Comenzado el	lunes, 21 de julio de 2025, 21:57
Estado	Finalizado
Finalizado en	lunes, 21 de julio de 2025, 21:57
Tiempo empleado	6 segundos
Calificación	**0,00** de 10,00 (**0**%)

Pregunta 1

Sin contestar

Se puntúa como 0 sobre 1,00

Dada la siguiente recurrencia, indica la respuesta correcta:

$T(n) = 3n^2$ si $n \leq 1$

$T(n) = 3T(n/2) + n^2$ si $n > 1$

Seleccione una:

a. $T(n) \in \Theta(n^2 \mathbf{log}\ n)$

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

c. $T(n) \in \Theta(\mathbf{log}\ n)$

d. Ninguna de las anteriores.

a. Falso. Aplicando el teorema de la división con a=3, b=2 y k=2, como $3 < 2^2$ tenemos que $T(n) \in \Theta(n^2) \neq \Theta(n^2 \mathbf{log}\ n)$.
b. Cierto. Aplicando el teorema de la división con a=3, b=2 y k=2, como $3 < 2^2$ tenemos que $T(n) \in \Theta(n^2) \neq \Theta(n)$.
c. Falso. Aplicando el teorema de la división con a=3, b=2 y k=2, como $3 < 2^2$ tenemos que $T(n) \in \Theta(n^2) \neq \Theta(\mathbf{log}\ n)$.
d. Falso. La respuesta correcta es: $T(n) \in \Theta(n^2)$

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

Pregunta 2

Sin contestar

Se puntúa como 0 sobre 1,00

¿Qué tipo de recursión es la de esta función?

```
int mcd(int a, int b)
{
 int m;
 if (a==b) { m = a;}
 else if (a>b) { m = mcd(a-b,b);}
 else {m=mcd(a,b-a);}
 return m;
}
```

Seleccione una:

- a. Recursión lineal final.
- b. Recursión múltiple.
- c. Ninguna de las anteriores.

a. Cierto. Se trata de recursión lineal porque en cada alternativa hay una única llamada recursiva y es final porque el resultado de la llamada recursiva en cada alternativa es el resultado de la función.
b. Falso. Se trata de recursión lineal porque en cada alternativa hay una única llamada recursiva y por tanto cada invocación a la función genera una sola llamada recursiva.
c. Falso. La respuesta correcta es: Recursión lineal final.

La respuesta correcta es: Recursión lineal final.

Pregunta 3

Sin contestar

Se puntúa como 0 sobre 1,00

En los algoritmos recursivos en los que se reduce el tamaño del problema por sustracción:

$T(n) = c_0$ si $0 \leq n \leq n_0$

$T(n) = aT(n-b) + cn^k$ si $n > n_0$

Seleccione una:

- a. Si $a = 1$, el coste es lineal.
- b. El coste siempre es exponencial.
- c. Para que sea polinómico es necesario que b>a.
- d. Ninguna de las anteriores.

a. Falso. En caso de que $a = 1$ el coste está en $\Theta\left(n^{k+1}\right)$. Por tanto si $k > 0$ no es lineal.
b. Falso. En caso de que $a = 1$ el coste está en $\Theta\left(n^{k+1}\right)$, y es por tanto polinómico.
c. Falso. Para que sea polinómico es necesario que a=1, independientemente del valor de b.
d. Cierto. La respuesta correcta es: Si a>1, es siempre exponencial por muy grande que sea b.

La respuesta correcta es: Ninguna de las anteriores.

Pregunta 4

Sin contestar

Se puntúa como 0 sobre 1,00

¿En qué casos no termina este algoritmo recursivo ?

```
int f(int x)
{
  int resultado;
  if (x==0)
    {resultado=0;}
  else
    {resultado = f(x-2);}
  return resultado;
}
```

Seleccione una:

- a. Solamente en los números negativos y en los positivos impares.
- b. Siempre termina, pues el argumento de la llamada decrece.
- c. Solo en los números negativos.
- d. Ninguna de las anteriores.

a. Cierto. No termina en los números positivos impares, ya que al restar dos en cada llamada recursiva no se alcanza el caso base 0. Tampoco en los enteros negativos.
b. Falso. No siempre termina, por ejemplo en los números negativos al restar dos en cada llamada recursiva no se alcanza el caso base 0.
c. Falso. Tampoco termina en los números positivos impares, ya que al restar dos en cada llamada recursiva no se alcanza el caso base 0.
d. Falso. La respuesta correcta es: Solamente en los números negativos y en los positivos impares.

La respuesta correcta es: Solamente en los números negativos y en los positivos impares.

Pregunta 5

Sin contestar

Se puntúa como 0 sobre 1,00

Indica cuál es la afirmación correcta acerca de este algoritmo de búsqueda binaria en el que la precondición es $0 \leq c \leq f+1 \leq longitud\,(v)$:

```
int f(int v[], int x, int c, int f)
{
  int resultado;
  if (c>f)
   {resultado = c-1;}
  else
   {
    int m = (c+f)/2;
    if (v[m] > x)
     {resultado = f(v,x,c,m);}
    else
     {resultado = f(v,x,m,f);}
   }
  return resultado;
}
```

Seleccione una:

- a. Solo termina si $c > f$.
- b. Solo termina si $c < f$.
- c. Siempre termina.
- d. Ninguna de las anteriores.

a. Cierto. El algoritmo no termina en todos aquellos casos en los que $c \leq f$. El punto medio cumple $c \leq m \leq f$ y por tanto las llamadas recursivas (con c,m y m,f) de nuevo cumplen que $c \leq f$. Luego, si inicialmente $c \leq f$ se llegará a un caso base en el que $c = m = f$, que repetidamente se invocará a sí mismo, generando una recursión infinita.

b. Falso. El algoritmo no termina en todos aquellos casos en los que $c \leq f$. El punto medio cumple $c \leq m \leq f$ y por tanto las llamadas recursivas (con c,m y m,f) de nuevo cumplen que $c \leq f$. Luego, si inicialmente $c \leq f$ se llegará a un caso base en el que $c = m = f$, que repetidamente se invocará a sí mismo, generando una recursión infinita.

c. Falso. El algoritmo no termina en todos aquellos casos en los que $c \leq f$. El punto medio cumple $c \leq m \leq f$ y por tanto las llamadas recursivas (con c,m y m,f) de nuevo cumplen que $c \leq f$. Luego, si inicialmente $c \leq f$ se llegará a un caso base en el que $c = m = f$, que repetidamente se invocará a sí mismo, generando una recursión infinita.

d. Falso. La respuesta correcta es: Solo termina si $c > f$.

La respuesta correcta es: Solo termina si $c > f$.

Pregunta 6

Sin contestar

Se puntúa como 0 sobre 1,00

Indica cual de las siguientes recurrencias representa el coste del algoritmo mergesort:

Seleccione una:

- a. $T(n) = c$ si $n \leq 1, T(n) = 2T(n/2) + c'$ si $n > 1$
- b. $T(n) = c$ si $n \leq 1, T(n) = 2T(n-1) + c'n$ si $n > 1$
- c. $T(n) = c$ si $n \leq 1, T(n) = 2T(n/2) + c'n$ si $n > 1$
- d. Ninguna de las anteriores.

a. Falso. El coste adicional a las llamadas recursivas, que se debe fundamentalmente al algoritmo de mezcla, es lineal en el tamaño del problema n=f-c+1, no constante. Debería ser $c'n$, no c'.
b. Falso. El tamaño de las llamadas recursivas es la mitad del tamaño de la llamada original, debería ser $2T(n/2)$, no $2T(n-1)$.
c. Cierto. En cada llamada a la función se general dos llamadas recursivas de la mitad de tamaño y se invierte un coste lineal en el tamaño del segmento para realizar la mezcla.
d. Falso. La respuesta correcta es: $T(n) = c$ si $n \leq 1, T(n) = 2T(n/2) + c'n$ si $n > 1$

La respuesta correcta es: $T(n) = c$ si $n \leq 1, T(n) = 2T(n/2) + c'n$ si $n > 1$

Pregunta 7

Sin contestar

Se puntúa como 0 sobre 1,00

La generalización es una técnica que se utiliza para:

Seleccione una o más de una:

- a. Diseñar algoritmos recursivos.
- b. Mejorar la eficiencia de los algoritmos recursivos.
- c. Disponer de funciones más generales que la que se deseaba.
- d. Obtener versiones recursivas finales de funciones recursivas lineales no finales.

a. Correcta. Añadir parámetros adicionales permite definir recursivamente algunos algoritmos, por ejemplo la búsqueda binaria o los métodos de ordenación quicksort y mergesort en los que se añade como parámetros adicionales los extremos del segmento del vector en que actúa el algoritmo.
b. Correcta. Añadir parámetros y/o resultados adicionales puede ayudar a mejorar la eficiencia de los algoritmos recursivos.
c. Correcta. Añadir parámetros adicionales permite disponer de versiones más generales, por ejemplo en los métodos de ordenación quicksort y mergesort, el hecho de añadir como parámetros adicionales los extremos del segmento del vector en que actúa el algoritmo nos permite ordenar un segmento cualquiera del vector y en particular todo el vector.
d. Correcta. Generar versiones recursivas finales a partir de versiones no finales requiere añadir un parámetro adicional acumulador.

Las respuestas correctas son: Diseñar algoritmos recursivos., Mejorar la eficiencia de los algoritmos recursivos., Disponer de funciones más generales que la que se deseaba., Obtener versiones recursivas finales de funciones recursivas lineales no finales.

Pregunta 8

Sin contestar

Se puntúa como 0 sobre 1,00

Indica la menor función $f(n)$ que cumple que $T(n) \in O(f(n))$:

$T(1) = 1$

$T(n) = T(n-1) + n$ si $n > 1$

Seleccione una:

- a. 2^n
- b. n^2
- c. $\log n$
- d. Ninguna de las anteriores.

a. Falso. Aplicando el teorema de la resta con a=1, b=1 y k=1, tenemos que $T(n) \in \Theta(n^2) \neq \Theta(2^n)$.
b. Cierto. Aplicando el teorema de la resta con a=1, b=1 y k=1, tenemos que $T(n) \in \Theta(n^2)$.
c. Falso. Aplicando el teorema de la resta con a=1, b=1 y k=1, tenemos que $T(n) \in \Theta(n^2) \neq \Theta(\log n)$.
d. Falso. La respuesta correcta es: n^2

La respuesta correcta es: n^2

Pregunta 9

Sin contestar

Se puntúa como 0 sobre 1,00

Indica a qué orden de complejidad pertenece $T(n)$ definida por la recurrencia:

$T(n) = c_0$ si $n \leq 1$

$T(n) = T(n-1) + T(n-2) + c_1$ si $n > 1$

Seleccione una:

- a. $T(n) \in O(n \log n)$
- b. $T(n) \in O(n)$
- c. $T(n) \in O(2^n)$
- d. Ninguna de las anteriores.

a. Falso. Puesto que $T(n) = T(n-1) + T(n-2) + c_1 \leq 2T(n-1) + c_1$, aplicando el teorema de la división con a=2, b=1, k=0, tenemos que $T(n) \in O(2^n)$. Análogamente, $T(n) = T(n-1) + T(n-2) + c_1 \geq 2T(n-2) + c_1$ y aplicando el teorema de la división con a=2, b=2, k=0, tenemos que $T(n) \in \Omega(2^{n/2})$. Por tanto $T(n) \in O(n \log n)$.
b. Falso. Puesto que $T(n) = T(n-1) + T(n-2) + c_1 \leq 2T(n-1) + c_1$, aplicando el teorema de la división con a=2, b=1, k=0, tenemos que $T(n) \in O(2^n)$. Análogamente, $T(n) = T(n-1) + T(n-2) + c_1 \geq 2T(n-2) + c_1$ y aplicando el teorema de la división con a=2, b=2, k=0, tenemos que $T(n) \in \Omega(2^{n/2})$. Por tanto $T(n) \in O(n)$.
c. Cierto. Puesto que $T(n) = T(n-1) + T(n-2) + c_1 \leq 2T(n-1) + c_1$, aplicando el teorema de la división con a=2, b=1, k=0, tenemos que $T(n) \in O(2^n)$. Análogamente, $T(n) = T(n-1) + T(n-2) + c_1 \geq 2T(n-2) + c_1$ y aplicando el teorema de la división con a=2, b=2, k=0, tenemos que $T(n) \in \Omega(2^{n/2})$.
d. Falso. La respuesta correcta es: $T(n) \in O(2^n)$

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

Pregunta 10

Sin contestar

Se puntúa como 0 sobre 1,00

El caso mejor del algoritmo quicksort que usa el primer elemento del segmento como pivote y la fase de partición reparte los elementos en menores o iguales y mayores o iguales, se da cuando:

Seleccione una:

a. El primer elemento de cada segmento con el que se realiza una llamada es la mediana del mismo.

b. El primer elemento es la mediana del array.

c. El último elemento es la mediana del array.

d. Ninguna de las anteriores.

a. Cierto. El caso mejor se da cuando en cada llamada recursiva, el primer elemento que se usa como pivote queda colocado en la mitad del segmento después de la partición generando dos llamadas recursivas de la mitad de tamaño, y eso sucede cuando es la mediana del segmento. Esto sucede por ejemplo en este array: 4,1,3,2,6,5,7.
b. Falso. Si el primer elemento es la mediana del array se generarán dos llamadas recursivas de tamaño la mitad, pero las sucesivas llamadas podrían dejar el pivote en un extremo, dando lugar al caso peor con coste en $O\left(n^2\right)$, siendo n el número de elementos del array.
c. Falso. El caso mejor se da cuando en cada llamada recursiva, el primer elemento que se usa como pivote queda colocado en la mitad del segmento después de la partición generando dos llamadas recursivas de la mitad de tamaño, y eso sucede cuando es la mediana del segmento.
d. Falso. La respuesta correcta es: El primer elemento de cada segmento con el que se realiza una llamada es la mediana del mismo.

La respuesta correcta es: El primer elemento de cada segmento con el que se realiza una llamada es la mediana del mismo.