Evaluación Continua Análisis de Eficiencia y Complejidad

Fecha de Creación	@24 de septiembre de 2023 15:59
Asignatura	FAL
Fecha de Modificación	@24 de septiembre de 2023 16:01

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

Sea $t(n) = n\,log_2\,n + 5n + 2log_2 n + 18$ la función de tiempo de un algoritmo A. El coste de A está en:

Seleccione una o más de una:

- a. $\Theta(log\,n)$
- b. $O(n^2)$
- c. $\Theta(n\,log\,n)$ ✔ **Cierto**. Es el término que domina el crecimiento de la función.
- d. $\Omega(n)$ ✔ **Cierto**. La función contiene términos que crecen más rápido que n. Por tanto, su comportamiento temporal puede acotarse inferiormente mediante una función del orden de n.
- e. $\Theta(1)$

a. **Falso**. La función contiene términos que crecen más rápido que $log\,n$.
b. **Cierto**. Todos los términos de la función crecen más despacio que n^2. De hecho, la función está en $\Theta(n\,log\,n)$, y, por tanto, en $O(n\,log\,n) \subset O(n^2)$.
c. **Cierto**. Es el término que domina el crecimiento de la función.
d. **Cierto**. La función contiene términos que crecen más rápido que n. Por tanto, su comportamiento temporal puede acotarse inferiormente mediante una función del orden de n.
e. **Falso**. La función contiene términos no constantes.

Las respuestas correctas son: $O(n^2)$
, $\Theta(n\,log\,n)$
, $\Omega(n)$

Pregunta 2
Correcta
Se puntúa 1,00 sobre 1,00
Desmarcar

¿Cuáles de las siguientes afirmaciones son verdaderas?:

Seleccione una o más de una:

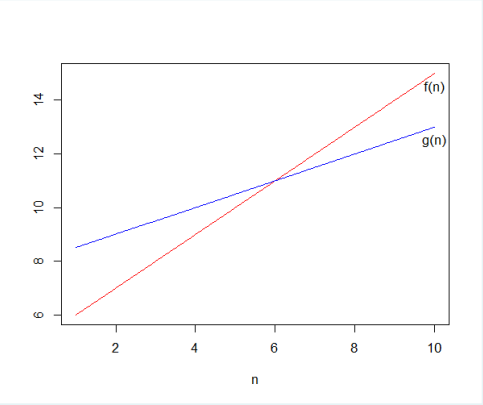
- a. El coste de la condición de una instrucción if es siempre $\Theta(1)$.
- b. El coste de un bucle puede siempre computarse independientemente del contexto en el que aparece dicho bucle.
- c. Un algoritmo con bucles anidados tendrá necesariamente un coste superior al lineal.
- d. Un algoritmo sin bucles ni llamadas a funciones tiene necesariamente un coste constante. ✔ **Cierto**. La ejecución de este tipo de algoritmos se traducirá a la ejecución de secuencias de instrucciones básicas de longitud máxima independiente del tamaño de los datos de entrada. Como consecuencia, el coste estará en $\Theta(1)$
- e. El coste de un bucle que no contiene bucles anidados es necesariamente lineal.

a. **Falso**. En la condición puede invocarse a una función que tiene un coste arbitrario (lineal, cuadrático...).
b. **Falso**. El bucle puede depender (y lo más normal es que dependa) de otras variables del programa, y, por tanto, su ejecución dependerá del contexto (es decir, de aquello que le rodea).
c. **Falso**. El coste dependerá del número de iteraciones y del coste de cada iteración, no de la estructura sintáctica del algoritmo.
d. **Cierto**. La ejecución de este tipo de algoritmos se traducirá a la ejecución de secuencias de instrucciones básicas de longitud máxima independiente del tamaño de los datos de entrada. Como consecuencia, el coste estará en $\Theta(1)$
e. **Falso**. El coste dependerá del número de iteraciones y del coste de cada iteración, no de la estructura sintáctica del algoritmo.

La respuesta correcta es: Un algoritmo sin bucles ni llamadas a funciones tiene necesariamente un coste constante.

Pregunta 3
Incorrecta
Se puntúa 0,00 sobre 1,00
Marcar pregunta

Considera las funciones $f(n)$ y $g(n)$ representadas en las siguiente gráfica:



¿Cuáles de las siguientes afirmaciones son ciertas?:

Seleccione una o más de una:

- a. $g(n)$ crece asintóticamente más deprisa que $f(n)$
- b. $f(n)$ crece asintóticamente más deprisa que $g(n)$ ✖ **Falso**. Ambas funciones son lineales.
- c. $O(f(n)) = O(g(n))$
- d. $f(n) \notin O(g(n))$ ✖ **Falso**. Ambas funciones son lineales.
- e. $g(n) \in O(f(n))$

a. **Falso**. Ambas funciones son lineales.
b. **Falso**. Ambas funciones son lineales.
c. **Cierto**. Ambas funciones son lineales.
d. **Falso**. Ambas funciones son lineales.
e. **Cierto**. Ambas funciones son lineales.

Las respuestas correctas son: $O(f(n)) = O(g(n))$
, $g(n) \in O(f(n))$

Pregunta 4
Parcialmente correcta
Se puntúa 0,17 sobre 1,00
Marcar pregunta

El siguiente algoritmo determina si un vector de enteros está o no ordenado crecientemente:

```
bool esta_ordenado(int a[], int n) {
    bool resul = true;
    int i = 0;
    while (resul && i < n) {
        int j = i + 1;
        while (resul && j < n) {
            resul = (a[i] <= a[j]);
            j++;
        }
        i++;
    }
    return resul;
}
```

Suponiendo que el algoritmo se ejecuta siempre sobre vectores con, al menos, dos elementos, el bucle interno:

Seleccione una o más de una:

- a. En el mejor de los casos, se ejecuta 1 vez. ✔ **Cierto**. En el mejor caso, el primer elemento es mayor que el segundo, hecho que se descubre en la primera iteración del bucle interno.
- b. En el peor de los casos, se ejecuta $\frac{n(n-1)}{2}$ veces.
- c. En el peor de los casos, se ejecuta $\frac{n(n+1)}{2}$ veces. ✖ **Falso**. En el peor de los casos se ejecuta $\frac{n(n-1)}{2}$ veces.
- d. En el mejor de los casos, se ejecuta n veces.
- e. En el peor de los casos, se ejecuta n veces.

a. **Cierto**. En el mejor caso, el primer elemento es mayor que el segundo, hecho que se descubre en la primera iteración del bucle interno.
b. **Cierto**. El bucle externo se ejecuta n veces. Cuando $i = 0$, el bucle interno se ejecuta $n-1$ veces, cuando $i = 1$, $n-2$ veces, cuando $i = 2$, $n-3$ veces, ..., cuando $i = k$, $n-(k+1)$ veces, ... cuando $i = n-1$, $n-(n-1+1) = 0$ veces. Por tanto, el bucle interno se ejecutará $\sum_{i=0}^{n-1}(n-(i+1)) = \sum_{i=0}^{n-1} n - \sum_{i=0}^{n-1} i - \sum_{i=0}^{n-1} 1 = n^2 - \frac{n(n-1)}{2} - n = \frac{n(n-1)}{2}$ veces.
c. **Falso**. En el peor de los casos se ejecuta $\frac{n(n-1)}{2}$ veces.
d. **Falso**. En el mejor caso se ejecuta 1 vez.
e. **Falso**. En el peor de los casos se ejecuta $\frac{n(n-1)}{2}$ veces.

Las respuestas correctas son: En el mejor de los casos, se ejecuta 1 vez.
, En el peor de los casos, se ejecuta $\frac{n(n-1)}{2}$ veces.

Pregunta 5

Correcta

Se puntúa 1,00 sobre 1,00

Marcar pregunta

Sabiendo que $lim_{n\to\infty}\frac{f(n)}{g(n)} = 5$, ¿cuáles de las siguientes afirmaciones son verdaderas?:

Seleccione una o más de una:

- [x] a. $g(n) \in \Theta(f(n))$ ✔ **Cierto**. El comportamiento asintótico de $f(n)$ y $g(n)$ es equivalente.
- [ ] b. $f(n) \notin O(g(n))$ y $g(n) \notin O(f(n))$
- [ ] c. $f(n) \notin O(g(n))$ y $g(n) \in O(f(n))$
- [ ] d. $f(n) \in O(g(n))$ y $g(n) \notin O(f(n))$
- [x] e. $f(n) \in O(g(n))$ y $g(n) \in O(f(n))$ ✔ **Cierto**. El comportamiento asintótico de $f(n)$ y $g(n)$ es equivalente.

a. **Cierto**. El comportamiento asintótico de $f(n)$ y $g(n)$ es equivalente.
b. **Falso**. El comportamiento asintótico de $f(n)$ y $g(n)$ es equivalente.
c. **Falso**. El comportamiento asintótico de $f(n)$ y $g(n)$ es equivalente.
d. **Falso**. El comportamiento asintótico de $f(n)$ y $g(n)$ es equivalente.
e. **Cierto**. El comportamiento asintótico de $f(n)$ y $g(n)$ es equivalente.

Las respuestas correctas son: $g(n) \in \Theta(f(n))$
, $f(n) \in O(g(n))$ y $g(n) \in O(f(n))$

Pregunta 6

Incorrecta

Se puntúa 0,00 sobre 1,00

Marcar pregunta

¿Cuáles de las siguientes afirmaciones son ciertas?:

Seleccione una o más de una:

- [ ] a. Hay funciones que están en $O(f(n))$, pero no en $\Omega(f(n))$.
- [x] b. Hay funciones que están en $\Theta(f(n))$, pero no en $O(f(n))$. ✖ **Falso**. $\Theta(f(n)) = O(f(n)) \cap \Omega(f(n)) \subseteq O(f(n))$.
- [ ] c. Hay funciones que están en $\Omega(f(n))$, pero no en $\Theta(f(n))$.
- [ ] d. Hay funciones que están en $O(f(n))$, pero no en $\Theta(f(n))$.
- [ ] e. Hay funciones que están en $\Theta(f(n))$, pero no en $\Omega(f(n))$.

a. **Cierto**. Por ejemplo, $n \in O(n^2)$, pero $n \notin \Omega(n^2)$.
b. **Falso**. $\Theta(f(n)) = O(f(n)) \cap \Omega(f(n)) \subseteq O(f(n))$.
c. **Cierto**. Por ejemplo, $n^2 \in \Omega(n)$, pero $n^2 \notin O(n)$, y, por tanto, $n^2 \notin \Theta(n)$.
d. **Cierto**. Por ejemplo, $n \in O(n^2)$, pero $n \notin \Omega(n^2)$, y, por tanto, $n \notin \Theta(n^2)$.
e. **Falso**. $\Theta(f(n)) = O(f(n)) \cap \Omega(f(n)) \subseteq \Omega(f(n))$.

Las respuestas correctas son: Hay funciones que están en $O(f(n))$, pero no en $\Omega(f(n))$.
, Hay funciones que están en $\Omega(f(n))$, pero no en $\Theta(f(n))$.
, Hay funciones que están en $O(f(n))$, pero no en $\Theta(f(n))$.

Pregunta 7

Correcta

Se puntúa 1,00 sobre 1,00

Marcar pregunta

El siguiente algoritmo determina si un vector de enteros está o no ordenado crecientemente:

```
bool esta_ordenado(int a[], int n) {
    bool resul = true;
    int i = 1;
    while (resul && i < n) {
        resul = (a[i] >= a[i - 1]);
        i++;
    }
    return resul;
}
```

Suponiendo que siempre se ejecuta sobre vectores con, al menos, dos elementos, ¿cuáles de las siguientes afirmaciones son ciertas?:

Seleccione una o más de una:

- [ ] a. En el mejor de los casos, el coste de este algoritmo está en $\Theta(n^2)$.
- [ ] b. En el peor de los casos, el coste de este algoritmo está en $\Theta(n^2)$.
- [ ] c. En el mejor de los casos, el coste de este algoritmo está en $\Theta(n)$.
- [x] d. En el mejor de los casos, el coste de este algoritmo está en $\Theta(1)$. ✔ **Cierto**. En el mejor de los casos el algoritmo realiza una única iteración, y el coste de esta iteración es constante.
- [x] e. En el peor de los casos, el coste de este algoritmo está en $\Theta(n)$. ✔ **Cierto**. En el peor de los casos, el algoritmo realiza n iteraciones, siendo el coste de cada iteración $\Theta(1)$.

a. **Falso**. En el mejor de los casos el coste es constante.
b. **Falso**. En el peor de los casos, el coste es lineal.
c. **Falso**. En el mejor de los casos el coste es constante.
d. **Cierto**. En el mejor de los casos el algoritmo realiza una única iteración, y el coste de esta iteración es constante.
e. **Cierto**. En el peor de los casos, el algoritmo realiza n iteraciones, siendo el coste de cada iteración $\Theta(1)$.

Las respuestas correctas son: En el mejor de los casos, el coste de este algoritmo está en $\Theta(1)$.
, En el peor de los casos, el coste de este algoritmo está en $\Theta(n)$.

?

Pregunta 8
Correcta
Se puntúa 1,00 sobre 1,00
Marcar pregunta

Se dispone de dos algoritmos, A_0 y A_1 para resolver un mismo problema P. Las funciones de tiempo de estos algoritmos son, respectivamente, $t_0(n) = 12 \times (log_2(n))^2$ y $t_1(n) = 144 \times log_2(n)$. ¿A partir de qué tamaño n comienza a ser más rentable utilizar el algoritmo A_1?

Seleccione una o más de una:

- a. 4093
- b. 4096
- c. 4101
- d. 4100
- e. Ninguna de las anteriores ✔

Cierto: La solución correcta es 4097, ya que $t_0(4096) = t_1(4096)$ y $t_0(4097) > t_1(4097)$, por lo que 4097 es el primer n a partir del cual es más rentable utilizar A_1, ya que, hasta 4096, A_0 no se comporta peor que A_1, pero a partir de $n = 4097$, A_0 siempre tardará más en resolver el problema que A_1.

a. **Falso**, ya que $t_0(4093) < t_1(4093)$, por lo que resulta más rentable usar A_0, ya que tarda menos tiempo en resolver el problema.
b. **Falso**, ya que $t_0(4096) = t_1(4096)$, por lo que ambos algoritmos resultan igual de rentables.
c. **Falso**, ya que $t_0(4100) > t_1(4100)$, por lo que hay tamaños de problemas más pequeños que 4101 para los que usar A_1 resulta más rentable.
d. **Falso**, ya que $t_0(4099) > t_1(4099)$, por lo que hay tamaños de problemas más pequeños que 4100 para los que usar A_1 resulta más rentable.
e. **Cierto**: La solución correcta es 4097, ya que $t_0(4096) = t_1(4096)$ y $t_0(4097) > t_1(4097)$, por lo que 4097 es el primer n a partir del cual es más rentable utilizar A_1, ya que, hasta 4096, A_0 no se comporta peor que A_1, pero a partir de $n = 4097$, A_0 siempre tardará más en resolver el problema que A_1.

La respuesta correcta es: Ninguna de las anteriores

Pregunta 9
Incorrecta
Se puntúa 0,00 sobre 1,00
Marcar pregunta

El siguiente algoritmo calcula la diferencia entre el número de elementos pares y el número de elementos impares en un vector almacenado en los n primeros elementos de un array de enteros a:

```
int dif_pares_impares(int a[], unsigned int n) {
    int i = 0;
    int pares = 0;
    while (i < n) {
        if (a[i] % 2 == 0) {
            pares++;
        }
        i++;
    }
    i = 0;
    int impares = 0;
    while (i < n) {
        if (a[i] % 2 != 0) {
            impares++;
        }
        i++;
    }
    return pares - impares;
}
```

El coste de este algoritmo está en:

Seleccione una o más de una:

- a. En el peor de los casos, el coste de este algoritmo está en $\Theta(n)$.
- b. En el mejor de los casos, el coste de este algoritmo está en $\Theta(n)$.
- c. En el peor de los casos, el coste de este algoritmo está en $\Theta(n^2)$. ✖ **Falso**. El coste de este algoritmo es lineal
- d. En el mejor de los casos, el coste de este algoritmo está en $\Theta(n^2)$. ✖ **Falso**. El coste del algoritmo es lineal (no hay diferencia entre mejor y peor caso).
- e. En el mejor de los casos, el coste de este algoritmo está en $\Theta(1)$.

a. **Cierto**. El algoritmo encadena un bloque con coste constante, seguido del primer bucle (coste lineal), seguido de un tercer bloque con coste constante, seguido del segundo bucle (coste lineal), seguido de una finalización con coste constante. En total, el coste está en $\Theta(n)$.
b. **Cierto**. El coste del algoritmo es lineal (no hay diferencia entre mejor y peor caso).
c. **Falso**. El coste de este algoritmo es lineal
d. **Falso**. El coste del algoritmo es lineal (no hay diferencia entre mejor y peor caso).
e. **Falso**. El coste del algoritmo es lineal (no hay diferencia entre mejor y peor caso).

Las respuestas correctas son: En el peor de los casos, el coste de este algoritmo está en $\Theta(n)$.
, En el mejor de los casos, el coste de este algoritmo está en $\Theta(n)$.

Pregunta 10

Correcta

Se puntúa 1,00 sobre 1,00

Marcar pregunta

El siguiente algoritmo determina si un vector de enteros está o no ordenado crecientemente:

```
bool esta_ordenado(int a[], int n) {
    bool resul = true;
    int i = 0;
    while (resul && i < n) {
        int j = i + 1;
        while (resul && j < n) {
            resul = (a[i] <= a[j]);
            j++;
        }
        i++;
    }
    return resul;
}
```

Suponiendo que siempre se ejecuta con vectores que tienen, al menos, dos elementos, el coste de este algoritmo está en:

Seleccione una o más de una:

- a. En el mejor de los casos, el coste de este algoritmo está en $\Theta(n^2)$.
- b. En el peor de los casos, el coste de este algoritmo está en $\Theta(n)$.
- c. En el mejor de los casos, el coste de este algoritmo está en $\Theta(1)$. ✔ **Cierto**. El coste del algoritmo es constante en el mejor caso: si el primer elemento del vector es mayor que el segundo, entra en el bucle externo, entra en el bucle interno, descubre el desorden, sale del bucle interno, sale del bucle externo, y termina. Todo este proceso supone un coste constante, que no depende del tamaño del vector.
- d. En el peor de los casos, el coste de este algoritmo está en $\Theta(n^2)$. ✔ **Cierto**. El coste del algoritmo es cuadrático en el peor caso.
- e. En el mejor de los casos, el coste de este algoritmo está en $\Theta(n)$.

a. **Falso**. El coste del algoritmo es constante en el mejor caso.
b. **Falso**. El coste del algoritmo es cuadrático en el peor caso.
c. **Cierto**. El coste del algoritmo es constante en el mejor caso: si el primer elemento del vector es mayor que el segundo, entra en el bucle externo, entra en el bucle interno, descubre el desorden, sale del bucle interno, sale del bucle externo, y termina. Todo este proceso supone un coste constante, que no depende del tamaño del vector.
d. **Cierto**. El coste del algoritmo es cuadrático en el peor caso.
e. **Falso**. El coste del algoritmo es constante en el mejor caso.

Las respuestas correctas son: En el mejor de los casos, el coste de este algoritmo está en $\Theta(1)$.
, En el peor de los casos, el coste de este algoritmo está en $\Theta(n^2)$.