Cuestionario Global de Continua

Fecha de Creación	@22 de diciembre de 2023 13:47
Asignatura	FAL
Fecha de Modificación	@22 de diciembre de 2023 13:49

Pregunta 1
Parcialmente correcta
Se puntúa 0,67 sobre 1,00
Marcar pregunta

Sea $t(n) = 20\ n\ log_2\ n + 5$ la función de tiempo de un algoritmo A. El coste de A está en:

Seleccione una o más de una:

- a. $\Omega(log\ n)$
- b. $O(\log\ n)$
- c. $\Omega(1)$ ✔ **Cierto**. Esta función está en $\Theta(n\ log\ n)$. Por tanto, 1 crece más despacio.
- d. $\Theta(log\ n)$
- e. $\Theta(n\ log\ n)$ ✔ **Cierto**. El término dominante en esta función es $20\ n\ log_2\ n$.

a. **Cierto**. Esta función está en $\Theta(n\ log\ n)$. Por tanto, $log\ n$ crece más despacio.
b. **Falso**. Esta función está en $\Theta(n\ log\ n)$. Por tanto, $log\ n$ crece más despacio, no más deprisa.
c. **Cierto**. Esta función está en $\Theta(n\ log\ n)$. Por tanto, 1 crece más despacio.
d. **Falso**. Esta función está en $\Theta(n\ log\ n)$. Por tanto, $log\ n$ crece más despacio.
e. **Cierto**. El término dominante en esta función es $20\ n\ log_2\ n$.

Las respuestas correctas son: $\Omega(log\ n)$
, $\Omega(1)$
, $\Theta(n\ log\ n)$

Pregunta 2
Parcialmente correcta
Se puntúa 0,67 sobre 1,00
Desmarcar

Considérese el siguiente algoritmo:

```
int algo(int a[], int n) {
    int resul = 0;
    int i = 0;
    int j = n-1;
    while (j >= i) {
        if (a[i] > 0) {
            resul++;
        }
        if (j > i && a[j] > 0) {
            resul++;
        }
        i++;
        j--;
    }
    return resul;
}
```

¿Cuáles de las siguientes afirmaciones son ciertas?

Seleccione una o más de una:

- a. $resul = \#\ u : 0 \leq u < i \vee j < u < n : a[u] > 0$ es un invariante del bucle.
- b. La postcondición puede especificarse como $resul = \#\ i : 0 \leq i < n : a[i] > 0$. ✔ **Cierto**. El algoritmo calcula el número de valores positivos del vector, que es, precisamente, lo que especifica el predicado.
- c. $j - i$ es una expresión de cota del bucle. ✔ **Cierto**. $j - i$ decrece en cada iteración, y nunca baja de -1.
- d. $resul = \#\ u : j < u < n : a[u] > 0$ es un invariante del bucle.
- e. j es una expresión de cota del bucle.

a. **Cierto**. El vector se procesa en *abanico*: en $resul$ estará el número de positivos en los tramos a[0..i) y a(j..n), que es, precisamente, lo que dice este predicado.
b. **Cierto**. El algoritmo calcula el número de valores positivos del vector, que es, precisamente, lo que especifica el predicado.
c. **Cierto**. $j - i$ decrece en cada iteración, y nunca baja de -1.
d. **Falso**. El vector se procesa en *abanico*: en $resul$ estará el número de positivos en los tramos a[0..i) y a(j..n), y lo que dice este predicado es que en $resul$ está el número de positivos en a(j..n).
e. **Falso**. Aunque j disminuye en cada iteración, no puede garantizarse que tenga un mínimo.

Las respuestas correctas son: $resul = \#\ u : 0 \leq u < i \vee j < u < n : a[u] > 0$ es un invariante del bucle.
, La postcondición puede especificarse como $resul = \#\ i : 0 \leq i < n : a[i] > 0$.
, $j - i$ es una expresión de cota del bucle.

Pregunta 3

Parcialmente correcta

Se puntúa 0,17 sobre 1,00

Marcar pregunta

Considérese la siguiente función recursiva:

```
int algo(int a) {
    if (a < 2) return a;
    else return algo(a - 1) + algo(a - 2);
}
```

¿Cuáles de las siguientes afirmaciones son ciertas?

Seleccione una o más de una:

- [x] a. Una de las llamadas es no final, y la otra es final. ✖ **Falso**. Ambas llamadas son no finales.
- [x] b. Tiene recursión múltiple. ✔ **Cierto**. El caso recursivo involucra dos llamadas recursivas.
- [ ] c. Todas las llamadas recursivas son no finales.
- [ ] d. Todas las llamadas recursivas son finales.
- [ ] e. Tiene recursión simple.

a. **Falso**. Ambas llamadas son no finales.
b. **Cierto**. El caso recursivo involucra dos llamadas recursivas.
c. **Cierto**. Tras la primera llamada, queda pendiente realizar la segunda llamada, y la suma. Tras la segunda llamada, queda pendiente realizar la suma.
d. **Falso**. Ninguna de las llamadas son finales.
e. **Falso**. El caso recursivo involucra dos llamadas recursivas.

Las respuestas correctas son: Tiene recursión múltiple., Todas las llamadas recursivas son no finales.

Pregunta 4

Parcialmente correcta

Se puntúa 0,17 sobre 1,00

Marcar pregunta

En relación con los algoritmos de *vuelta atrás* ¿Cuáles de las siguientes afirmaciones son ciertas?

Seleccione una o más de una:

- [x] a. Si se utiliza un algoritmo de *vuelta atrás* para resolver un problema de optimización, el algoritmo debe explorar todas las soluciones finales para determinar cuál es la mejor. ✖ **Falso**. Pueden utilizarse podas para eliminar alternativas que no van a mejorar la mejor solución encontrada hasta el momento.
- [ ] b. Un algoritmo de *vuelta atrás* que enumera todas las soluciones finales no puede utilizar podas.
- [ ] c. Un algoritmo de *vuelta atrás* que se detiene cuando encuentra la primera solución también puede beneficiarse del uso de marcadores.
- [x] d. Las podas son útiles en todos los algoritmos de *vuelta atrás*, aunque estos no resuelvan problemas de optimización. ✔ **Cierto**. Las podas permiten reducir el espacio de búsqueda, independientemente del tipo de problema a resolver. Lo que hay que garantizar es que no descartan soluciones que deban tenerse en cuenta.
- [ ] e. Las podas y los marcadores son incompatibles.

a. **Falso**. Pueden utilizarse podas para eliminar alternativas que no van a mejorar la mejor solución encontrada hasta el momento.
b. **Falso**. Pueden podarse alternativas para las cuáles es posible determinar que no van a conducir a nuevas soluciones.
c. **Cierto**. Los marcadores permiten reconocer soluciones viables y finales de forma más eficiente.
d. **Cierto**. Las podas permiten reducir el espacio de búsqueda, independientemente del tipo de problema a resolver. Lo que hay que garantizar es que no descartan soluciones que deban tenerse en cuenta.
e. **Falso**. Ambas técnicas contribuyen a mejorar la eficiencia de los algoritmos de *vuelta atrás*, y, por tanto, son complementarias.

Las respuestas correctas son: Un algoritmo de *vuelta atrás* que se detiene cuando encuentra la primera solución también puede beneficiarse del uso de marcadores., Las podas son útiles en todos los algoritmos de *vuelta atrás*, aunque estos no resuelvan problemas de optimización.

Pregunta 5

Parcialmente correcta

Se puntúa 0,50 sobre 1,00

Marcar pregunta

A es un algoritmo *divide y vencerás* consistente en un caso base que aplica un algoritmo B, y un caso recursivo que descompone el problema a resolver P en dos subproblemas S_1 y S_2, y aplica un proceso de combinación C a las soluciones de los subproblemas para obtener la solución final. ¿Cuáles de las siguientes afirmaciones son ciertas?:

Seleccione una o más de una:

- [ ] a. Los tamaños de S_1 y S_2 deben ser aproximadamente los mismos.
- [x] b. Los tamaños de S_1 y S_2 deben ser una fracción del tamaño de P. ✔ **Cierto**. Es una característica distintiva de este tipo de algoritmos.
- [ ] c. La complejidad del algoritmo B que se utiliza para resolver el caso base debe ser constante.
- [ ] d. La complejidad de la combinación C debe ser necesariamente constante.
- [ ] e. Si n es el tamaño de P, entonces los tamaños de S_1 y S_2 pueden ser $n-1$.

a. **Cierto**. El problema se descompone en subproblemas cuyo tamaño es, aproximadamente, una fracción del tamaño del problema original.
b. **Cierto**. Es una característica distintiva de este tipo de algoritmos.
c. **Falso**. Considérese, por ejemplo, la versión del *mergesort* que invoca un algoritmo de complejidad cuadrática para resolver problemas de tamaños suficientemente pequeños.
d. **Falso**. Considérese, por ejemplo, el proceso de mezcla utilizado en el *mergesort*.
e. **Falso**. El problema se descompone en subproblemas cuyo tamaño es, aproximadamente, una fracción del tamaño del problema original.

Las respuestas correctas son: Los tamaños de S_1 y S_2 deben ser aproximadamente los mismos.
, Los tamaños de S_1 y S_2 deben ser una fracción del tamaño de P.

Pregunta 6

Correcta

Se puntúa 1,00 sobre 1,00

Marcar pregunta

Sea $P(x)$ un predicado. ¿Cuáles de las siguientes afirmaciones son ciertas?:

Seleccione una o más de una:

- a. El predicado $\exists\ x : false : P(x)$ es siempre *falso*. ✔ **Cierto**. Como el rango de variación de x está vacío, no existe ningún x que satisfaga $P(x)$.
- b. La expresión $\Sigma\ x : false : P(x)$ no está definida.
- c. El valor de la expresión $min\ x : false : P(x)$ depende de $P(x)$.
- d. El predicado $(\Sigma\ x : false : P(x)) = (\#\ x : false : P(x))$ es siempre *cierto*. ✔ **Cierto**. Las expresiones a ambos lados de la igualdad valen ambas 0.
- e. La expresión $\prod\ x : false : P(x)$ es igual a 1. ✔ **Cierto**. El producto de los valores de un conjunto vacío es, por definición, 1.

a. **Cierto**. Como el rango de variación de x está vacío, no existe ningún x que satisfaga $P(x)$.
b. **Falso**. Esta expresión es igual a 0.
c. **Falso**. Esta expresión está indefinida, independientemente de cómo sea $P(x)$.
d. **Cierto**. Las expresiones a ambos lados de la igualdad valen ambas 0.
e. **Cierto**. El producto de los valores de un conjunto vacío es, por definición, 1.

Las respuestas correctas son: El predicado $\exists\ x : false : P(x)$ es siempre *falso*.
, El predicado $(\Sigma\ x : false : P(x)) = (\#\ x : false : P(x))$ es siempre *cierto*.
, La expresión $\prod\ x : false : P(x)$ es igual a 1.

Pregunta 7

Parcialmente correcta

Se puntúa 0,67 sobre 1,00

Marcar pregunta

Supón que estás demostrando la *corrección total* de un algoritmo iterativo que tiene la siguiente estructura:

```
<código de inicialización>
while(<condición del bucle>) {
  <cuerpo del bucle>
}
```

¿Qué debes comprobar?

Seleccione una o más de una:

- a. Que el invariante se satisface en cualquier estado al que se llega inmediatamente antes de comenzar a ejecutar el bucle.
- b. Que el invariante se satisface en cualquier estado que satisface a la precondición.
- c. Que, en cualquier estado en el que se satisface el invariante y la condición del bucle, se satisface también la postcondición.
- d. Que, si se supone cierto el invariante, y falsa la condición del bucle, entonces puede derivarse la postcondición. ✔ **Cierto**. En este momento se sale del bucle, por lo que será necesario demostrar que se cumple la postcondición.
- e. Que, si comienza en cualquier estado que satisface la precondición, entonces el algoritmo termina. ✔ **Cierto**. Porque se está demostrando la corrección total. Entonces, si, al comienzo de la ejecución, se satisface la precondición, el algoritmo debe necesariamente terminar en un estado que satisface la postcondición.

a. **Cierto**. Porque el invariante debe satisfacerse inmediatamente antes de ejecutar el bucle.
b. **Falso**. Debe ejecutarse el código de inicialización.
c. **Falso**. Debe haberse falsificado la condición, para poder salir del bucle.
d. **Cierto**. En este momento se sale del bucle, por lo que será necesario demostrar que se cumple la postcondición.
e. **Cierto**. Porque se está demostrando la corrección total. Entonces, si, al comienzo de la ejecución, se satisface la precondición, el algoritmo debe necesariamente terminar en un estado que satisface la postcondición.

Las respuestas correctas son: Que el invariante se satisface en cualquier estado al que se llega inmediatamente antes de comenzar a ejecutar el bucle., Que, si se supone cierto el invariante, y falsa la condición del bucle, entonces puede derivarse la postcondición., Que, si comienza en cualquier estado que satisface la precondición, entonces el algoritmo termina.

Pregunta 8

Correcta

Se puntúa 1,00 sobre 1,00

Marcar pregunta

La función de tiempo $t(n)$ de un algoritmo A satisface la siguiente recurrencia:

$t(0) = 1$

$t(n) = 3n + 5 + 16t(n/4), n > 0$

Esta función:

Seleccione una o más de una:

- a. Está en $O(n^2)$. ✔ **Cierto**. La recurrencia se ajusta al patrón de división, con $a = 16$, $k = 1$, y $b = 4$. Por tanto, $a = 16 > b^k = 4^1 = 4$. Como resultado, $T(n) \in O(n^{log_4 16}) = O(n^2)$.
- b. Está en $O(n)$.
- c. Está en $O(\sqrt{n})$.
- d. Está en $O(\sqrt{n\ log\ n})$.
- e. Está en $O(n\ log\ n)$.

a. **Cierto**. La recurrencia se ajusta al patrón de división, con $a = 16$, $k = 1$, y $b = 4$. Por tanto, $a = 16 > b^k = 4^1 = 4$. Como resultado, $T(n) \in O(n^{log_4 16}) = O(n^2)$.
b. **Falso**. Se obtiene el orden de complejidad $O(n^2)$.
c. **Falso**. Se obtiene el orden de complejidad $O(n^2)$.
d. **Falso**. Se obtiene el orden de complejidad $O(n^2)$.
e. **Falso**. Se obtiene el orden de complejidad $O(n^2)$.

La respuesta correcta es: Está en $O(n^2)$.

Pregunta 9

Correcta

Se puntúa 1,00 sobre 1,00

Marcar pregunta

Supón que tanto $f(n)$ como $g(n)$ son funciones de tiempo de algoritmos. Sabiendo que $f(n) \in O(g(n))$, ¿cuáles de las siguientes afirmaciones son verdaderas?:

Seleccione una o más de una:

- [x] a. $lim_{n\to\infty} \frac{f(n)}{g(n)}$ puede ser igual a 0. ✔ **Cierto**. Considérese, por ejemplo, el caso en el que $f(n) = 1$ y $g(n) = n$.
- [ ] b. $lim_{n\to\infty} \frac{f(n)}{g(n)}$ puede ser ∞.
- [ ] c. $lim_{n\to\infty} \frac{f(n)}{g(n)}$ puede ser 0, ∞ o $c > 0$, dependiendo de cómo sean $f(n)$ y $g(n)$.
- [ ] d. $lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ siempre.
- [x] e. $lim_{n\to\infty} \frac{f(n)}{g(n)}$ puede ser igual a un valor finito $c > 0$. ✔ **Cierto**. Considérese, por ejemplo, el caso en el que $f(n) = g(n) = n$.

a. **Cierto**. Considérese, por ejemplo, el caso en el que $f(n) = 1$ y $g(n) = n$.
b. **Falso**. $g(n)$ crecerá, al menos, tan rápido con $f(n)$, por lo que este límite nunca puede irse a ∞.
c. **Falso**. $g(n)$ crecerá, al menos, tan rápido con $f(n)$, por lo que este límite nunca puede irse a ∞.
d. **Falso**. Considérese, por ejemplo, el caso en el que $f(n) = g(n) = n$.
e. **Cierto**. Considérese, por ejemplo, el caso en el que $f(n) = g(n) = n$.

Las respuestas correctas son: $lim_{n\to\infty} \frac{f(n)}{g(n)}$ puede ser igual a 0.
, $lim_{n\to\infty} \frac{f(n)}{g(n)}$ puede ser igual a un valor finito $c > 0$.

Pregunta 10

Parcialmente correcta

Se puntúa 0,67 sobre 1,00

Marcar pregunta

Supongamos que hemos almacenado un vector de enteros en las n primeras posiciones de un array a. Sabemos que en este vector hay al menos un valor par. ¿Cuáles de las siguientes afirmaciones podemos asegurar que se cumplen?

Seleccione una o más de una:

- [x] a. El predicado $(\#\ i : 0 \leq i < n : a[i]\%2 = 0) > 0$ es $cierto$. ✔ **Cierto**. Porque al haber, al menos, un par, el número de pares debe ser, como mínimo, 1.
- [x] b. El predicado $\exists i : 0 \leq i < n : a[i]\%2 = 0$ es $cierto$. ✔ **Cierto**. Porque hay, como mínimo, un par.
- [ ] c. El predicado $\forall i : 0 \leq i < n : a[i]\%2 = 0$ es $falso$.
- [ ] d. La expresión $\min\ i : 0 \leq i < n : a[i]$ está definida.
- [ ] e. El predicado $(\min\ i : 0 \leq i < n : a[i])\%2 = 0$ es $cierto$.

a. **Cierto**. Porque al haber, al menos, un par, el número de pares debe ser, como mínimo, 1.
b. **Cierto**. Porque hay, como mínimo, un par.
c. **Falso**. Puede ocurrir que todos los valores sean pares.
d. **Cierto**. Porque al haber, al menos, un par, hay al menos un valor, por lo que el mínimo estará definido.
e. **Falso**. El mínimo no tiene porque ser par.

Las respuestas correctas son: El predicado $(\#\ i : 0 \leq i < n : a[i]\%2 = 0) > 0$ es $cierto$.
, El predicado $\exists i : 0 \leq i < n : a[i]\%2 = 0$ es $cierto$.
, La expresión $\min\ i : 0 \leq i < n : a[i]$ está definida.