Análisis y Diseño de Algoritmos
5 de julio de 2012
Examen final (duración: 120 min.)

Instrucciones:

- No podéis consultar ningún material ni hablar con nadie.
- Poned vuestros datos en la hoja de respuestas de lectura óptica que se os entregará.
- **No olvidéis indicar la modalidad del examen en la hoja de respuestas.**
- Elegid en cada pregunta la opción que creáis correcta, y marcadla cómo se indica en la hoja de respuestas.
- La nota del examen se calcula así:

$$\text{nota} = 10 \times (\text{aciertos} - \frac{1}{2}\text{errores})/\text{preguntas}$$

- Las respuestas en blanco ni restan ni suman puntos.
- Hay que entregar tanto las hojas de preguntas como la de respuestas (aunque las hojas de preguntas las podéis marcar).
- Tenéis 120 min. para hacer el examen.
- Si tenéis alguna duda, levantad la mano y un profesor irá a atenderos.

Modalidad 2

Preguntas:

1. Los algoritmos de ordenación *Quicksort* y *Mergesort* tienen en común...
   (a) ... que ordenan el vector sin usar espacio adicional.
   (b) ... que se ejecutan en tiempo $O(n)$.
   (c) ... que aplican la estrategia de *divide y vencerás*.
2. ¿Cuál es la diferencia principal entre una solución de *vuelta atrás* y una solución de *ramificación y poda* para el problema de la mochila?
   (a) El coste asintótico en el caso peor.
   (b) El hecho que la solución de *ramificación y poda* puede empezar con una solución subóptima voraz y la de *vuelta atrás* no.
   (c) El orden de exploración de las soluciones.

3. 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.

4. Di cuál de estos resultados de coste temporal asintótico es falsa:

   (a) La ordenación de un vector usando el algoritmo *Quicksort* requiere en el peor caso un tiempo en $\Omega(n^2)$.

   (b) La ordenación de un vector usando el algoritmo *Mergesort* requiere en el peor caso un tiempo en $\Omega(n^2)$.

   (c) La búsqueda binaria en un vector ordenado requiere en el peor caso un tiempo en $O(\log n)$.

5. En el problema del viajante de comercio (*travelling salesman problem*) queremos listar todas las soluciones factibles.

   PUEDE SER: [A] o [B]

   (a) Lo más adecuado sería usar una técnica de ramificación y poda ya que es muy importante el orden en el que se exploran las soluciones parciales.

   (b) El orden en el que se exploran las soluciones parciales no es relevante; por ello, la técnica ramificación y poda no aporta nada con respecto a vuelta atrás.

   (c) Lo más importante es conseguir una cota pesimista muy adecuada. Las diferencias entre ramificación y poda y vuelta atrás son irrelevantes en este caso.

6. La complejidad temporal (o coste temporal asintótico) en el mejor de los casos...

   (a) ... es una función de la talla, o tamaño del problema, que tiene que estar definida para todos los posibles valores de ésta.

   (b) ... es el tiempo que tarda el algoritmo en resolver la talla más pequeña que se le puede presentar.

   (c) Las dos anteriores son verdaderas.

7. Tenemos n substancias diferentes en polvo y queremos generar todas las distintas formas de mezclarlas de forma que el peso total no supere un gramo. Como la balanza que tenemos solo tiene una precisión de 0.1 gramos, no se considerarán pesos que no sean múltiplos de esta cantidad. Queremos hacer un programa que genere todas las combinaciones posibles.

   (a) No hay ningún problema en usar una técnica de vuelta atrás.

   (b) No se puede usar vuelta atrás porque las decisiones no son valores discretos.

   (c) No se puede usar vuelta atrás porque el número de combinaciones es infinito.

PUEDE SER: [A] o [C]

8. 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 (o complejidad temporal) exponencial.

   (c) Las dos anteriores son verdaderas.

9. Dado un problema de optimización, ¿cuándo se puede aplicar el método de vuelta atrás?

   (a) Es condición necesaria (aunque no suficiente) que el dominio de las decisiones sea discreto o discretizable.

   (b) Es condición necesaria y suficiente que el dominio de las decisiones sea discreto o discretizable.

   (c) No sólo es condición necesaria que el dominio de las decisiones sea discreto o discretizable; además, debe cumplirse que se puedan emplear mecanismos de poda basados en la mejor solución hasta el momento.

10. ¿Cuál de estas tres expresiones es cierta?

    (a) $O(2^{\log(n)}) \subset O(n^2) \subset O(2^n)$

    (b) $O(n^2) \subset O(2^{\log(n)}) \subset O(2^n)$

    (c) $O(n^2) \subset O(2^{\log(n)}) \subset O(2^n)$

11. Sea $f(n)$ la solución de la relación de recurrencia $f(n) = 2f(n/2) + n$; $f(1) = 1$. Indicad cuál de estas tres expresiones es cierta:

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

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

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

12. Indicad cuál de estas tres expresiones es falsa:

    (a) $\Theta(n/2) = \Theta(n)$

    (b) $\Theta(n) \subset O(n)$

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

13. Indica cuál es el coste temporal asintótico (o complejidad temporal), en función de n, del programa siguiente:

```
s=0; for(i=0;i<n;i++) for(j=i;j<n;j++) s+=n*i+j;
```

(a) Es $O(n^2)$ pero no $\Omega(n^2)$.

(b) Es $\Theta(n^2)$

(c) Es $\Theta(n)$

14. Un programa con dos bucles anidados uno dentro del otro, cada uno de los cuales hace aproximadamente n iteraciones, tarda un tiempo

(a) $O(n^2)$

(b) $O(2^n)$

(c) $O(n)$

15. La eficiencia de los algoritmos voraces se basa en...

(a) ... el hecho de que, con antelación, las posibles decisiones se ordenan de mejor a peor.

(b) ... el hecho de que las decisiones tomadas no se reconsideran.

(c) En el esquema voraz no se puede hablar de eficiencia puesto que a menudo no resuelve el problema.

16. 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.

17. Sea $f(n)$ la solución de la relación de recurrencia $f(n) = 2f(n-1) + 1$; $f(1) = 1$. Indicad cuál de estas tres expresiones es cierta:

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

(b) $f(n) \in \Theta(2^n)$

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

18. Pertenece $3n^2 + 3$ a $O(n^3)$?

(a) No.

(b) Sólo para $c = 1$ y $n_0 = 5$

(c) Sí.

19. Las relaciones de recurrencia...

(a) ... aparecen sólo cuando la solución sea del tipo *divide y vencerás*.

(b) ... expresan recursivamente el coste temporal de un algoritmo.

(c) ... sirven para reducir el coste temporal de una solución cuando es prohibitivo.

20. El coste temporal de un algoritmo se ajusta a la siguiente ecuación de recurrencia:

$$T(n) \quad \begin{cases} 1 & n-0 \\ n+\sum_{j=0}^{n-1} T(j) & n>1 \end{cases}$$

¿qué coste temporal asintotico (o complejidad temporal) tendrá el algoritmo?

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

(b) $O(n^2)$

(c) $O(2^n)$

21. La versión de *Quicksort* que utiliza como pivote el elemento del vector que ocupa la posición central...

(a) ... no presenta caso mejor y peor para instancias del mismo tamaño.

(b) ... se comporta mejor cuando el vector ya está ordenado.

(c) ... se comporta peor cuando el vector ya está ordenado.

22. ¿Cual es el coste espacial asintotico del siguiente algoritmo?

```
int f( int n ) {
  int a = 1, r = 0;
  for( int i = 0; i
    r = a + r;
    a = 2*r;
  }
return r;
}
```

PUEDE SER: [A] o [C]

(a) $O(1)$

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

(c) $O(n)$

23. De los problemas siguientes, indicad cuál no se puede tratar eficientemente como los otros dos:

NO SEGURO DEL 100%

(a) El problema de cortar un tubo de forma que se obtenga el máximo beneficio posible

(b) El problema del cambio, o sea, el de encontrar la manera de entregar una cantidad de dinero usando las mínimas monedas.

(c) El problema del viajante de comercio

24. ¿Qué algoritmo es asintóticamente más rápido, el *quicksort* o el *mergesort*?

    (a) como su nombre indica, el *quicksort*.

    (b) son los dos son igual de rápidos, ya que el coste temporal asintótico de ambos es $O(n \log(n))$.

    (c) el *mergesort* es siempre más rápido o igual (salvo una constante) que el *quicksort*.

25. El coste temporal del algoritmo de ordenación por inserción es...

    (a) ... $O(n^2)$.

    (b) ... $O(n)$.

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

26. Los algoritmos de programación dinámica hacen uso...

    (a) ...de que la solución óptima se puede construir añadiendo el componente óptimo de los restantes, uno a uno.

    (b) ...de que se puede ahorrar esfuerzo guardando los resultados de esfuerzos anteriores.

    (c) ...de una estrategia trivial consistente en examinar todas las soluciones posibles.

27. ¿Cuál de estos tres problemas de optimización no tiene una solución voraz (*greedy*) que sea óptima?

    (a) El problema de la mochila continua o con fraccionamiento.

    (b) El problema de la mochila discreta.

    (c) El árbol de cobertura de coste mínimo de un grafo conexo.

28. El problema de la función compuesta mínima consiste en encontrar, a partir de un conjunto de funciones dadas, la secuencia mínima de composiciones de éstas que permita transformar un número n en otro m. Se quiere resolver mediante ramificación y poda. ¿Cuál sería la forma más adecuada de representar las posibles soluciones?

    (a) Mediante un vector de booleanos.

    (b) Mediante un vector de reales.

    (c) Este problema no se puede resolver usando ramificación y poda si no se fija una cota superior al número total de aplicaciones de funciones.

29. ¿Cuál de estas tres expresiones es falsa?

    PUEDE SER: [A] o [C]

    (a) $2n^2 + 3n + 1 \in O(n^3)$

    (b) $n + n\log(n) \in \Omega(n)$

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

30. Si el coste temporal de un algoritmo es $T(n)$, ¿cuál de las siguientes situaciones es imposible?

    (a) $T(n) \in O(n)$ y $T(n) \in \Theta(n)$

    (b) $T(n) \in \Omega(n)$ y $T(n) \in \Theta(n^2)$

    (c) $T(n) \in \Theta(n)$ y $T(n) \in \Omega(n^2)$

31. El coste temporal asintótico de insertar un elemento en un vector ordenado de forma que continue ordenado es...

    PUEDE SER: [A] o [B]

    (a) ... $O(n)$.

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

    (c) ... $O(n^2)$.

32. ¿Qué nos proporciona la media entre el coste temporal asintótico (o complejidad temporal) en el peor caso y el coste temporal asintótico en el mejor caso?

    PUEDE SER: [A] o [C]

    (a) El coste temporal promedio.

    (b) El coste temporal asintótico en el caso medio.

    (c) Nada de interés.

33. El coste temporal asintótico del programa

```
s=0;for(i=0;i<n;i++) for(j=i;j<n;j++) s+=i*j;
```

    y el del programa

```
s=0; for(i=0;i<n;i+ ) for(j=0;j<n;j  ) s+=i*i*j;
```

    son...

    (a) ... el del primero, menor que el del segundo.

    (b) ... el del segundo, menor que el del primero.

    (c) ... iguales.

34. Se desea obtener todas las permutaciones de una lista compuesta por n elementos. ¿Qué esquema es el más adecuado?

    (a) Divide y vencerás, puesto que la división en sublistas se podría hacer en tiempo constante.

    (b) Ramificación y poda, puesto que con buenas funciones de cota es más eficiente que vuelta atrás.

    (c) Vuelta atrás, es el esquema más eficiente para este problema.

35. En ausencia de cotas optimistas y pesimistas, la estrategia de vuelta atrás...

    PUEDE SER: [B] o [C]

    (a) ... no se puede usar para resolver problemas de optimización.

    (b) ... no recorre todo el árbol si hay manera de descartar subárboles que representan conjuntos de soluciones no factibles.

    (c) ... debe recorrer siempre todo el árbol.

36. La solución recursiva ingenua a un determinado problema de optimización muestra estas dos características: por un lado, se basa en obtener soluciones óptimas a problemas parciales más pequeños, y por otro, estos subproblemas se resuelven más de una vez durante el proceso recursivo. Este problema es candidato a tener una solución alternativa basada en...

    (a) ... un algoritmo del estilo de *divide y vencerás*.

    (b) ... un algoritmo de programación dinámica.

    (c) ... un algoritmo voraz.

37. La función γ de un número semientero positivo (un número es semientero si al restarle 0.5 es entero) se define como:

```
double gamma( double n ) {    Se asume n>=0.5 y n-0.5 enter
  if( n == 0.5 )
    return sqrt(PI);
  return n * gamma( n - 1);
}
```

PUEDE SER: [A] o [B]

¿Se puede calcular usando programación dinámica iterativa?

    (a) Sí.

    (b) No, ya que el índice del almacén sería un número real y no entero.

    (c) No, ya que no podríamos almacenar los resultados intermedios en el almacén.

38. Decid cuál de estas tres es la cota optimista más ajustada al valor óptimo de la mochila discreta:

    (a) El valor de la mochila continua correspondiente .

    (b) El valor de la mochila discreta que se obtiene usando un algoritmo voraz basado en el valor específico de los objetos.

    (c) El valor de una mochila que contiene todos los objetos aunque se pase del peso máximo permitido.

39. Un tubo de n centímetros de largo se puede cortar en segmentos de 1 centímetro, 2 centímetros, etc. Existe una lista de los precios a los que se venden los segmentos de cada longitud. Una de las maneras de cortar el tubo es la que más ingresos nos producirá. Se quiere resolver el problema mediante vuelta atrás. ¿cual sería la forma más adecuada de representar las posibles soluciones?

NO SEGURO DEL 100%

    (a) un vector de booleanos.

    (b) un par de enteros que indiquen los cortes realizados y el valor acumulado.

    (c) una tabla que indique, para cada posición donde se va a cortar, cada uno de los posibles valores acumulados.

40. Cuando la descomposición de un problema da lugar a subproblemas de tamaño similar al original, muchos de los cuales se repiten, ¿qué esquema es a priori más apropiado?

(a) Divide y vencerás.

(b) Programación dinámica.

(c) Ramificación y poda.

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

Seleccione una:

a. ... será más eficiente cuanto más equitativa sea la división en subproblemas

b. Las demás opciones son verdaderas.

c. ... nunca tendrá una complejidad exponencial.

La respuesta correcta es: ... será más eficiente cuanto más equitativa sea la división en subproblemas.

Indicad cuál de estas tres expresiones es falsa:

Seleccione una:

a. $\Theta(n/2) = \Theta(n)$

b. $\Theta(n) \subseteq \Theta(n^2)$

c. $\Theta(n) \subseteq O(n)$

La respuesta correcta es: $\Theta(n) \subseteq \Theta(n^2)$

¿Cuál de estas tres expresiones es falsa?

Seleccione una

a. $3n^2+1 \in O(n^3)$

b. $n+n\log(n) \in \Omega(n)$

c. $n+n\log(n) \in \Theta(n)$ ✓

Indica cuál es la complejidad en el peor caso de la función replace

```
unsigned bound( const vector<int> &v
    for( unsigned i    ; i < v.size(); i++ )
        if( v[i] == '0'
            return i;
    return v.size();


v   replace( ve   <int>   , int
    for( unsigned i = 0; i < b  nd ); i++
        v[i] = c;
```

Seleccione una

a $O(n \log n)$

b $O(n^2)$ ✓

c $O(n)$

¿Cuál es la complejidad temporal de la siguiente función recursiva?

```
unsigned desperdicio (unsigned n){
if (n<=1)
   return 0;
unsigned sum = desperdicio (n/2) + desperdicio (n/2);
for (unsigned i=1; i<n-1; i++)
    for (unsigned j=1; j<=i; j++)
        sum+=1;
return sum;
```

Seleccione una:

a $\Theta(n^2)$ ✓

b $\Theta(2^n)$

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

Sea $f(n)$ la solución de la relación de recurrencia $f(n)=2f(n/2)+1;\ f(1)=1$ Indicad cuál de estas tres expresiones es cierta:

Seleccione una:

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

b. $f(n)\in\Theta(n^2)$

c. $f(n)\in\Theta(n\log(n))$

La respuesta correcta es: $f(n)\in\Theta(n)$

Considerad estos dos fragmentos:

```
s=0;for(i=0;i<n;i++) s+=i;
```

y

```
s=0;for(i=0;i<n;i++) if (a[i]!=0) s+=i;
```

y un array `a[i]` de números enteros. Indicad cuál de estas tres afirmaciones es cierta:

Seleccione una:

a. El coste temporal asintótico del primer programa en el caso peor es más alto que en el segundo.

b. El coste temporal asintótico, tanto en el caso mejor como en el caso peor, de los dos programas es el mismo.

c. El coste temporal asintótico del segundo programa en el caso peor es más alto que en el primero.

La respuesta correcta es: El coste temporal asintótico, tanto en el caso mejor como en el caso peor, de los dos programas es el mismo.

Indica cuál es la complejidad, en función de n, del fragmento siguiente:

```
int a = 0;
for( int i = 0; i < n; i++ )
    for(  int j = i; j > 0; j -=2 )
        a += A[i][j];
```

Seleccione una:

a. $O(n\log n)$

b. $O(n)$

c. $O(n^2)$

La respuesta correcta es: $O(n\log n)$

Indica cuál es la complejidad en función de n, donde k es una constante (no depende de n), del fragmento siguiente :

```
for( int i = k; i < n - k; i++){
    A[i] = 0;
    for( int j = i - k; j < i + k; j++ )
        A[i] += B[j];
```

Seleccione una:

a. $O(n)$

b. $O(n \log n)$

c. $O(n^2)$

La respuesta correcta es: $O(n)$

Pertenece $3n^2+3$ a $O(n^3)$?

Seleccione una:

a. Sólo para $c = 1$ y $n_0 = 5$.

b. No.

c. Sí.

La respuesta correcta es: Sí.

La complejidad temporal en el mejor de los casos...

Seleccione una:

a. Las demás opciones son verdaderas.

b. ... es el tiempo que tarda el algoritmo en resolver la talla más pequeña que se le puede presentar.

c. ... es una función de la talla que tiene que estar definida para todos los posibles valores de ésta.

La respuesta correcta es: .. es una función da la talla que tiene que estar definida para todos los posibles valores de ésta.

La versión de *Quicksort* que utiliza como pivote la mediana del vector ...

Seleccione una:

a. ... se comporta mejor cuando el vector ya está ordenado.

b. ... se comporta peor cuando el vector ya está ordenado.

c. ... El hecho de que el vector estuviera previamente ordenado o no no influye en la complejidad temporal de este algoritmo.

La respuesta correcta es: ... El hecho de que el vector estuviera previamente ordenado o no, no influye en la complejidad temporal de este algoritmo.

Dada la siguiente relación de recurrencia, ¿Qué cota es verdadera?

$$f(n)=\begin{cases}1 & n=1\\ \sqrt{n}+3f(n/3) & n>1\end{cases}$$

Seleccione una:

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

b. $f(n)\in\Theta(n^3)$

c. $f(n)\in\Theta(\sqrt{n}\log n)$

La respuesta correcta es: $f(n)\in\Theta(n)$

Un problema de tamaño n puede transformarse en tiempo $O(n^2)$ en nueve de tamaño $n/3$, por otro lado, la solución al problema cuando la talla es 1 requiere un tiempo constante.

¿cual de estas clases de coste temporal asintótico es la más ajustada?

Seleccione una:

a. $O(n^2)$

b. $O(n^2\log n)$

c. $O(n\log n)$

La respuesta correcta es: $O(n^2\log n)$

Indica cuál es la complejidad en función de n del siguiente fragmento de código

```
s=1; i r(i= ;i<n;i++) r(j=i;j<n; ++  s+=i ;
```

Seleccione una

a. $\Theta(n^2)$ ✓

b. $O(n^2)$ pero no $\Omega(n^2)$

c. $\Theta(n$

Sea $f(n)$ la solución de la relación de recurrencia $f(n)=2f(n-1)+1\ f(1)-1$ Indicad cuál de estas tres expresiones es cierta

Seleccione una

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

b. $f(n)\in\Theta(2^n)$ ✓

c. $f(n)\in\Theta(n^2)$

Un programa con dos bucles anidados uno dentro del otro. El primero haca n iteraciones aproximadamente y el segundo la mitad tarda un tiempo

Seleccione una

a $O(n\log n)$

b $O(n^2)$ ✓

c. $O(n\sqrt{n})$

Un problema de tamaño n puede transformarse en tiempo $O(n^2)$ en nueve de tamaño $n/3$ por otro lado. la solución al problema cuando la talla es 1 requiere un tiempo constante

¿cual de estas clases de coste temporal asintótico es la más ajustada?

Seleccione una:

a $O(n\log n)$

b. $O(n^2\log n)$ ✓

c $O(n^2)$

¿Cuál es la complejidad temporal de la siguiente función recursiva?

```
unsigned desperdicio (unsigned n){
if (n<=1)
    return 0;
unsigned sum = desperdicio (n/2) - desperdicio (n/2);
for (unsigned i=1; i<n-1; i--)
    for (unsigned j=i; j<=i; j--)
        for (unsigned k=1; k<=j; k++)
            sum+=i*j*k;
return sum;
}
```

Selecc one una

a $\Theta(2^n)$

b $\Theta(n^3\log n)$

c $\Theta(n^3)$ ✓

Los algoritmos do ordenación *Quicksort* y *Mergesort* tienen en común

Seleccione una

a. . que se ejecutan en tiempo $O(n)$

b que ordenan el vector sin usar espacio adicional

c. .. que aplican la estrategia de *divide y vencerás.* ✓

Indica cuál es la complejidad de la funcion siguiente

```
unsigned sum( const mat &A )    // A es una matriz cuadrada
unsigned d = A.n_rows();
unsigned a = 0;
    for( unsigned i = 0; i < d; i-- )
        for( unsigned j = 0; j < d; j-- )
            a -= A(i,j);
return a;
}
```

Seleccione una

a $O(n^2)$

b. $O(n)$ ✓

c $O(n \log n)$

Indicad cuál de estas tres expresiones es falsa

Seleccione una

a $\Theta(n/2) = \Theta(n)$

b $\Theta(n) \subseteq O(n)$

c $\Theta(n) \subseteq \Theta(n^2)$ ✓

¿Cuál de estos tres problemas de optimización no tiene o no se le conoce una solución voraz óptima?

Seleccione una

a El árbol de cobertura de coste mínimo de un grafo conexo

b El problema de la mochila discreta o sin fraccionamiento ✓

c El problema de la mochila continua o con fraccionamiento

Los algoritmos de programación dinámica hacen uso

Seleccione una

a de que la solución óptima se puede construir añadiendo a la solución el elemento óptimo de los elementos restantes uno a uno

b de que se puede ahorrar cálculos guardando resultados anteriores en un almacen ✓

c de una estrategia trivial consistente en examinar todas las soluciones posibles

Cuando se calculan los coeficientes binomiales usando la recursión $\binom{n}{r}=\binom{n-1}{r}+\binom{n-1}{r-1}$ con $\binom{n}{0}=\binom{n}{n}=1$ qué problema se da y cómo se puede resolver?

Seleccione una

a La recursión puede ser infinita y por tanto es necesario organizarla según el esquema iterativo de programación dinamica

b Se repiten muchos cálculos y ello se puede evitar haciendo uso de una estrategia voraz

c Se repiten muchos cálculos y ello se puede evitar usando programación dinámica ✓

Sea $f(n)$ la solución de la relación de recurrencia $f(n)=2f(n/2)+n \quad f(1)=1$ Indicad cuál de estas

Seleccione una:

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

b $f(n)\in\Theta(n)$

c. $f(n)\in\Theta(n\log(n))$ ✓

Para que la complejidad de un algoritmo presente caso mejor y peor distintos ...

Seleccione una.

a. es condición necesaria y suficiente que existan instancias distintas del problema con el mismo tamaño.

b. ... es condición necesaria que existan instancias distintas del problema con el mismo tamaño. ✓

c es condición suficiente que existan instancias distintas del problema con el mismo tamaño

Un problema de tamaño n puede transformarse en tiempo $O(n)$ en siete de tamaño $n/7$. por otro lado, la solución al problema cuando la talla es 1 requiere un tiempo constante

¿cual de estas clases de coste temporal asintótico es la más ajustada?

Seleccione una:

a $O(n^2)$

b $O(n)$

c $O(n\log n)$ ✓

Indicad cuál de estas tres expresiones es cierta

Seleccione una

a. $O(n^2) \subset O(2^{\log(n)}) \subset O(2^n)$

b. $O(n^2) \subset O(2^{\log(n)}) \subseteq O(2^n)$

c. $O(2^{\log(n)}) \subset O(n^2) \subset O(2^n)$ ✓

La complejidad temporal en el mejor de los casos de un algoritmo recursivo..

Seleccione una:

a. coincide con el valor del caso base de la ecuación de recurrencia que expresa la complejidad temporal del algoritmo

b. Las demás opciones son falsas ✓

c. siempre coincidirá con la complejidad temporal de las instancias que están en el caso base del algoritmo recursiva

Considerad la función siguiente:

```
int M( int i, int f) {
    if ( i == f )
        return i;
    else {
        e = v[ M( i, (i+f)/2 ) ];
        f = v[ M( (i+f)/2+1, f ) ];
        if (e<f)
            return e;
        else
            return f;
    }
}
```

Si la talla del problema viene dada por $n = f - i + 1$, ¿cuál es el coste temporal asintótico en el supuesto de que n sea una potencia de 2?

Seleccione una

a. $O(n)$ ✓

b. $O(n^2)$

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

El coste temporal asintótico del fragmento

```
s=0; for(i=0;i<n;i++) for(j=i;j<n;j++) s+=i*j;
```

y el del fragmento

```
s=0; for(i=0;i<n;i++) for(j=0;j<n;j++) s+=i*i*j;
```

son ...

Seleccione una:

- a. ... iguales. ✓
- b. ... el del segundo, menor que el del primero.
- c. ... el del primero, menor que el del segundo.

Dada la siguiente relación de recurrencia, ¿Qué cota es verdadera?

$$f(n) = \begin{cases} 1 & n = 1 \\ n^2 + 3f(n/3) & n > 1 \end{cases}$$

Seleccione una:

SIN RESPUESTA

- a. $f(n) \in \Theta(n^2)$
- b. $f(n) \in \Theta(n^2 \log n)$
- c. $f(n) \in \Theta(n)$

La versión de *Quicksort* que utiliza como pivote el elemento del vector que ocupa la primera posición ...

Seleccione una:

- a. ... se comporta mejor cuando el vector ya está ordenado.
- b. ... se comporta peor cuando el vector ya está ordenado. ✓
- c. ... El hecho de que el vector estuviera previamente ordenado o no, no influye en la complejidad temporal de este algoritmo.

La versión de *Quicksort* que utiliza como pivote el elemento del vector que ocupa la posición central ...

Seleccione una:

- a. ... se comporta mejor cuando el vector ya está ordenado ✓
- b. ... se comporta peor cuando el vector ya está ordenado.
- c. ... no presenta casos mejor y peor distintos para instancias del mismo tamaño

Dada la siguiente relación de recurrencia, ¿Qué cota es verdadera?

$$f(n) = \begin{cases} 1 & n = 1 \\ n + 3f(n/3) & n > 1 \end{cases}$$

Seleccione una:

- a. $f(n) \in \Theta(n \log n)$ ✓
- b. $f(n) \in \Theta(n^3)$
- c. $f(n) \in \Theta(n)$

Cuando se resuelve usando un algoritmo de *vuelta atrás* un problema de n decisiones, en el que siempre hay como mínimo dos opciones para cada decisión, ¿cuál de las siguientes complejidades en el caso peor es la mejor que nos podemos encontrar?

Seleccione una:

- a $O(2^n)$ ✓
- b $O(n!)$
- c $O(n^2)$

Al resolver el problema del viajante de comercio mediante *vuelta atrás* y asumiendo un grafo de n vértices totalmente conexo, ¿cuál de estas es una buena cota pesimista al iniciar la búsqueda?

Seleccione una

a Se multiplica n por la distancia de la arista más corta que nos queda por considerar

b Se ordenan las aristas restantes de menor a mayor distancia y se calcula la suma de las n aristas más cortas

c. Se resuelve el problema usando un algoritmo voraz que añade cada vez al camino el vértice más cercano al último añadido ✓

Se desea obtener todas las permutaciones de una lista compuesta por n elementos ¿Qué esquema es el más adecuado?

Seleccione una

a *Ramificación y poda* puesto que con buenas funciones de cota es más eficiente para este problema que *vuelta atrás*

b *Divide y vencerás* puesto que la división en sublistas se podría hacer en tiempo constante

c *Vuelta atrás* para este problema no hay un esquema más eficiente ✓

La complejidad en el mejor de los casos de un algoritmo de *ramificación y poda*

Seleccione una

a ... es siempre exponencial con el número de decisiones a tomar.

b. ... suele ser polinómica con el número de alternativas por cada decisión

c. ... puede ser polinómica con el número de decisiones a tomar. ✓

La complejidad en el peor de los casos de un algoritmo de *ramificación y poda*

Seleccione una

a ... puede ser exponencial con el número de alternativas por cada decisión

b ... puede ser polinómica con el número de decisiones a tomar

c ... es exponencial con el número de decisiones a tomar ✓

La estrategia de *ramificación y poda* genera las soluciones posibles al problema mediante ...

Seleccione una:

a. ... un recorrido guiado por estimaciones de las mejores ramas del árbol que representa el espacio de soluciones ✓

b. ... un recorrido en profundidad del árbol que representa el espacio de soluciones.

c. ... un recorrido en anchura del árbol que representa el espacio de soluciones.

¿Para qué sirven las cotas pesimistas en *ramificación y poda*?

Seleccione una

a Para tener la certeza de que la cota optimista está bien calculada

b Para descartar nodos basándose en la preferencia por algún otro nodo ya completado

c Para descartar nodos basándose en el beneficio esperado ✓

En los algoritmos de *ramificación y poda*, ¿el valor de una cota pesimista es mayor que el valor de una cota optimista? (entendiendo que ambas cotas se aplican sobre el mismo nodo)

Seleccione una:

a. En general sí, si se trata de un problema de maximización, aunque en ocasiones ambos valores pueden coincidir.

b. En general sí, si se trata de un problema de minimización, aunque en ocasiones ambos valores pueden coincidir. ✓

c. No, nunca es así.

En los algoritmos de *ramificación y poda*, ¿el valor de una cota pesimista es menor que el valor de una cota optimista? (entendiendo que ambas cotas se aplican sobre el mismo nodo)

Seleccione una

a. En general sí, si se trata de un problema de minimización, aunque en ocasiones ambos valores pueden coincidir

b. En general sí, si se trata de un problema de maximización, aunque en ocasiones ambos valores pueden coincidir ✓

c. Sí, siempre es así.

En los algoritmos de *ramificación y poda* ...

Seleccione una:

a. El uso de cotas pesimistas sólo resulta eficaz cuando se dispone de una posible solución de partida

b. Una cota optimista es necesariamente un valor insuperable, de no ser así se podría podar el nodo que conduce a la solución óptima. ✓

c. Una cota optimista es necesariamente un valor alcanzable, de no ser así no está garantizado que se encuentre la solución óptima.

La ventaja de la estrategia *ramificación y poda* frente a *vuelta atrás* es que la primera genera las soluciones posibles al problema mediante ...

Seleccione una:

a. ... un recorrido guiado por una cola de prioridad de donde se extraen primero los nodos que representan los subárboles más prometedores del espacio de soluciones.

b. Las otras dos opciones son verdaderas. ✓

c. ... un recorrido guiado por estimaciones de las mejores ramas del árbol que representa el espacio de soluciones.

¿Cuál es la diferencia principal entre una solución de *vuelta atrás* y una solución de *ramificación y poda* para el problema de la mochila?

Seleccione una:

a. El orden de exploración de las soluciones. ✓

b. El coste asintótico en el caso peor.

c. El hecho que la solución de *ramificación y poda* puede empezar con una solución subóptima voraz y la de *vuelta atrás* no.

Tratándose de un problema de optimización, en la lista de nodos vivos de *ramificación y poda* ...

Seleccione una:

a. ... sólo se introducen nodos prometedores, es decir, nodos que pueden mejorar la mejor solución que se tiene en ese momento.

b. ... puede haber nodos que no son prometedores.

c. Las otras dos opciones son ciertas. ✓

Cuando resolvemos un problema mediante un esquema de ramificación y poda

Seleccione una

a. los valores entre los cuales se elige en cada una de las decisiones tienen que formar un conjunto finito ✓

b. las decisiones sólo pueden ser binarias

c. los valores entre los cuales se elige en cada una de las decisiones pueden formar un conjunto infinito

La estrategia de ramificación y poda necesita cotas pesimistas

Seleccione una

a para decidir el orden de visita de los nodos del árbol de soluciones

b sólo si se usa para resolver problemas de optimización ✓

c para determinar si una solución es factible

El uso de funciones de cota en ramificación y poda …

Seleccione una:

a. … transforma en polinómicas complejidades que antes eran exponenciales.

b. … puede reducir el número de instancias del problema que pertenecen al caso peor. ✓

c. … garantiza que el algoritmo va a ser más eficiente ante cualquier instancia del problema.

En la estrategia de *ramificación y poda*

Seleccione una:

a cada nodo tiene su propia cota pesimista y también su propia cota optimista ✓

b cada nodo tiene su propia cota pesimista, la cota optimista sin embargo. es común para todos los nodos.

c. ... cada nodo tiene su propia cota optimista, la cota pesimista sin embargo, es común para todos los nodos.

Si para resolver un mismo problema usamos un algoritmo de *vuelta atrás* y lo modificamos mínimamente para convertirlo en un algoritmo de *ramificación y poda*, ¿qué cambiamos realmente?

Seleccione una:

a. La comprobación de las soluciones factibles: en *ramificación y poda* no es necesario puesto que sólo genera nodos factibles ✗

b. Cambiamos la función que damos a la cota pesimista.

c. Aprovechamos mejor las cotas optimistas.

Si para resolver un mismo problema usamos un algoritmo de ramificación y poda y lo modificamos mínimamente para convertirlo en un algoritmo de vuelta atrás ¿qué cambiamos realmente?

Seleccione una

a. Cambiamos la función que damos a la cota pesimista

b. Provocamos que las cotas optimistas pierdan eficacia

c. Sería necesario comprobar si las soluciones son factibles o no puesto que ramificación y poda solo genera nodos factibles

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

Seleccione una:

a. Las dos anteriores son verdaderas.

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

c. pueden eliminar soluciones parciales que son factibles. ✓

En ausencia de cotas optimistas y pesimistas, la estrategia de *vuelta atrás* ...

Seleccione una:

a ... no se puede usar para resolver problemas de optimización.

b ... debe recorrer siempre todo el árbol.

c. ... no recorre todo el árbol si hay manera de descartar subárboles que representan conjuntos de soluciones no factibles. ✓

La estrategia de *vuelta atrás* es aplicable a problemas de selección y optimización en los que:

Seleccione una:

a. El espacio de soluciones puede ser tanto finito como infinito pero en este último caso debe ser al menos numerable

b. El espacio de soluciones es un conjunto infinito.

c. El espacio de soluciones es un conjunto finito. ✓

Decid cuál de estas tres es la cota optimista que poda más eficientemente cuando se usa la estrategia de vuelta atrás para resolver el problema de la mochila

Seleccione una

a El valor de la mochila discreta que se obtiene usando un algoritmo voraz basado en el valor específico de los objeto

b El valor de una mochila que contiene todos los objetos aunque se pase del peso máximo permitido

c El valor óptimo de la mochila continua correspondiente

Decid cuál de estas tres no sirve como cota optimista para obtener el valor óptimo de la mochila discreta:

Seleccione una:

a. El valor de una mochila que contiene todos los objetos aunque se pase del peso máximo permitido.

b. El valor de la mochila discreta que se obtiene usando un algoritmo voraz basado en el valor específico de los objetos ✓

c. El valor de la mochila continua correspondiente.

Decid cuál de estas tres es la cota pesimista más ajustada al valor óptimo de la mochila discreta:

Seleccione una:

a. El valor de la mochila discreta que se obtiene usando un algoritmo voraz basado en el valor específico de los objetos.

b. El valor de la mochila continua correspondiente.

c. El valor de una mochila que contiene todos los objetos aunque se pase del peso máximo permitido.

En un problema de optimización, si el dominio de las decisiones es un conjunto infinito,

Seleccione una:

a. podremos aplicar el esquema *vuelta atrás* siempre que se trate de un conjunto infinito numerable.

b. es probable que a través de *programación dinámica* se obtenga un algoritmo eficaz que lo solucione.

c. una estrategia *voraz* puede ser la única alternativa. ✓

Dado un problema de optimización cualquiera. ¿la estrategia de *vuelta atrás* garantiza la solución óptima?

Seleccione una:

a. Es condición necesaria que el dominio de las decisiones sea discreto o discretizable y que el número de decisiones a tomar esté acotado.

b. Sí. puesto que ese método analiza todas las posibilidades.

c. Sí. siempre que el dominio de las decisiones sea discreto o discretizable y además se empleen mecanismos de poda basados en la mejor solución hasta el momento.

Se desea encontrar el camino mas corto entre dos ciudades
Para ello se dispone de una tabla con la distancia entre los pares de ciudades en los que hay carreteras o un valor centinela (por ejemplo -1) si no hay por lo que para ir de la ciudad inicial a la final es posible que haya que pasar por varias ciudades También se conocen las coordenadas geográficas de cada ciudad y por tanto la distancia geográfica (en línea recta) entre cada par de ciudades Para limitar la búsqueda en un algoritmo de *vuelta atrás* se utiliza la solución de un algoritmo *voraz* basado en moverse en cada paso a la ciudad de entre las posibles según el mapa de carreteras que esté más cercana al destino en línea recta

¿Qué tipo de cota sería?

Seleccione una

a Sería una cota pesimista siempre que se tenga la certeza de que esa aproximación encuentra una solución factible ✓

b Ninguna de las otras dos opciones

c Sería una cota optimista siempre que se tenga la certeza de que esa aproximación encuentra una solución factible

Se desea encontrar el camino mas corto entre dos ciudades
Para ello se dispone de una tabla con la distancia entre los pares de ciudades en los que hay carreteras o un valor centinela (por ejemplo, -1) si no hay, por lo que para ir de la ciudad inicial a la final es posible que haya que pasar por varias ciudades. Como también se conocen las coordenadas geográficas de cada ciudad se quiere usar la distancia geográfica (en linea recta) entre cada par de ciudades para como cota para limitar la búsqueda en un algoritmo de *vuelta atrás*

¿Qué tipo de cota sería?

Seleccione una:

a No se trataría de ninguna poda puesto que es posible que esa heurística no encuentre una solución factible

b Una cota pesimista

c Una cota optimista ✓

Se desea encontrar el camino mas corto entre dos ciudades.
Para ello se dispone de una tabla con la distancia entre los pares de ciudades en los que hay carreteras o un valor centinela (por ejemplo, -1) si no hay, por lo que para ir de la ciudad inicial a la final es posible que haya que pasar por varias ciudades. También se conocen las coordenadas geográficas de cada ciudad y por tanto la distancia geográfica (en línea recta) entre cada par de ciudades. Se pretende acelerar la búsqueda de un algoritmo de *ramificación y poda* priorizando los nodos vivos (ciudades) que estén a menor distancia geográfica de la ciudad objetivo.

Seleccioneuna

a. El nuevo algoritmo solo será más rápido para algunas instancias del problema ✓

b. Esta estrategia no asegura que se obtenga el camino mas corto.

c. El nuevo algoritmo siempre sea más rápido

Se desea encontrar el camino mas corto entre dos ciudades
Para ello se dispone de una tabla con la distancia entre los pares de ciudades en los que hay carreteras o un valor centinela (por ejemplo -1) si no hay por lo que para ir de la ciudad inicial a la final es posible que haya que pasar por varias ciudades También se conocen las coordenadas geográficas de cada ciudad y por tanto la distancia geográfica (en linea recta) entre cada par de ciudades Para limitar la búsqueda en un algoritmo de *vuelta atrás* se utiliza la solución de un algoritmo *voraz* basado en moverse en cada paso a la ciudad de entre las posibles según el mapa de carreteras que esté más cercana al destino segun su distancia geográfica
Este algoritmo voraz ¿serviría como cota pesimista?

Seleccione una

a No ya que no asegura que se encuentre una solución factible ✓

b Sí puesto que la distancia geográfica asegura que otra solución mejor no es posible

c No ya que en algunos casos puede dar distancias menores que la óptima

El problema de cortar un tubo de longitud n en segmentos de longitud entera, de manera que el precio total de sus partes sea máximo de acuerdo con una lista de precios por longitudes .

Seleccione una:

a. no se puede resolver usando un algoritmo de *vuelta atrás*.

b. se puede resolver mediante un algoritmo de *vuelta atrás* pero existe una solución asintóticamente mucho más eficiente. ✓

c. se debe resolver mediante un algoritmo de *vuelta atrás*, dado que otros algoritmos no consideran todas las posibles maneras de cortar el tubo.

Al resolver el problema del viajante de comercio mediante *vuelta atrás* y asumiendo un grafo de n vértices totalmente conexo. ¿cuál de estas es una buena cota pesimista al iniciar la búsqueda?

Seleccione una

a Se resuelve el problema usando un algoritmo voraz que añade cada vez al camino el vértice más cercano al último añadido.

b Se ordenan las aristas restantes de menor a mayor distancia y se calcula la suma de las n aristas más cortas.

c Se multiplica n por la distancia de la arista más corta que nos queda por considerar

Al resolver el problema del viajante de comercio mediante *vuelta atrás*, ¿cuál de estas cotas optimistas se espera que pode mejor el árbol de búsqueda?

Seleccione una:

a. Se multiplica k por la distancia de la arista más corta que nos queda por considerar, donde k es el número de saltos que nos quedan por dar.

b. Se resuelve el resto del problema usando un algoritmo voraz que añade cada vez al camino el vértice más cercano al último añadido.

c. Se ordenan las aristas restantes de menor a mayor distancia y se calcula la suma de las k aristas más cortas, donde k es el número de saltos que nos quedan por dar ✓

Di cuál de estas tres soluciones a problemas de optimización no comporta, en el peor caso tener que considerar $O(n!)$ posibilidades.

Seleccione una:

a. La solución de *vuelta atrás* al problema del viajante de comercio (*travelling salesman problem*), o sea, el de encontrar un ciclo hamiltoniano de coste mínimo en un grafo conexo de n vértices donde cada arista tiene un coste asignado. ✗

b. La solución de *ramificación y poda* al problema de la asignación de n tareas a n trabajadores de forma que cada trabajador hace exactamente una tarea y cada tarea es asignada a un trabajador exactamente, de forma que la suma de los costes de las tareas es mínimo. ✗

c. La solución al problema de buscar un árbol que cubre todos los vértices de un grafo de n vértices de forma que el coste es mínimo (*minimum spanning tree*).

SEGUNDO PARCIAL

Pregunta 6
Correcta
Puntúa como 1,00
Marcar pregunta

¿Cuál de estos tres problemas de optimización no tiene o no se le conoce una solución voraz óptima?

Seleccione una:

a. El árbol de cobertura de coste mínimo de un grafo conexo

b. El problema de la mochila discreta o sin fraccionamiento ✓

c. El problema de la mochila continua o con fraccionamiento

Pregunta 8
Correcta
Puntúa como 1,00
Marcar pregunta

Los algoritmos de programación dinámica hacen uso

Seleccione una

a. de que la solución óptima se puede construir añadiendo a la solución el elemento óptimo de los elementos restantes uno a uno.

b. de que se puede ahorrar cálculos guardando resultados anteriores en un almacén ✓

c. de una estrategia trivial consistente en examinar todas las soluciones posibles

Pregunta 10
Correcta
Puntúa como 1,00
Marcar pregunta

De los problemas siguientes indicad cuál no se puede tratar eficientemente como los otros dos

Seleccione una

a. El problema del cambio, o sea, el de encontrar la manera de entregar una cantidad de dinero usando el mínimo de monedas posibles.

b. El problema de la mochila sin fraccionamiento y sin restricciones en cuanto al dominio de los pesos de los objetos y de sus valores ✓

c. El problema de cortar un tubo de forma que se obtenga el máximo beneficio posible

Pregunta 1
Correcta
Puntúa como 1,00
Marcar pregunta

Un informático quiere subir a una montaña y para ello decide que tras cada paso, el siguiente debe tomarlo en la dirección de máxima pendiente hacia arriba. Además, entenderá que ha alcanzado la cima cuando llegue a un punto en el que no haya ninguna dirección que sea cuesta arriba. ¿qué tipo de algoritmo está usando nuestro informático?

Seleccione una:

a. un algoritmo voraz. ✓

b. un algoritmo divide y vencerás.

c. un algoritmo de programación dinámica.

Pregunta 2
Correcta
Puntúa como 1,00
Marcar pregunta

La mejora que en general aporta la programación dinámica frente a la solución ingenua se consigue gracias al hecho de que ...

Seleccione una:

a. ... en la solución ingenua se resuelve pocas veces un número relativamente grande de subproblemas distintos.

b. ... en la solución ingenua se resuelve muchas veces un número relativamente pequeño de subproblemas distintos. ✓

c. El número de veces que se resuelven los subproblemas no tiene nada que ver con la eficiencia de los problemas resueltos mediante programación dinámica.

Pregunta 3

Correcta

Puntúa como 1,00

Marcar pregunta

En la solución al problema de la mochila continua ¿por qué es conveniente la ordenación previa de los objetos?

Seleccione una:

- a. Para reducir la complejidad temporal en la toma de cada decisión: de $O(n)$ a $O(1)$, donde n es el número de objetos a considerar. ✓
- b. Porque si no se hace no es posible garantizar que la toma de decisiones siga un criterio voraz.
- c. Para reducir la complejidad temporal en la toma de cada decisión: de $O(n^2)$ a $O(n \log n)$, donde n es el número de objetos a considerar.

Pregunta 4

Correcta

Puntúa como 1,00

Marcar pregunta

Dada la suma de la recurrencia

$$T(n) = \begin{cases} 1 & n = 0 \\ \sum_{k=0}^{n-1} T(k) & n > 0 \end{cases}$$

¿cuál de las siguientes afirmaciones es cierta?

Seleccione una:

- a. $T(n) \in \Theta(n^2)$
- b. $T(n) \in \Theta(n!)$
- c. $T(n) \in \Theta(2^n)$ ✓

Pregunta 5

Correcta

Puntúa como 1,00

Marcar pregunta

Cuando la descomposición recursiva de un problema da lugar a subproblemas de tamaño similar, ¿qué esquema promete ser más apropiado?

Seleccione una:

- a. El método voraz.
- b. Divide y vencerás, siempre que se garantice que los subproblemas no son del mismo tamaño.
- c. Programación dinámica. ✓

Pregunta 7

Correcta

Puntúa como 1,00

Marcar pregunta

En el método voraz ...

Seleccione una:

- a. ... siempre se encuentra solución pero puede que no sea la óptima.
- b. ... es habitual preparar los datos para disminuir el coste temporal de la función que determina cuál es la siguiente decisión a tomar. ✓
- c. ... el dominio de las decisiones sólo pueden ser conjuntos discretos o discretizables.

Pregunta 8

Correcta

Puntúa como 1,00

Marcar pregunta

¿ Cómo se vería afectada la solución voraz al problema de la asignación de tareas en el caso de que se incorporaran restricciones que contemplen que ciertas tareas no pueden ser adjudicadas a ciertos trabajadores ?

Seleccione una:

a. Ya no se garantizaría la solución óptima pero sí una factible.

b. Habría que replantearse el criterio de selección para comenzar por aquellos trabajadores con más restricciones en cuanto a las tareas que no pueden realizar para asegurar, al menos, una solución factible.

c. La solución factible ya no estaría garantizada, es decir, pudiera ser que el algoritmo no llegue a solución alguna. ✓

Pregunta 9

Correcta

Puntúa como 1,00

Marcar pregunta

Se pretende implementar mediante programación dinámica iterativa la función recursiva:

```
unsigned f( unsigned x, unsigned v[] ) {
  if (x==0)
    return 0;
  unsigned m = 0;
  for ( unsigned k = 0; k < x; k++ )
     m = max( m, v[k] + f( x-k, v ) );
  return m;
}
```

¿Cuál es la mejor estructura para el almacén?

Seleccione una:

a. `int A`

b. `int A[]` ✓

c. `int A[][]`

Pregunta 8

Correcta

Puntúa como 1,00

Marcar pregunta

Se pretende implementar mediante programación dinámica iterativa la función recursiva

```
unsigned f( unsigned y, unsigned x) .  // suponemos y >= x
    if (x==0    y==x) return 1;
    return f(y-1, x-1) + f(y-1, x);
}
```

¿Cuál es la mejor estructura para el almacén?

Seleccione una

a. `int A[]`

b. `int A`

c. `int A[][]` ✓

Pregunta 12
Correcta
Puntúa como 1,00
Marcar pregunta

¿Cuál de estas estrategias para calcular el n-ésimo elemento de la serie de Fibonacci $(f(n) = f(n-1) + f(n-2),\ f(1) = f(2) = 1)$ es más eficiente?

Seleccione una:

a. La estrategia voraz.

• b. Programación dinámica. ✓

c. Las dos estrategias citadas serían similares en cuanto a eficiencia.

Pregunta 2
Correcta
Puntúa como 1,00
Marcar pregunta

Supongamos que una solución recursiva a un problema de optimización muestra estas dos características: por un lado, se basa en obtener soluciones óptimas a problemas parciales más pequeños, y por otro, estos subproblemas se resuelven más de una vez durante el proceso recursivo. Este problema es candidato a tener una solución alternativa basada en

Seleccione una:

a un algoritmo voraz.

b un algoritmo de programación dinámica ✓

c un algoritmo del estilo de *divide y vencerás*

Pregunta 11
Correcta
Puntúa como 1,00
Marcar pregunta

De los problemas siguientes, indicad cuál no se puede tratar eficientemente como los otros dos.

Seleccione una:

a. El problema de cortar un tubo de forma que se obtenga el máximo beneficio posible.

b. El problema del cambio, o sea, el de encontrar la manera de entregar una cantidad de dinero usando el mínimo de monedas posibles.

• c. El problema de la mochila sin fraccionamiento y sin restricciones en cuanto al dominio de los pesos de los objetos y de sus valores. ✓

Pregunta 5
Correcta
Puntúa como 1,00
Marcar pregunta

La solución de programación dinámica iterativa del problema de la mochila discreta

Seleccione una:

a. calcula menos veces el valor de la mochila que la correspondiente solución de programación dinámica recursiva

b. tiene un coste temporal asintótico exponencial con respecto al número de objetos

c. tiene la restricción de que los valores tienen que ser enteros positivos ✓

Pregunta 6
Puntúa como 1,00
Marcar pregunta

Cuando se calculan los coeficientes binomiales usando la recursión $\binom{n}{r} = \binom{n-1}{r} + \binom{n-1}{r-1}$ con $\binom{n}{0} = \binom{n}{n} = 1$ ¿qué problema se da y cómo resolver?

Seleccione una:

El problema de encontrar el árbol de recubrimiento de coste mínimo para un grafo no dirigido, conexo y ponderado ...

Seleccione una:

a. sólo se puede resolver con una estrategia voraz si existe una arista para cualquier par de vértices del grafo.

b. ... no se puede resolver en general con una estrategia voraz.

• c. ... se puede resolver siempre con una estrategia voraz. ✓

Si ante un problema de decisión existe un criterio de selección voraz entonces ...

Seleccione una:

- a. ... al menos una solución factible está garantizada.
- b. Ninguna de las otras dos opciones es cierta. ✓
- c. ... la solución óptima está garantizada.

Se pretende implementar mediante programación dinámica iterativa la función recursiva:

```
unsigned f( unsigned y, unsigned x){  // suponemos y >= x
    if (x==0 || y==x) return 1;
    return f(y-1, x-1) + f(y-1, x);
}
```

¿Cuál es la mejor complejidad espacial que se puede conseguir?

Seleccione una:

- a. $O(1)$
- b. $O(y^2)$
- c. $O(y)$ ✓

El valor que se obtiene con el método voraz para el problema de la mochila discreta es ...

Seleccione una:

- a. ... una cota inferior para el valor óptimo que a veces puede ser igual a este. ✓
- b. ... una cota inferior para el valor óptimo, pero que nunca coincide con este.
- c. ... una cota superior para el valor óptimo.

¿Cuál de estas tres estrategias voraces obtiene un mejor valor para la mochila discreta?

Seleccione una:

a. Meter primero los elementos de menor peso.

b. Meter primero los elementos de mayor valor específico o valor por unidad de peso. ✓

c. Meter primero los elementos de mayor valor.

La eficiencia de los algoritmos voraces se basa en el hecho de que ...
Seleccione una:

a. ... antes de tomar una decisión se comprueba si satisface las retricciones del problema.

b. ... con antelación, las posibles decisiones se ordenan de mejor a peor.

c. ... las decisiones tomadas nunca se reconsideran. ✓

La mejora que en general aporta la programación dinámica frente a la solución ingenua se consigue gracias al hecho de que ...

Seleccione una:

a. ... en la solución ingenua se resuelve pocas veces un número relativamente grande de subproblemas distintos.

b. El número de veces que se resuelven los subproblemas no tiene nada que ver con la eficiencia de los problemas resueltos mediante programación dinámica.

c. ... en la solución ingenua se resuelve muchas veces un número relativamente pequeño de subproblemas distintos. ✓

Supongamos que una solución recursiva a un problema de optimización muestra estas dos características: por un lado, se basa en obtener soluciones óptimas a problemas parciales más pequeños, y por otro, estos subproblemas se resuelven más de una vez durante el proceso recursivo. Este problema es candidato a tener una solución alternativa basada en ...

Seleccione una:

a. ... un algoritmo de programación dinámica. ✓

Pregunta 4
Correcta
Puntúa como 1.00
Marcar pregunta

¿Cuál de estos tres problemas de optimización no tiene o no se le conoce una solución voraz óptima?

Seleccione una:

a. El problema de la mochila discreta o sin fraccionamiento. ✓

b. El árbol de cobertura de coste mínimo de un grafo conexo

c. El problema de la mochila continua o con fraccionamiento.

Pregunta 6

Correcta

Puntúa como 1,00

Marcar pregunta

Si ante un problema de decisión existe un criterio de selección voraz entonces

Seleccione una:

a. ... la solución óptima está garantizada

b. Ninguna de las otras dos opciones es cierta ✓

c. ... al menos una solución factible está garantizada

Pregunta 10

Correcta

Puntúa como 1.00

Marcar pregunta

Supongamos que una solución recursiva a un problema de optimización muestra estas dos características: por un lado se basa en obtener soluciones óptimas a problemas parciales más pequeños subproblemas se resuelven más de una vez durante el proceso recursivo. Este problema es candidato a tener una solución alternativa basada en

Seleccione una

a un algoritmo del estilo de *divide y vencerás*

b un algoritmo de programación dinámica ✓

c un algoritmo voraz

Pregunta 3

Correcta

Puntúa como 1,00

Marcar pregunta

Supongamos que una solución recursiva a un problema de optimización muestra estas dos características: por un lado, se basa en obtener soluciones óptimas a problemas parciales más pequeños, y por otro, estos subproblemas se resuelven más de una vez durante el proceso recursivo. Este problema es candidato a tener una solución alternativa basada en ...

Seleccione una:

a. ... un algoritmo del estilo de *divide y vencerás*.

b. ... un algoritmo voraz.

c. ... un algoritmo de programación dinámica. ✓

Pregunta 7

Correcta

Puntúa como 1,00

Marcar pregunta

En la solución al problema de la mochila continua ¿por qué es conveniente la ordenación previa de los objetos?

Seleccione una:

a. Para reducir la complejidad temporal en la toma de cada decisión: de $O(n)$ a $O(1)$, donde n es el número de objetos a considerar. ✓

b. Porque si no se hace no es posible garantizar que la toma de decisiones siga un criterio voraz.

c. Para reducir la complejidad temporal en la toma de cada decisión: de $O(n^2)$ a $O(n \log n)$, donde n es el número de objetos a considerar.

Pregunta 3

Correcta

Puntúa como 1,00

Marcar pregunta

¿ Cómo se vería afectada la solución voraz al problema de la asignación de tareas en el caso de que se incorporaran restricciones que contemplen que ciertas tareas no pueden ser adjudicadas a ciertos trabajadores ?

Seleccione una:

a. Ya no se garantizaría la solución óptima pero sí una factible.

b. La solución factible ya no estaría garantizada, es decir, pudiera ser que el algoritmo no llegue a solución alguna. ✓

c. Habría que replantearse el criterio de selección para comenzar por aquellos trabajadores con más restricciones en cuanto a las tareas que no pueden realizar para asegurar, al menos, una solución factible.

Pregunta 7

Correcta

Puntúa como 1,00

Marcar pregunta

Cuando la descomposición recursiva de un problema da lugar a subproblemas de tamaño similar, ¿qué esquema promete ser más apropiado?

Seleccione una:

a. Programación dinámica. ✓

b. El método voraz.

c. Divide y vencerás, siempre que se garantice que los subproblemas no son del mismo tamaño.

Pregunta 2

Correcta

Puntúa como 1,00

Marcar pregunta

Un tubo de n centímetros de largo se puede cortar en segmentos de 1 centímetro, 2 centímetros, etc. Existe una lista de los precios a los que se venden los segmentos de cada longitud. Una de las maneras de cortar el tubo es la que más ingresos nos producirá. Di cuál de estas tres afirmaciones es falsa.

Seleccione una:

a. Hacer una evaluación exhaustiva "de fuerza bruta" de todas las posibles maneras de cortar el tubo consume un tiempo $\Theta(2^n)$

b. Es posible evitar hacer la evaluación exhaustiva "de fuerza bruta" guardando, para cada posible longitud $j<n$ el precio más elevado posible que se puede obtener dividiendo el tubo correspondiente.

c. Hacer una evaluación exhaustiva "de fuerza bruta" de todas las posibles maneras de cortar el tubo consume un tiempo $\Theta(n!)$ ✓

Pregunta 3

Correcta

Puntúa como 1,00

Marcar pregunta

En la solución al problema de la mochila continua ¿por qué es conveniente la ordenación previa de los objetos?

Seleccione una:

a. Porque si no se hace no es posible garantizar que la toma de decisiones siga un criterio voraz.

b. Para reducir la complejidad temporal en la toma de cada decisión: de $O(n)$ a $O(1)$ donde n es el número de objetos a considerar ✓

c. Para reducir la complejidad temporal en la toma de cada decisión: de $O(n^2)$ a $O(n \log n)$ donde n es el número de objetos a considerar.

Pregunta 10

Correcta

Puntúa como 1,00

Marcar pregunta

Si ante un problema de decisión existe un criterio de selección voraz entonces ...

Seleccione una:

a. Ninguna de las otras dos opciones es cierta. ✓

b. ... al menos una solución factible está garantizada.

c. ... la solución óptima está garantizada.

Pregunta **10**

Correcta

Puntúa como 1,00

Marcar pregunta

¿ Cómo se vería afectada la solución voraz al problema de la asignación de tareas en el caso de que se incorporaran restricciones que contemplen que ciertas tareas no pueden ser adjudicadas a ciertos trabajadores ?

Seleccione una

a. Habría que replantearse el criterio de selección para comenzar por aquellos trabajadores con más restricciones en cuanto a las tareas que no pueden realizar para asegurar, al menos, una solución factible

b. La solución factible ya no estaría garantizada, es decir, pudiera ser que el algoritmo no llegue a solución alguna ✓

c. Ya no se garantizaría la solución óptima pero sí una factible

Pregunta **11**

Correcta

Puntúa como 1,00

Marcar pregunta

Se pretende implementar mediante programación dinámica iterativa la función recursiva:

```
int f( int x, int y )
  if( x <= y ) return 1;
  return x + f(x-1,y);
```

¿Cuál es la mejor complejidad espacial que se puede conseguir?

Seleccione una

a. $O(x^2)$

b. $O(1)$ ✓

c. $O(x)$

Se pretende implementar mediante programación dinámica iterativa la función recursiva:

```
float f(unsigned x, int y)
  if( y < 0 ) return 0;
  float A = 0.0;
  if ( v1[y] <= x )
       A = v2[y] + f( x-v1[y], y-1 );
  float B = f( x, y-1 );
  return min(A,2+B);
```

¿Cuál es la mejor complejidad espacial que se puede conseguir?

Seleccione una:

a. $O(y^2)$

b. $O(1)$

c. $O(y)$ ✓

Pregunta **12**

Correcta

Puntúa como 1,00

Marcar pregunta

La solución de programación dinámica iterativa del problema de la mochila discreta

Seleccione una

a. .. calcula menos veces el valor de la mochila que la correspondiente solución de programación dinámica recursiva.

b. ... tiene un coste temporal asintótico exponencial con respecto al número de objetos.

c. . tiene la restricción de que los valores tienen que ser enteros positivos. ✓

Pregunta **5**

Correcta

Puntúa como 1,00

Marcar pregunta

La programación dinámica...

Seleccione una:

a. ... en algunos casos se puede utilizar para resolver problemas de optimización con dominios continuos pero probablemente pierda su eficacia ya que puede disminuir drásticamente el número de subproblemas repetidos.

b. Las otras dos opciones son ciertas. ✓

c. .. normalmente se usa para resolver problemas de optimización con dominios discretizables puesto que las tablas se han de indexar con este tipo de valores.

Pregunta **9**

Correcta

Puntúa como 1,00

Marcar pregunta

¿Cuál de los siguientes pares de problemas son equivalentes en cuanto al tipo de solución (óptima, factible, etc.) aportada por el método voraz?

Seleccione una

a. El fontanero diligente y la mochila continua ✓

b. El fontanero diligente y la asignación de tareas

c. El fontanero diligente y el problema del cambio.

Pregunta 1

Correcta

Puntúa como 1,00

Marcar pregunta

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

Seleccione una:

a. Las demás opciones son verdaderas.

b. ... nunca tendrá una complejidad exponencial.

c. ... será más eficiente cuanto más equitativa sea la división en subproblemas. ✓

Pregunta 8

Correcta

Puntúa como 1,00

Marcar pregunta

Indicad cuál de estas tres expresiones es cierta:

Seleccione una:

a. $O(n^2) \subset O(2^{\log(n)}) \subset O(2^n)$

b. $O(2^{\log(n)}) \subset O(n^2) \subset O(2^n)$ ✓

c. $O(n^2) \subset O(2^{\log(n)}) \subseteq O(2^n)$

Pregunta 3

Correcta

Puntúa como 1,00

Marcar pregunta

Dado un problema de optimización, el método voraz ...

Seleccione una:

a. ... siempre obtiene una solución factible

b. ... siempre obtiene la solución óptima.

c. ... garantiza la solución óptima sólo para determinados problemas ✓

Pregunta 5

Correcta

Puntúa como 1,00

Marcar pregunta

Un informático quiere subir a una montaña y para ello decide que tras cada paso, el siguiente debe tomarlo en la dirección de máxima pendiente hacia arriba. Además, entenderá que ha alcanzado la cima cuando llegue a un punto en el que no haya ninguna dirección que sea cuesta arriba. ¿qué tipo de algoritmo está usando nuestro informático?

Seleccione una:

- a. un algoritmo voraz. ✓
- b. un algoritmo de programación dinámica.
- c. un algoritmo divide y vencerás.

Pregunta 8

Correcta

Puntúa como 1,00

Marcar pregunta

Se pretende implementar mediante programación dinámica iterativa la función recursiva:

```
int f( int x, int y ) {
  if( x <= y ) return 1;
  return x + f(x-1,y);
}
```

¿Cuál es la mejor complejidad espacial que se puede conseguir?

Seleccione una:

- a. $O(1)$ ✓
- b. $O(x^2)$
- c. $O(x)$

Pregunta 12

Correcta

Puntúa como 1,00

Marcar pregunta

Un tubo de n centímetros de largo se puede cortar en segmentos de 1 centímetro, 2 centímetros, etc. Existe una lista de los precios a los que se venden los segmentos de cada longitud. Una de las maneras de cortar el tubo es la que más ingresos nos producirá. Di cuál de estas tres afirmaciones es falsa.

Seleccione una:

a. Es posible evitar hacer la evaluación exhaustiva "de fuerza bruta" guardando, para cada posible longitud $j < n$ el precio más elevado posible que se puede obtener dividiendo el tubo correspondiente.

b. Hacer una evaluación exhaustiva "de fuerza bruta" de todas las posibles maneras de cortar el tubo consume un tiempo $\Theta(n!)$ ✓

c. Hacer una evaluación exhaustiva "de fuerza bruta" de todas las posibles maneras de cortar el tubo consume un tiempo $\Theta(2^n)$

Pregunta 2

Correcta

Puntúa como 1,00

Marcar pregunta

La mejora que en general aporta la programación dinámica frente a la solución ingenua se consigue gracias al hecho de que . .

Seleccione una:

a. . en la solución ingenua se resuelve muchas veces un número relativamente pequeño de subproblemas distintos. ✓

b. . en la solución ingenua se resuelve pocas veces un número relativamente grande de subproblemas distintos.

c. El número de veces que se resuelven los subproblemas no tiene nada que ver con la eficiencia de los problemas resueltos mediante programación dinámica.

Pregunta 9

Correcta

Puntúa como 1,00

Marcar pregunta

¿Cuál de estas tres estrategias voraces obtiene un mejor valor para la mochila discreta?

Seleccione una:

a. Meter primero los elementos de menor peso.

b. Meter primero los elementos de mayor valor específico o valor por unidad de peso. ✓

c. Meter primero los elementos de mayor valor.

Pregunta **2**

Correcta

Puntúa como 1,00

Marcar pregunta

En la solución al problema de la mochila continua ¿por qué es conveniente la ordenación previa de los objetos?

Seleccione una

a. Porque si no se hace no es posible garantizar que la toma de decisiones siga un criterio voraz.

b. Para reducir la complejidad temporal en la toma de cada decisión de $O(n)$ a $O(1)$ donde n es el número de objetos a considerar ✓

c. Para reducir la complejidad temporal en la toma de cada decisión: de $O(n^2)$ a $O(n \log n)$ donde n es el número de objetos a considerar

Pregunta **4**

Correcta

Puntúa como 1,00

Marcar pregunta

En el método voraz .

Seleccione una

a. siempre se encuentra solución pero puede que no sea la óptima.

b. es habitual preparar los datos para disminuir el coste temporal de la función que determina cuál es la siguiente decisión a tomar ✓

c. el dominio de las decisiones sólo pueden ser conjuntos discretos o discretizables.