $f(n) \in O(n)$.

17. Un problema de optimización de un determinado indicador por selección discreta de los elementos de un vector de n componentes tiene una solución subóptima voraz que es a menudo una excelente aproximación. ¿Tiene sentido usar el valor obtenido mediante la solución subóptima para establecer la prioridad de visita de los nodos en un algoritmo de ramificación y poda?

(a) No hay problema, en principio, en usar una cota pesimista para establecer las prioridades; de hecho, en algún problema podría establecer un orden de visita muy favorable.

(b) No, porque ello interferiría con la poda basada en cotas optimistas y el mejor valor conocido.

(c) No, porque se trata de una cota pesimista y establecería unas prioridades erróneas que retrasarían la obtención de la solución óptima.

18. La función $\texttt{met}(m, n)$, con $1 \leqslant m \leqslant n$, se puede calcular recursivamente siguiendo el siguiente esquema:

$$\texttt{met}(m,n) = \begin{cases} \texttt{ogw}(1,n) & \text{if } m = 1 \\ \min_{k=m-1}^{n-1} \texttt{met}(m-1,k) + \texttt{ogw}(k,n) & \text{if } n > m \end{cases}$$

Estamos interesados en utilizar programación dinámica para acelerar el cálculo pero queremos usar la mínima memoria posible. ¿Qué estructura del almacén sería la más adecuada?

(a) un vector (un índice)

(b) una constante (ningún índice)

(c) una tabla (dos índices)

19. Estamos trabajando en el problema del encaminamiento óptimo. Para calcular el tráfico de una determinada configuración tenemos un vector en el que guardamos, para cada ordenador que no es puerta de acceso, cuál es su puerta de acceso más cercana. Si ahora queremos convertir un ordenador que no era puerta de acceso en puerta de acceso, ¿podemos aprovechar la información de ese vector para ahorrar cálculos?
    (a) Sí, usando una estrategia similar a la usada en el algoritmo de Prim
    (b) No
    (c) Sí, usando una estrategia similar a la usada en el algoritmo de Kruskal

20. De las siguientes expresiones, o bien dos son ciertas y una es falsa, o bien al contrario, una es cierta y dos son falsas. Marca la que en este sentido es diferente a las otras dos.
    (a) Si $f(n) \in O(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = \infty$
    (b) Si $f(n) \in \Theta(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = k$
    (c) Si $f(n) \in O(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = 0$

21. Se dispone de un conjunto de n valores numéricos dispuestos en forma de árbol binario y se desea obtener el valor de la suma de todos ellos. ¿Cuál es la complejidad temporal del mejor algoritmo que se puede escribir?
    (a) $O(\log n)$
    (b) $O(n \log n)$
    (c) $O(n)$

22. De las siguientes expresiones, o bien dos son ciertas y una es falsa, o bien al contrario, una es cierta y dos son falsas. Marca la que en este sentido es diferente a las otras dos.
    (a) $\sum_{i=1}^{n} \sum_{j=1}^{\log i} 1 \in O(n)$
    (b) $\sum_{i=1}^{\log n} \sum_{j=1}^{n} 2^j \in O(n \log n)$
    (c) $\sum_{i=1}^{\log n} \sum_{j=1}^{i} 2^j \in O(n)$

23. Sea el vector $v[8] = \{8, 6, 4, 5, 4, 3, 2, 2\}$. Indica cuál de las siguientes opciones es cierta. (se asume la notación del lenguaje C/C++ en la que el primer elemento del vector está en la posición 0, es decir, en $v[0]$).
    (a) El vector v es un montículo máximo.
    (b) El vector v no es un montículo máximo porque el elemento v[2]=4 debe ser "hundido" (desplazado hacia la derecha)..
    (c) El vector v no es un montículo máximo porque el elemento v[3]=5 debe ser "flotado" (desplazado hacia la izquierda).

24. ¿Se puede resolver mediante ramificación y poda un problema de selección discreta en el que cada una de las n decisiones se toman de un conjunto finito diferente?
    (a) No. Las decisiones deben ser todas de la misma naturaleza.
    (b) Sí, pero sólo si las decisiones son todas binarias.
    (c) Sí, siempre .

25. Se pretende resolver la versión simplificada del problema del encaminamiento óptimo mediante la técnica "divide y vencerás". ¿Se podría obtener la mejor disposición de las puertas?

    (a) No, ya que no cumple el teorema de reducción.

    (b) Sí, pero a costa de una complejidad temporal prohibitiva.

    (c) No, ya que no se cumple la propiedad "subestructura óptima".

26. Queremos imprimir los n enteros positivos que contiene un vector v, ordenados de menor a mayor. Suponed que el entero más grande que puede aparecer es m. Para ello hacemos un algoritmo que hace lo siguiente:

    a) crea un vector de enteros auxiliar (c) de tamaño ($m + 1$) y lo inicializa con ceros.

    b) para cada i desde 0 a $n - 1$ ejecuta `c[v[i]]++`

    c) para cada i desde 0 a m imprime `a[i]` veces el valor i.

    ¿Qué complejidad tiene este algoritmo en función de n?

    (a) $O(n \log n)$

    (b) $O(n)$

    (c) $O(n)$, pero este algoritmo no ordena bien en todos los casos

27. Con respecto al parámetro n, ¿Cuál sería la complejidad temporal de la siguiente función si se aplicara memoización?

```
long f( unsigned n ) {
  if( n <= 1 )
    return 1;
  return n * f( n - 2);
}
```

    (a) constante

    (b) logarítmica

    (c) lineal

28. Se pretende resolver el problema del encaminamiento óptimo (versión general) mediante ramificación y poda y para ello se hace uso de una cota que consiste en asumir que los restantes nodos aún no tratados no van a ser puerta de enlace. ¿Que podemos decir de esta cota?

    (a) Que es una cota optimista.

    (b) Las otras dos opciones son ambas falsas.

    (c) Que es una cota pesimista solo si ya se han colocado exactamente m puertas.

29. ¿Cuál de las siguientes recurrencias expresa la complejidad en el caso más desfavorable del algoritmo (visto en clase) que encuentra el k-ésimo elemento más grande de un vector?

    (a) $f(n) = f(n-1) + n$; $f(1) = 1$

    (b) $f(n) = 2f(n-1) + n$; $f(1) = 1$

    (c) $f(n) = f(n/2) + n$; $f(1) = 1$

30. Una cuadrícula de $m \times n$ cuadrados forma un laberinto. La casilla $(0, 0)$ es la casilla en la que se sitúa el explorador al principio y la casilla $(m-1, n-1)$ es la casilla a la que debe llegar el explorador atravesando el número mínimo de casillas. Las casillas pueden estar o no ocupadas por un bloque que impide que el explorador entre en ella. El explorador solo se puede mover hacia la derecha (es decir, de (i, j) a $(i+1, j)$) o hacia abajo (es decir, de (i, j) a $(i, j+1)$). Los bloqueos están situados de manera que el problema siempre tiene solución. Indica cuál de las siguientes afirmaciones es cierta.

    (a) El problema carece de subestructura óptima y por tanto la complejidad temporal en el peor caso es $O(2^{m+n})$ ya que el explorador realiza un máximo de $m + n$ movimientos de entre dos posibles (problema de selección discreta).

    (b) Existe siempre una solución voraz al problema que consiste en elegir, desde cada casilla visitada (i, j), aquella casilla de entre las dos casillas vecinas $(i+1, j)$ e $(i, j+1)$ que minimiza en cada momento el número de casillas visitadas hasta el momento.

    (c) La solución se puede obtener a través de un algoritmo de programación dinámica iterativa, ya que el problema muestra subestructura óptima, y dicha solución tiene una complejidad temporal $O(mn)$

31. ¿Cuál es la complejidad temporal de la función $\mathrm{proc}()$ en función del tamaño n del vector?

$$\begin{aligned}\mathrm{proc}(\mathrm{vec}(a,b)) = \mathrm{ifelse}(&a > b,\\ &\mathrm{concat}(\mathrm{proc}(\mathrm{vec}(a,(a+b)/2)), \mathrm{proc}(\mathrm{vec}((a+b)/2+1,b)),\\ &\mathrm{vec}(a,a)\\ &)\end{aligned}$$

    La notación $\mathrm{vec}(a, b)$ indica el subvector que va de la coordenada a a la coordenada b (tiene $b-a+1$ componentes; cuando $b = a-1$ tiene cero componentes). La función $\mathrm{concat}()$ concatena dos vectores y tiene un coste constante y la función $\mathrm{ifelse}()$ es análoga a una estructura "if…else" en C.

    (a) $\Theta(n)$

    (b) $\Theta(\log n)$

    (c) $\Theta(2^n)$

32. Estamos trabajando el problema del encaminamiento óptimo usando ramificación y poda. Codificamos la solución como un vector de booleanos para indicar si el ordenador en cuestión es una puerta de enlace o no. Empezamos suponiendo que no hay puertas de enlace y vamos añadiendo nuevas. ¿es una buena poda basada en la cota optimista parar la exploración cuando el número de puertas de enlace es superior a m (el número de puertas de enlace requerido)?

    (a) No, podría ser que al expandir el nodo el número de puertas disminuya

    (b) No, eso no es una poda basada en la cota optimista

    (c) Sí

33. Estamos trabajando en el problema del encaminamiento óptimo, pero ahora nos dicen que tenemos que tener en cuenta el ancho de banda de las tarjetas de red de los ordenadores, por lo que el tráfico pasará a calcularse como: $\sum_{i=1}^{n} c_i a_i d(i, p_i)$, donde c_i, a_i y p_i son respectivamente, la capacidad, el ancho de banda y la puerta de acceso más cercana al ordenador i. ¿Cuál de las siguientes frases es correcta?

    (a) El algoritmo para resolver el problema va a ser prácticamente igual, la complejidad no cambiará

    (b) Esto simplifica el problema y podrá resolverse mediante programación dinámica.

    (c) Al ser a_i un número real, la solución ya no podrá expresarse como un vector. Por tanto, el problema no podrá resolverse por ramificación y poda

34. ¿Pertenece $n^2 + n + 1$ a $\Omega(n)$?
    (a) No
    (b) Si n es pequeño, no
    (c) Sí

35. ¿Puede resolverse el problema del encaminamiento óptimo (versión simplificada) usando una estrategia de ramificación y poda?
    (a) Sí
    (b) Sí y solo si las capacidades son números enteros positivos
    (c) No

36. Se debe implantar fibra óptica en una isla, en la que la población está dispersa en n pequeños núcleos de población, los cuales deben quedar conectados. Los ingenieros tienen a mano una tabla en la que para cada par de localidades (i, j), se indica el coste C_{ij} de tender una línea de fibra entre i y j. Las localidades pueden estar conectadas directamente o indirectamente a través de otras localidades; el coste de los repetidores que se deben instalar para las conexiones indirectas es despreciable comparado con el de las líneas. Indica cuál de las siguientes afirmaciones es correcta en cuanto al tendido óptimo (es decir, de coste mínimo total) de fibra óptica.
    (a) El tendido óptimo tiene siempre forma de árbol, con $n - 1$ tramos de línea y puede obtenerse mediante un algoritmo voraz.
    (b) El tendido óptimo tiene siempre una estructura radial, y existe un algoritmo muy eficiente para elegir el centro.
    (c) El tendido óptimo es siempre un recorrido en bus (es decir, con $n - 1$ tramos de línea) elegido mediante un algoritmo voraz.

37. La complejidad temporal del algoritmo Mergesort cuando se aplica a un vector de tamaño n es... (selecciona la más ajustada)
    (a) ... $\Theta(n \log n)$.
    (b) ... $\Omega(n \log n)$ y $O(n^2)$.
    (c) ... $\Omega(n)$ y $O(n^2)$.

38. En un problema de minimización resuelto mediante ramificación y poda, una cota pesimista es ...
    (a) ... una cota inferior para el valor óptimo que a veces coincide con este.
    (b) Ninguna de las otras dos opciones es cierta.
    (c) ... una cota superior para el valor óptimo, pero que nunca coincide con este.

Nombre:

DNI / NIE:

Análisis y Diseño de Algoritmos
Examen final C4 - 6 de julio de 2022

Instrucciones:

- ANTES DE COMENZAR EL EXAMEN poned vuestros datos en el cuadernillo de preguntas y también en la hoja de respuestas.
- NO OLVIDES poner la modalidad en la hoja de respuestas.
- DEJAD encima de la mesa vuestro DNI o vuestra TIU.
- No podéis consultar ningún material ni hablar con nadie.
- Cada pregunta solo tiene una opción correcta, marcadla como se indica en la hoja de respuestas.
- Cada respuesta incorrecta resta la mitad de una correcta. Las preguntas sin contestar ni suman ni restan puntos.
- Tenéis 90 min. para hacer el examen.

Normativa: (*Reglamento para la evaluación de los aprendizajes*, 27-11-2015)

- Está prohibido acceder al aula del examen con cualquier tipo de dispositivo electrónico.
- Además de dos bolígrafos o lápices, del documento de identificación personal y del material suministrado por el profesorado, NO se permite tener ningún objeto o documento, ni en la mesa ni en sus inmediaciones.
- Si tienes dudas acerca de un objeto o dispositivo concreto, pregunta al profesor antes de que comience la prueba.
- El incumplimiento de esta normativa puede conllevar, entre otras, la expulsión del aula del examen sin posibilidad de realizar la prueba.

Algunas preguntas hacen referencia al problema con el que hemos trabajado en las sesiones de prácticas. Su enunciado común es el siguiente:

Encaminamiento óptimo: *Se deben situar m puertas de acceso en una red de n servidores. Cada servidor i tiene una capacidad de almacenamiento $c_i \geqslant 0$ y d_{i,p_i} es la distancia de i a la puerta de enlace más cercana p_i. El tráfico total de la red es $\sum_{i=1}^{n} c_i d_{i,p_i}$, y debe minimizarse disponiendo de manera óptima las m puertas.*

De este problema hemos distinguido dos versiones según cómo están interconectados los nodos:

- **versión general:** Los nodos y sus interconexiones constituyen un grafo.
- **versión simplificada:** Los nodos se conectan entre sí mediante un único *bus* bidireccional.

Modalidad 3

Preguntas:

1. ¿Cuál es el caso peor del algoritmo de ordenación Quicksort que toma el primer elemento como pivote?
   (a) Cuando el elemento pivote queda siempre en uno de los dos extremos.
   (b) Cuando el elemento pivote queda siempre en medio.
   (c) Cuando debemos hundir el pivote hasta el fondo del montículo.
2. Se pretende resolver la versión simplificada del problema del encaminamiento óptimo mediante la técnica "divide y vencerás". ¿Se podría obtener la mejor disposición de las puertas?
   (a) No, ya que no se cumple la propiedad "subestructura óptima".
   (b) No, ya que no cumple el teorema de reducción.
   (c) Sí, pero a costa de una complejidad temporal prohibitiva.
3. ¿Se puede resolver mediante ramificación y poda un problema de selección discreta en el que cada una de las n decisiones se toman de un conjunto finito diferente?
   (a) Sí, siempre .
   (b) No. Las decisiones deben ser todas de la misma naturaleza.
   (c) Sí, pero sólo si las decisiones son todas binarias.
4. Se pretende resolver el problema del encaminamiento óptimo (versión general) mediante ramificación y poda. ¿Qué podemos asegurar si el subóptimo de partida coincide con la cota optimista del nodo inicial?
   (a) Las otras dos opciones son ambas falsas.
   (b) Que la cota optimista está mal calculada.
   (c) Que la cota pesimista está mal calculada.
5. De las siguientes expresiones, o bien dos son ciertas y una es falsa, o bien al contrario, una es cierta y dos son falsas. Marca la que en este sentido es diferente a las otras dos.
   (a) Si $f(n) \in O(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = 0$
   (b) Si $f(n) \in O(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = \infty$
   (c) Si $f(n) \in \Theta(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = k$
6. ¿Puede resolverse el problema del encaminamiento óptimo (versión simplificada) usando una estrategia de ramificación y poda?
   (a) No
   (b) Sí
   (c) Sí y solo si las capacidades son números enteros positivos
7. Si $\lim_{n\to\infty} \frac{f(n)}{n^k} = a$ con $a > 0$, qué podemos afirmar de la función f?
   (a) $f(n) \in \Theta(n^k)$.
   (b) $f(n) = an^k$.
   (c) $f(n) \in O(n)$.
8. La complejidad temporal del algoritmo Mergesort cuando se aplica a un vector de tamaño n es... (selecciona la más ajustada)
   (a) $\ldots \Omega(n \log n)$ y $O(n^2)$.
   (b) $\ldots \Theta(n \log n)$.
   (c) $\ldots \Omega(n)$ y $O(n^2)$.

9. Sea $f(n) = n^2 + 4$. Dos de las tres afirmaciones siguientes prueban que $f(n) \in O(n^2)$. ¿Cuál es la que no?

   (a) Para todo $n > 2$ se cumple que $n^2 + 4 < 2n^2$

   (b) Para todo $n > \sqrt{\frac{4}{c-1}}$, con $c > 1$, se cumple que $n^2 + 4 > cn^2$.

   (c) Para todo $n < \sqrt{\frac{4}{c-1}}$, con $c > 1$, se cumple que $n^2 + 4 < cn^2$.

10. Con respecto al parámetro n, ¿Cuál sería la complejidad temporal de la siguiente función si se aplicara memoización?

```
long f( unsigned n ) {
  if( n <= 1 )
    return 1;
  return n * f( n - 2);
}
```

   (a) lineal

   (b) constante

   (c) logarítmica

11. ¿Pertenece $n^2 + n + 1$ a $\Omega(n)$?

   (a) Sí

   (b) No

   (c) Si n es pequeño, no

12. Se debe implantar fibra óptica en una isla, en la que la población está dispersa en n pequeños núcleos de población, los cuales deben quedar conectados. Los ingenieros tienen a mano una tabla en la que para cada par de localidades (i, j), se indica el coste C_{ij} de tender una línea de fibra entre i y j. Las localidades pueden estar conectadas directamente o indirectamente a través de otras localidades; el coste de los repetidores que se deben instalar para las conexiones indirectas es despreciable comparado con el de las líneas. Indica cuál de las siguientes afirmaciones es correcta en cuanto al tendido óptimo (es decir, de coste mínimo total) de fibra óptica.

   (a) El tendido óptimo es siempre un recorrido en bus (es decir, con $n - 1$ tramos de línea) elegido mediante un algoritmo voraz.

   (b) El tendido óptimo tiene siempre forma de árbol, con $n - 1$ tramos de línea y puede obtenerse mediante un algoritmo voraz.

   (c) El tendido óptimo tiene siempre una estructura radial, y existe un algoritmo muy eficiente para elegir el centro.

13. Tenemos dos algoritmos de ordenación. El algoritmo A pertenece a $\Theta(n \log n)$ y el B pertenece a $\Theta(n^2)$. ¿Es cierto que el algoritmo A va a ser siempre más rápido que el B?

   (a) Sí, ya que $n \log n$ crece mas lento que n^2

   (b) Sí, ya que n^2 crece mas lento que $n \log n$

   (c) No

14. Estamos trabajando el problema del encaminamiento óptimo usando ramificación y poda. Codificamos la solución como un vector de booleanos para indicar si el ordenador en cuestión es una puerta de enlace o no. Empezamos suponiendo que no hay puertas de enlace y vamos añadiendo nuevas. ¿es una buena poda basada en la cota optimista parar la exploración cuando el número de puertas de enlace es superior a m (el número de puertas de enlace requerido)?

   (a) Sí

   (b) No, podría ser que al expandir el nodo el número de puertas disminuya

   (c) No, eso no es una poda basada en la cota optimista

15. ¿Cuál de las siguientes recurrencias expresa la complejidad en el caso más desfavorable del algoritmo (visto en clase) que encuentra el k-ésimo elemento más grande de un vector?

    (a) $f(n) = f(n/2) + n$; $f(1) = 1$

    (b) $f(n) = f(n-1) + n$; $f(1) = 1$

    (c) $f(n) = 2f(n-1) + n$; $f(1) = 1$

16. Estamos trabajando en el problema del encaminamiento óptimo. Para calcular el tráfico de una determinada configuración tenemos un vector en el que guardamos, para cada ordenador que no es puerta de acceso, cuál es su puerta de acceso más cercana. Si ahora queremos convertir un ordenador que no era puerta de acceso en puerta de acceso, ¿podemos aprovechar la información de ese vector para ahorrar cálculos?

    (a) Sí, usando una estrategia similar a la usada en el algoritmo de Kruskal

    (b) Sí, usando una estrategia similar a la usada en el algoritmo de Prim

    (c) No

17. ¿Puede resolverse el problema del encaminamiento óptimo (versión general) usando una estrategia de divide y vencerás?

    (a) Sí, aunque aparecerán muchos subproblemas repetidos. Será mucho mejor resolverlo usando programación dinámica

    (b) Sí. Además es muy poco probable que aparezcan subproblemas repetidos y, por tanto, no será efectivo recurrir a la programación dinámica

    (c) No, ya que el problema no tiene una subestructura óptima

18. Tenemos un conjunto de n enteros positivos y queremos encontrar el subconjunto de tamaño m de suma mínima.

    (a) Lo más adecuado sería usar una técnica de ramificación y poda, aunque en el peor caso el coste temporal asintótico (o complejidad temporal) sería exponencial.

    (b) Para encontrar la solución habría que probar con todas las combinaciones posibles de m enteros, con lo que la técnica de ramificación y poda no aporta nada con respecto a vuelta atrás.

    (c) Una técnica voraz daría una solución óptima.

19. Sea el vector $v[8] = \{8, 6, 4, 5, 4, 3, 2, 2\}$. Indica cuál de las siguientes opciones es cierta. (se asume la notación del lenguaje C/C++ en la que el primer elemento del vector está en la posición 0, es decir, en $v[0]$).

    (a) El vector v no es un montículo máximo porque el elemento v[3]=5 debe ser "flotado" (desplazado hacia la izquierda).

    (b) El vector v es un montículo máximo.

    (c) El vector v no es un montículo máximo porque el elemento v[2]=4 debe ser "hundido" (desplazado hacia la derecha)..

20. ¿Puede utilizarse relaciones de recurrencia para analizar la complejidad de un algoritmo de vuelta atrás?

    (a) No, ya que siempre saldría una complejidad exponencial

    (b) No, las relaciones de recurrencia no se pueden aplicar en este caso.

    (c) Sí.

21. La función $\texttt{met}(m, n)$, con $1 \leqslant m \leqslant n$, se puede calcular recursivamente siguiendo el siguiente esquema:

$$\texttt{met}(m,n) = \begin{cases} \texttt{ogw}(1,n) & \text{if } m = 1 \\ \min_{k=m-1}^{n-1} \texttt{met}(m-1,k) + \texttt{ogw}(k,n) & \text{if } n > m \end{cases}$$

Estamos interesados en utilizar programación dinámica para acelerar el cálculo pero queremos usar la mínima memoria posible. ¿Qué estructura del almacén sería la más adecuada?

(a) una tabla (dos índices)

(b) un vector (un índice)

(c) una constante (ningún índice)

22. Se dispone de un conjunto de n valores numéricos dispuestos en forma de árbol binario y se desea obtener el valor de la suma de todos ellos. ¿Cuál es la complejidad temporal del mejor algoritmo que se puede escribir?

(a) $O(n)$

(b) $O(\log n)$

(c) $O(n \log n)$

23. En el esquema de vuelta atrás, los mecanismos de poda basados en la mejor solución hasta el momento...

(a) ... pueden eliminar vectores que representan posibles soluciones factibles.

(b) ... garantizan que no se va a explorar todo el espacio de soluciones posibles.

(c) Las dos anteriores son verdaderas.

24. ¿Para cuál de estos tres algoritmos de ordenación la complejidad temporal en el caso peor es $O(n^2)$?

(a) Mergesort

(b) Quicksort

(c) Heapsort

25. Un problema de optimización de un determinado indicador por selección discreta de los elementos de un vector de n componentes tiene una solución subóptima voraz que es a menudo una excelente aproximación. ¿Tiene sentido usar el valor obtenido mediante la solución subóptima para establecer la prioridad de visita de los nodos en un algoritmo de ramificación y poda?

(a) No, porque se trata de una cota pesimista y establecería unas prioridades erróneas que retrasarían la obtención de la solución óptima.

(b) No hay problema, en principio, en usar una cota pesimista para establecer las prioridades; de hecho, en algún problema podría establecer un orden de visita muy favorable.

(c) No, porque ello interferiría con la poda basada en cotas optimistas y el mejor valor conocido.

26. Una cuadrícula de $m \times n$ cuadrados forma un laberinto. La casilla $(0, 0)$ es la casilla en la que se sitúa el explorador al principio y la casilla $(m-1, n-1)$ es la casilla a la que debe llegar el explorador atravesando el número mínimo de casillas. Las casillas pueden estar o no ocupadas por un bloque que impide que el explorador entre en ella. El explorador solo se puede mover hacia la derecha (es decir, de (i, j) a $(i+1, j)$) o hacia abajo (es decir, de (i, j) a $(i, j+1)$). Los bloqueos están situados de manera que el problema siempre tiene solución. Indica cuál de las siguientes afirmaciones es cierta.

    (a) La solución se puede obtener a través de un algoritmo de programación dinámica iterativa, ya que el problema muestra subestructura óptima, y dicha solución tiene una complejidad temporal $O(mn)$

    (b) El problema carece de subestructura óptima y por tanto la complejidad temporal en el peor caso es $O(2^{m+n})$ ya que el explorador realiza un máximo de $m + n$ movimientos de entre dos posibles (problema de selección discreta).

    (c) Existe siempre una solución voraz al problema que consiste en elegir, desde cada casilla visitada (i, j), aquella casilla de entre las dos casillas vecinas $(i+1, j)$ e $(i, j+1)$ que minimiza en cada momento el número de casillas visitadas hasta el momento.

27. En un problema de minimización resuelto mediante ramificación y poda, una cota pesimista es ...

    (a) ... una cota superior para el valor óptimo, pero que nunca coincide con este.

    (b) ... una cota inferior para el valor óptimo que a veces coincide con este.

    (c) Ninguna de las otras dos opciones es cierta.

28. Se pretende resolver el problema del encaminamiento óptimo (versión general) mediante ramificación y poda y para ello se hace uso de una cota que consiste en asumir que los restantes nodos aún no tratados no van a ser puerta de enlace. ¿Que podemos decir de esta cota?

    (a) Que es una cota pesimista solo si ya se han colocado exactamente m puertas.

    (b) Que es una cota optimista.

    (c) Las otras dos opciones son ambas falsas.

29. ¿Cuál es la complejidad temporal de la función $\mathrm{proc}()$ en función del tamaño n del vector?

$$\begin{aligned}\mathrm{proc}(\mathrm{vec}(a,b)) = \mathrm{ifelse}(a > b,\\ &\mathrm{concat}(\mathrm{proc}(\mathrm{vec}(a,(a+b)/2)), \mathrm{proc}(\mathrm{vec}((a+b)/2+1,b)),\\ &\mathrm{vec}(a,a)\\ &)\end{aligned}$$

    La notación $\mathrm{vec}(a, b)$ indica el subvector que va de la coordenada a a la coordenada b (tiene $b-a+1$ componentes; cuando $b = a-1$ tiene cero componentes). La función $\mathrm{concat}()$ concatena dos vectores y tiene un coste constante y la función $\mathrm{ifelse}()$ es análoga a una estructura "if...else" en C.

    (a) $\Theta(2^n)$

    (b) $\Theta(n)$

    (c) $\Theta(\log n)$

30. ¿Cuál sería la complejidad **espacial** mínima de una función, con el siguiente perfil:

```
void add(
   const vector<int>&a,
   const vector<int>&b,
   vector<int>&r
);
```

que suma los vectores `a` y `b` y devuelve el resultado en el vector `r`?

(a) $O(0)$

(b) $O(n)$

(c) $O(1)$

31. Encargamos a un becario que elabore un algoritmo para sumar todos los números de un vector. Al cabo de un rato nos viene con un algoritmo cuya complejidad es $O(\log n)$. ¿Qué hacemos?

(a) Damos las gracias al becario, ese es el algoritmo obvio.

(b) Despedimos al becario, eso es imposible.

(c) Subimos el sueldo al becario, ha encontrado un algoritmo innovador.

32. Queremos formar una suma M con el número mínimo posible de monedas tomadas (con repetición) de un conjunto de m monedas $C = \{c_1, c_2, \ldots c_m\}$. Una solución recursiva posible es la siguiente:

$$n_{\text{opt}}(M) = \begin{cases} 1 + \min_{1 \leqslant i \leqslant |C|} n_{\text{opt}}(M - c_i) & M > 0 \\ 0 & M = 0 \\ \infty & M < 0 \end{cases}$$

¿Sería posible convertir esta solución recursiva en un algoritmo de programación dinámica iterativa con complejidad temporal $O(M|C|)$?

(a) No, no hay manera de evitar la repetición de cálculos y por tanto la complejidad temporal es prohibitiva.

(b) Sí, ya que es posible calcular $n_{\text{opt}}(x)$ para cualquier $x > 0$ usando $n_{\text{opt}}(y)$ calculados para $y < x$.

(c) No, pero sí que se puede añadir memoización al cálculo para evitar la repetición de cálculos.

33. Sea la relación de recurrencia $T(n) = 2T(n/2) + 1; T(1) = 1$. Indica cuál de estas tres expresiones es cierta:

(a) $T(n) \in \Theta(n^2)$

(b) $T(n) \in \Theta(n \log n)$

(c) $T(n) \in \Theta(n)$

34. Un algoritmo recursivo basado en el esquema divide y vencerás ...

(a) ... alcanza su máxima eficiencia cuando el problema de tamaño n se divide en a problemas de tamaño n/a.

(b) ... nunca tendrá un coste temporal asintótico exponencial.

(c) Las dos anteriores son verdaderas.

35. Queremos imprimir los n enteros positivos que contiene un vector v, ordenados de menor a mayor. Suponed que el entero más grande que puede aparecer es m. Para ello hacemos un algoritmo que hace lo siguiente:

    a) crea un vector de enteros auxiliar (c) de tamaño ($m + 1$) y lo inicializa con ceros.

    b) para cada i desde 0 a $n - 1$ ejecuta `c[v[i]]++`

    c) para cada i desde 0 a m imprime `a[i]` veces el valor i.

    ¿Qué complejidad tiene este algoritmo en función de n?

    (a) $O(n)$, pero este algoritmo no ordena bien en todos los casos

    (b) $O(n \log n)$

    (c) $O(n)$

36. ¿Podemos saber cuál sería el elemento en posición k cuando ordenáramos un vector de n elementos sin tener que ordenarlo?

    (a) Sí, y el algoritmo es $\Omega(n)$ y $O(n^2)$, aunque la frecuencia de los casos peores disminuye muy rápidamente con n.

    (b) No. Debemos ordenarlo.

    (c) Sí, y el algoritmo es $\Omega(n \log n)$ y $O(n^2)$, aunque la frecuencia de los casos peores disminuye muy rápidamente con n.

37. Estamos trabajando en el problema del encaminamiento óptimo, pero ahora nos dicen que tenemos que tener en cuenta el ancho de banda de las tarjetas de red de los ordenadores, por lo que el tráfico pasará a calcularse como: $\sum_{i=1}^{n} c_i a_i d(i, p_i)$, donde c_i, a_i y p_i son respectivamente, la capacidad, el ancho de banda y la puerta de acceso más cercana al ordenador i. ¿Cuál de las siguientes frases es correcta?

    (a) Al ser a_i un número real, la solución ya no podrá expresarse como un vector. Por tanto, el problema no podrá resolverse por ramificación y poda

    (b) El algoritmo para resolver el problema va a ser prácticamente igual, la complejidad no cambiará

    (c) Esto simplifica el problema y podrá resolverse mediante programación dinámica.

38. De las siguientes expresiones, o bien dos son ciertas y una es falsa, o bien al contrario, una es cierta y dos son falsas. Marca la que en este sentido es diferente a las otras dos.

    (a) $\sum_{i=1}^{\log n} \sum_{j=1}^{i} 2^j \in O(n)$

    (b) $\sum_{i=1}^{n} \sum_{j=1}^{\log i} 1 \in O(n)$

    (c) $\sum_{i=1}^{\log n} \sum_{j=1}^{n} 2^j \in O(n \log n)$

Nombre:

DNI / NIE:

Análisis y Diseño de Algoritmos
Examen final C4 - 6 de julio de 2022

Instrucciones:

- ANTES DE COMENZAR EL EXAMEN poned vuestros datos en el cuadernillo de preguntas y también en la hoja de respuestas.
- NO OLVIDES poner la modalidad en la hoja de respuestas.
- DEJAD encima de la mesa vuestro DNI o vuestra TIU.
- No podéis consultar ningún material ni hablar con nadie.
- Cada pregunta solo tiene una opción correcta, marcadla como se indica en la hoja de respuestas.
- Cada respuesta incorrecta resta la mitad de una correcta. Las preguntas sin contestar ni suman ni restan puntos.
- Tenéis 90 min. para hacer el examen.

Normativa: (*Reglamento para la evaluación de los aprendizajes*, 27-11-2015)

- Está prohibido acceder al aula del examen con cualquier tipo de dispositivo electrónico.
- Además de dos bolígrafos o lápices, del documento de identificación personal y del material suministrado por el profesorado, NO se permite tener ningún objeto o documento, ni en la mesa ni en sus inmediaciones.
- Si tienes dudas acerca de un objeto o dispositivo concreto, pregunta al profesor antes de que comience la prueba.
- El incumplimiento de esta normativa puede conllevar, entre otras, la expulsión del aula del examen sin posibilidad de realizar la prueba.

Algunas preguntas hacen referencia al problema con el que hemos trabajado en las sesiones de prácticas. Su enunciado común es el siguiente:

Encaminamiento óptimo: *Se deben situar m puertas de acceso en una red de n servidores. Cada servidor i tiene una capacidad de almacenamiento $c_i \geqslant 0$ y d_{i,p_i} es la distancia de i a la puerta de enlace más cercana p_i. El tráfico total de la red es $\sum_{i=1}^{n} c_i d_{i,p_i}$, y debe minimizarse disponiendo de manera óptima las m puertas.*

De este problema hemos distinguido dos versiones según cómo están interconectados los nodos:

- **versión general:** Los nodos y sus interconexiones constituyen un grafo.
- **versión simplificada:** Los nodos se conectan entre sí mediante un único *bus* bidireccional.

Modalidad 4

Preguntas:

1. Sea la relación de recurrencia $T(n) = 2T(n/2) + 1; T(1) = 1$. Indica cuál de estas tres expresiones es cierta:
   (a) $T(n) \in \Theta(n \log n)$
   (b) $T(n) \in \Theta(n)$
   (c) $T(n) \in \Theta(n^2)$

2. La complejidad temporal del algoritmo Mergesort cuando se aplica a un vector de tamaño n es...(selecciona la más ajustada)
   (a) ...$\Theta(n \log n)$.
   (b) ...$\Omega(n \log n)$ y $O(n^2)$.
   (c) ...$\Omega(n)$ y $O(n^2)$.

3. Si $\lim_{n\to\infty} \frac{f(n)}{n^k} = a$ con $a > 0$, qué podemos afirmar de la función f?
   (a) $f(n) = an^k$.
   (b) $f(n) \in \Theta(n^k)$.
   (c) $f(n) \in O(n)$.

4. ¿Se puede resolver mediante ramificación y poda un problema de selección discreta en el que cada una de las n decisiones se toman de un conjunto finito diferente?
   (a) No. Las decisiones deben ser todas de la misma naturaleza.
   (b) Sí, pero sólo si las decisiones son todas binarias.
   (c) Sí, siempre .

5. Un problema de optimización de un determinado indicador por selección discreta de los elementos de un vector de n componentes tiene una solución subóptima voraz que es a menudo una excelente aproximación. ¿Tiene sentido usar el valor obtenido mediante la solución subóptima para establecer la prioridad de visita de los nodos en un algoritmo de ramificación y poda?
   (a) No hay problema, en principio, en usar una cota pesimista para establecer las prioridades; de hecho, en algún problema podría establecer un orden de visita muy favorable.
   (b) No, porque ello interferiría con la poda basada en cotas optimistas y el mejor valor conocido.
   (c) No, porque se trata de una cota pesimista y establecería unas prioridades erróneas que retrasarían la obtención de la solución óptima.

6. Con respecto al parámetro n, ¿Cuál sería la complejidad temporal de la siguiente función si se aplicara memoización?

```
long f( unsigned n ) {
  if( n <= 1 )
    return 1;
  return n * f( n - 2);
}
```

   (a) constante
   (b) logarítmica
   (c) lineal

7. De las siguientes expresiones, o bien dos son ciertas y una es falsa, o bien al contrario, una es cierta y dos son falsas. Marca la que en este sentido es diferente a las otras dos.

   (a) $\sum_{i=1}^{n} \sum_{j=1}^{\log i} 1 \in O(n)$

   (b) $\sum_{i=1}^{\log n} \sum_{j=1}^{n} 2^j \in O(n \log n)$

   (c) $\sum_{i=1}^{\log n} \sum_{j=1}^{i} 2^j \in O(n)$

8. Estamos trabajando el problema del encaminamiento óptimo usando ramificación y poda. Codificamos la solución como un vector de booleanos para indicar si el ordenador en cuestión es una puerta de enlace o no. Empezamos suponiendo que no hay puertas de enlace y vamos añadiendo nuevas. ¿es una buena poda basada en la cota optimista parar la exploración cuando el número de puertas de enlace es superior a m (el número de puertas de enlace requerido)?

   (a) No, podría ser que al expandir el nodo el número de puertas disminuya

   (b) No, eso no es una poda basada en la cota optimista

   (c) Sí

9. Estamos trabajando en el problema del encaminamiento óptimo. Para calcular el tráfico de una determinada configuración tenemos un vector en el que guardamos, para cada ordenador que no es puerta de acceso, cuál es su puerta de acceso más cercana. Si ahora queremos convertir un ordenador que no era puerta de acceso en puerta de acceso, ¿podemos aprovechar la información de ese vector para ahorrar cálculos?

   (a) Sí, usando una estrategia similar a la usada en el algoritmo de Prim

   (b) No

   (c) Sí, usando una estrategia similar a la usada en el algoritmo de Kruskal

10. ¿Puede utilizarse relaciones de recurrencia para analizar la complejidad de un algoritmo de vuelta atrás?

    (a) No, las relaciones de recurrencia no se pueden aplicar en este caso.

    (b) Sí.

    (c) No, ya que siempre saldría una complejidad exponencial

11. Encargamos a un becario que elabore un algoritmo para sumar todos los números de un vector. Al cabo de un rato nos viene con un algoritmo cuya complejidad es $O(\log n)$. ¿Qué hacemos?

    (a) Despedimos al becario, eso es imposible.

    (b) Subimos el sueldo al becario, ha encontrado un algoritmo innovador.

    (c) Damos las gracias al becario, ese es el algoritmo obvio.

12. En el esquema de vuelta atrás, los mecanismos de poda basados en la mejor solución hasta el momento...

    (a) ... garantizan que no se va a explorar todo el espacio de soluciones posibles.

    (b) Las dos anteriores son verdaderas.

    (c) ... pueden eliminar vectores que representan posibles soluciones factibles.

13. ¿Puede resolverse el problema del encaminamiento óptimo (versión simplificada) usando una estrategia de ramificación y poda?

    (a) Sí

    (b) Sí y solo si las capacidades son números enteros positivos

    (c) No

14. ¿Pertenece $n^2 + n + 1$ a $\Omega(n)$?

    (a) No

    (b) Si n es pequeño, no

    (c) Sí

15. Una cuadrícula de $m \times n$ cuadrados forma un laberinto. La casilla $(0, 0)$ es la casilla en la que se sitúa el explorador al principio y la casilla $(m-1, n-1)$ es la casilla a la que debe llegar el explorador atravesando el número mínimo de casillas. Las casillas pueden estar o no ocupadas por un bloque que impide que el explorador entre en ella. El explorador solo se puede mover hacia la derecha (es decir, de (i, j) a $(i+1, j)$) o hacia abajo (es decir, de (i, j) a $(i, j+1)$). Los bloqueos están situados de manera que el problema siempre tiene solución. Indica cuál de las siguientes afirmaciones es cierta.

    (a) El problema carece de subestructura óptima y por tanto la complejidad temporal en el peor caso es $O(2^{m+n})$ ya que el explorador realiza un máximo de $m + n$ movimientos de entre dos posibles (problema de selección discreta).

    (b) Existe siempre una solución voraz al problema que consiste en elegir, desde cada casilla visitada (i, j), aquella casilla de entre las dos casillas vecinas $(i+1, j)$ e $(i, j+1)$ que minimiza en cada momento el número de casillas visitadas hasta el momento.

    (c) La solución se puede obtener a través de un algoritmo de programación dinámica iterativa, ya que el problema muestra subestructura óptima, y dicha solución tiene una complejidad temporal $O(mn)$

16. ¿Cuál es la complejidad temporal de la función $\mathrm{proc}()$ en función del tamaño n del vector?

$$\begin{aligned}\mathrm{proc}(\mathrm{vec}(a, b)) = \mathrm{ifelse}(a > b,\\ &\mathrm{concat}(\mathrm{proc}(\mathrm{vec}(a, (a+b)/2)), \mathrm{proc}(\mathrm{vec}((a+b)/2+1, b)),\\ &\mathrm{vec}(a, a)\\ &)\end{aligned}$$

    La notación $\mathrm{vec}(a, b)$ indica el subvector que va de la coordenada a a la coordenada b (tiene $b-a+1$ componentes; cuando $b = a-1$ tiene cero componentes). La función $\mathrm{concat}()$ concatena dos vectores y tiene un coste constante y la función $\mathrm{ifelse}()$ es análoga a una estructura "if…else" en C.

    (a) $\Theta(n)$

    (b) $\Theta(\log n)$

    (c) $\Theta(2^n)$

17. ¿Podemos saber cuál sería el elemento en posición k cuando ordenáramos un vector de n elementos sin tener que ordenarlo?

    (a) No. Debemos ordenarlo.

    (b) Sí, y el algoritmo es $\Omega(n)$ y $O(n^2)$, aunque la frecuencia de los casos peores disminuye muy rápidamente con n.

    (c) Sí, y el algoritmo es $\Omega(n \log n)$ y $O(n^2)$, aunque la frecuencia de los casos peores disminuye muy rápidamente con n.

18. Queremos imprimir los n enteros positivos que contiene un vector v, ordenados de menor a mayor. Suponed que el entero más grande que puede aparecer es m. Para ello hacemos un algoritmo que hace lo siguiente:

    a) crea un vector de enteros auxiliar (c) de tamaño ($m + 1$) y lo inicializa con ceros.

    b) para cada i desde 0 a $n - 1$ ejecuta `c[v[i]]++`

    c) para cada i desde 0 a m imprime `a[i]` veces el valor i.

    ¿Qué complejidad tiene este algoritmo en función de n?

    (a) $O(n \log n)$

    (b) $O(n)$

    (c) $O(n)$, pero este algoritmo no ordena bien en todos los casos

19. ¿Cuál de las siguientes recurrencias expresa la complejidad en el caso más desfavorable del algoritmo (visto en clase) que encuentra el k-ésimo elemento más grande de un vector?

    (a) $f(n) = f(n-1) + n$; $f(1) = 1$

    (b) $f(n) = 2f(n-1) + n$; $f(1) = 1$

    (c) $f(n) = f(n/2) + n$; $f(1) = 1$

20. ¿Cuál sería la complejidad **espacial** mínima de una función, con el siguiente perfil:

```
void add(
   const vector<int>&a,
   const vector<int>&b,
   vector<int>&r
);
```

    que suma los vectores `a` y `b` y devuelve el resultado en el vector `r`?

    (a) $O(n)$

    (b) $O(1)$

    (c) $O(0)$

21. Sea el vector $v[8] = \{8, 6, 4, 5, 4, 3, 2, 2\}$. Indica cuál de las siguientes opciones es cierta. (se asume la notación del lenguaje C/C++ en la que el primer elemento del vector está en la posición 0, es decir, en $v[0]$).

    (a) El vector v es un montículo máximo.

    (b) El vector v no es un montículo máximo porque el elemento v[2]=4 debe ser "hundido" (desplazado hacia la derecha)..

    (c) El vector v no es un montículo máximo porque el elemento v[3]=5 debe ser "flotado" (desplazado hacia la izquierda).

22. ¿Puede resolverse el problema del encaminamiento óptimo (versión general) usando una estrategia de divide y vencerás?

    (a) Sí. Además es muy poco probable que aparezcan subproblemas repetidos y, por tanto, no será efectivo recurrir a la programación dinámica

    (b) No, ya que el problema no tiene una subestructura óptima

    (c) Sí, aunque aparecerán muchos subproblemas repetidos. Será mucho mejor resolverlo usando programación dinámica

23. En un problema de minimización resuelto mediante ramificación y poda, una cota pesimista es ...

    (a) ... una cota inferior para el valor óptimo que a veces coincide con este.

    (b) Ninguna de las otras dos opciones es cierta.

    (c) ... una cota superior para el valor óptimo, pero que nunca coincide con este.

24. Se pretende resolver el problema del encaminamiento óptimo (versión general) mediante ramificación y poda. ¿Qué podemos asegurar si el subóptimo de partida coincide con la cota optimista del nodo inicial?

    (a) Que la cota optimista está mal calculada.

    (b) Que la cota pesimista está mal calculada.

    (c) Las otras dos opciones son ambas falsas.

25. Un algoritmo recursivo basado en el esquema divide y vencerás ...

    (a) ... nunca tendrá un coste temporal asintótico exponencial.

    (b) Las dos anteriores son verdaderas.

    (c) ... alcanza su máxima eficiencia cuando el problema de tamaño n se divide en a problemas de tamaño n/a.

26. La función $\texttt{met}(m, n)$, con $1 \leqslant m \leqslant n$, se puede calcular recursivamente siguiendo el siguiente esquema:

$$\texttt{met}(m,n) = \begin{cases} \texttt{ogw}(1,n) & \text{if } m = 1 \\ \min_{k=m-1}^{n-1} \texttt{met}(m-1,k) + \texttt{ogw}(k,n) & \text{if } n > m \end{cases}$$

    Estamos interesados en utilizar programación dinámica para acelerar el cálculo pero queremos usar la mínima memoria posible. ¿Qué estructura del almacén sería la más adecuada?

    (a) un vector (un índice)

    (b) una constante (ningún índice)

    (c) una tabla (dos índices)

27. ¿Cuál es el caso peor del algoritmo de ordenación Quicksort que toma el primer elemento como pivote?

    (a) Cuando el elemento pivote queda siempre en medio.

    (b) Cuando el elemento pivote queda siempre en uno de los dos extremos.

    (c) Cuando debemos hundir el pivote hasta el fondo del montículo.

28. De las siguientes expresiones, o bien dos son ciertas y una es falsa, o bien al contrario, una es cierta y dos son falsas. Marca la que en este sentido es diferente a las otras dos.

    (a) Si $f(n) \in O(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = \infty$

    (b) Si $f(n) \in \Theta(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = k$

    (c) Si $f(n) \in O(g(n))$ entonces $\lim_{n\to\infty}(f(n)/g(n)) = 0$

29. Tenemos un conjunto de n enteros positivos y queremos encontrar el subconjunto de tamaño m de suma mínima.

    (a) Para encontrar la solución habría que probar con todas las combinaciones posibles de m enteros, con lo que la técnica de ramificación y poda no aporta nada con respecto a vuelta atrás.

    (b) Una técnica voraz daría una solución óptima.

    (c) Lo más adecuado sería usar una técnica de ramificación y poda, aunque en el peor caso el coste temporal asintótico (o complejidad temporal) sería exponencial.

30. Se dispone de un conjunto de n valores numéricos dispuestos en forma de árbol binario y se desea obtener el valor de la suma de todos ellos. ¿Cuál es la complejidad temporal del mejor algoritmo que se puede escribir?

    (a) $O(\log n)$

    (b) $O(n \log n)$

    (c) $O(n)$

31. Sea $f(n) = n^2 + 4$. Dos de las tres afirmaciones siguientes prueban que $f(n) \in O(n^2)$. ¿Cuál es la que no?

    (a) Para todo $n > \sqrt{\frac{4}{c-1}}$, con $c > 1$, se cumple que $n^2 + 4 > cn^2$.

    (b) Para todo $n > 2$ se cumple que $n^2 + 4 < 2n^2$

    (c) Para todo $n < \sqrt{\frac{4}{c-1}}$, con $c > 1$, se cumple que $n^2 + 4 < cn^2$.

32. Queremos formar una suma M con el número mínimo posible de monedas tomadas (con repetición) de un conjunto de m monedas $C = \{c_1, c_2, \ldots c_m\}$. Una solución recursiva posible es la siguiente:

$$n_{\text{opt}}(M) = \begin{cases} 1 + \text{mín}_{1 \leqslant i \leqslant |C|}\, n_{\text{opt}}(M - c_i) & M > 0 \\ 0 & M = 0 \\ \infty & M < 0 \end{cases}$$

    ¿Sería posible convertir esta solución recursiva en un algoritmo de programación dinámica iterativa con complejidad temporal $O(M|C|)$?

    (a) Sí, ya que es posible calcular $n_{\text{opt}}(x)$ para cualquier $x > 0$ usando $n_{\text{opt}}(y)$ calculados para $y < x$.

    (b) No, pero sí que se puede añadir memoización al cálculo para evitar la repetición de cálculos.

    (c) No, no hay manera de evitar la repetición de cálculos y por tanto la complejidad temporal es prohibitiva.

33. Se debe implantar fibra óptica en una isla, en la que la población está dispersa en n pequeños núcleos de población, los cuales deben quedar conectados. Los ingenieros tienen a mano una tabla en la que para cada par de localidades (i, j), se indica el coste C_{ij} de tender una línea de fibra entre i y j. Las localidades pueden estar conectadas directamente o indirectamente a través de otras localidades; el coste de los repetidores que se deben instalar para las conexiones indirectas es despreciable comparado con el de las líneas. Indica cuál de las siguientes afirmaciones es correcta en cuanto al tendido óptimo (es decir, de coste mínimo total) de fibra óptica.

    (a) El tendido óptimo tiene siempre forma de árbol, con $n - 1$ tramos de línea y puede obtenerse mediante un algoritmo voraz.

    (b) El tendido óptimo tiene siempre una estructura radial, y existe un algoritmo muy eficiente para elegir el centro.

    (c) El tendido óptimo es siempre un recorrido en bus (es decir, con $n - 1$ tramos de línea) elegido mediante un algoritmo voraz.

34. Estamos trabajando en el problema del encaminamiento óptimo, pero ahora nos dicen que tenemos que tener en cuenta el ancho de banda de las tarjetas de red de los ordenadores, por lo que el tráfico pasará a calcularse como: $\sum_{i=1}^{n} c_i a_i d(i, p_i)$, donde c_i, a_i y p_i son respectivamente, la capacidad, el ancho de banda y la puerta de acceso más cercana al ordenador i. ¿Cuál de las siguientes frases es correcta?

    (a) El algoritmo para resolver el problema va a ser prácticamente igual, la complejidad no cambiará

    (b) Esto simplifica el problema y podrá resolverse mediante programación dinámica.

    (c) Al ser a_i un número real, la solución ya no podrá expresarse como un vector. Por tanto, el problema no podrá resolverse por ramificación y poda

35. Se pretende resolver el problema del encaminamiento óptimo (versión general) mediante ramificación y poda y para ello se hace uso de una cota que consiste en asumir que los restantes nodos aún no tratados no van a ser puerta de enlace. ¿Que podemos decir de esta cota?
    (a) Que es una cota optimista.
    (b) Las otras dos opciones son ambas falsas.
    (c) Que es una cota pesimista solo si ya se han colocado exactamente m puertas.

36. ¿Para cuál de estos tres algoritmos de ordenación la complejidad temporal en el caso peor es $O(n^2)$?
    (a) Quicksort
    (b) Heapsort
    (c) Mergesort

37. Tenemos dos algoritmos de ordenación. El algoritmo A pertenece a $\Theta(n \log n)$ y el B pertenece a $\Theta(n^2)$. ¿Es cierto que el algoritmo A va a ser siempre más rápido que el B?
    (a) Sí, ya que n^2 crece mas lento que $n \log n$
    (b) No
    (c) Sí, ya que $n \log n$ crece mas lento que n^2

38. Se pretende resolver la versión simplificada del problema del encaminamiento óptimo mediante la técnica "divide y vencerás". ¿Se podría obtener la mejor disposición de las puertas?
    (a) No, ya que no cumple el teorema de reducción.
    (b) Sí, pero a costa de una complejidad temporal prohibitiva.
    (c) No, ya que no se cumple la propiedad "subestructura óptima".