

<b>Comenzado el</b>	viernes, 10 de enero de 2025, 13:22
<b>Estado</b>	Finalizado
<b>Finalizado en</b>	viernes, 10 de enero de 2025, 13:26
<b>Tiempo empleado</b>	3 minutos 37 segundos
<b>Calificación</b>	<b>8,00</b> de 10,00 ( <b>80%</b> )


**Pregunta 1**

Correcta

Se puntúa 1,00 sobre 1,00

Considerando el algoritmo de ordenación rápida o quicksort para un vector de tamaño mayor que 1, ¿cuál de estas afirmaciones es cierta? `

Seleccione una:

- ☐ a. Sólo se puede implementar de manera recursiva.
- ☐ b. Su coste en el mejor de los casos es  $\theta(1)$ .
- ☒ c. Su coste en el caso peor es  $\theta(n^2)$ .  Cierto. El peor caso de quicksort es cuando el pivote no divide el espacio de búsqueda en dos mitades sino que queda en un extremo. Por ejemplo si elegimos de pivote el primer elemento y el vector está ordenado ya. En esta circunstancia el coste es  $\theta(n^2)$ .
- ☐ d. Ninguna de las anteriores.

- a. Falso. Incorrecta. Quicksort es un algoritmo divide y vencerás que se puede implementar entero iterativamente como cualquier otro algoritmo Divide y Vencerás.
- b. Falso. Incorrecta. El mejor caso de quicksort es cuando el pivote divide el espacio de búsqueda en dos mitades. En este caso es  $\theta(n \log n)$ , nunca  $\theta(1)$ .
- c. Cierto. El peor caso de quicksort es cuando el pivote no divide el espacio de búsqueda en dos mitades sino que queda en un extremo. Por ejemplo si elegimos de pivote el primer elemento y el vector está ordenado ya. En esta circunstancia el coste es  $\theta(n^2)$ .
- d. Falso. La respuesta correcta es: Su coste en el caso peor es  $\theta(n^2)$ .

La respuesta correcta es: Su coste en el caso peor es  $\theta(n^2)$ .

**Pregunta 2**

Correcta

Se puntúa 1,00 sobre 1,00

Indica cuál de las siguientes es una función de cota que permite demostrar la terminación de este bucle, suponiendo que  $0 \leq n < \text{long}(v)$ :

```
int i = 0, j = n;
while (i < j){
    if (v[i]%10 > v[j]%10)
        j-- ;
    i++;
}
```

Seleccione una:

- ☒ a.  $n - i + 1$  ✓ Cierto. La variable  $j$  empieza en  $n$  y disminuye en algunas ocasiones, mientras que la variable  $i$  aumenta desde el valor 0 en cada iteración del bucle. En el caso peor, la  $j$  no disminuye nunca por lo que el bucle para cuando  $i = j = n$ , por lo tanto la expresión  $n - i + 1$  es una cota válida
- ☐ b.  $n - i - 5$ .
- ☐ c.  $n - j + 1$ .
- ☐ d. Ninguna de las anteriores.

- a. Cierto. La variable  $j$  empieza en  $n$  y disminuye en algunas ocasiones, mientras que la variable  $i$  aumenta desde el valor 0 en cada iteración del bucle. En el caso peor, la  $j$  no disminuye nunca por lo que el bucle para cuando  $i = j = n$ , por lo tanto la expresión  $n - i + 1$  es una cota válida
- b. Falso. Si  $n \leq 4$ , el valor inicial de  $i$  es 0, y esta expresión comienza siendo negativa, por lo que no es una cota válida
- c. Falso. La variable  $j$  no decrece en todas las iteraciones, depende de los valores del vector  $v$ . Además, para que la expresión sea una función de cota válida, debe decrecer en cada iteración y esto no está garantizado. Por ello no puede ser una función de cota.
- d. Falso. La respuesta correcta es:  $n - i + 1$ .

La respuesta correcta es:  $n - i + 1$ .

**Pregunta 3**

Sin contestar

Se puntúa como 0 sobre 1,00

Supuesto  $n \geq 0$  y  $c > 2$ , ¿qué se puede afirmar sobre la terminación del siguiente algoritmo?

```
int f(int n, int c){  
    if ( n > 10 )  
        return c;  
    else  
        return f (n/10, c+n%10);  
}
```

Seleccione una:

- ☐ a. No termina únicamente para los valores de  $n$  entre 0 y 10
- ☐ b. Termina únicamente para los valores de  $n$  entre 0 y 9
- ☐ c. No termina únicamente para los valores de  $n$  mayores que 10
- ☐ d. Ninguna de las anteriores.

- a. Cierto. El caso base es cuando el parámetro  $n > 10$ , por lo tanto para esos casos termina siempre. Cuando no termina es si  $0 \leq n \leq 10$  puesto que en esos valores se vuelve a llamar con  $n/10$  que será un valor entre 0 y 1.
- b. Falso. Para esos valores la función llamará a sí misma con  $n/10 = 0$  y como  $0/10 = 0$  se llamará a sí misma infinitas veces.
- c. Falso. El caso base es cuando el parámetro  $n > 10$ , por lo tanto para esos casos termina siempre.
- d. Falso. La respuesta correcta es: No termina únicamente para los valores de  $n$  entre 0 y 10

La respuesta correcta es: No termina únicamente para los valores de  $n$  entre 0 y 10

**Pregunta 4**


Correcta

Se puntúa 1,00 sobre 1,00

¿Cuál es el coste de la siguiente función recursiva?

```
int f(int n, int k){  
    if ( n < 10 )  
        return 10*k + 1;  
    else if( k < n % 10 )  
        return f(n/10, k + 1);  
    else  
        return f (n/10, k);  
}
```

Seleccione una:

- ☒ a.  $\theta(\log n)$   Cierto. El parámetro que determina el coste es  $n$ . La recurrencia es  $T(n) = c_0$  si  $n < 10$ ;  $T(n) = T(n/10) + c_1$  si  $n \geq 10$ , así que por el teorema de la división  $T(n) \in \theta(\log n)$ .
- ☐ b.  $\theta(n)$
- ☐ c.  $\theta(n^2)$
- ☐ d. Ninguna de las anteriores.

- a. Cierto. El parámetro que determina el coste es  $n$ . La recurrencia es  $T(n) = c_0$  si  $n < 10$ ;  $T(n) = T(n/10) + c_1$  si  $n \geq 10$ , así que por el teorema de la división  $T(n) \in \theta(\log n)$ .
- b. Falso. El parámetro que determina el coste es  $n$ . La recurrencia es  $T(n) = c_0$  si  $n < 10$ ;  $T(n) = T(n/10) + c_1$  si  $n \geq 10$ , así que por el teorema de la división  $T(n) \in \theta(\log n)$ .
- c. Falso. El parámetro que determina el coste es  $n$ . La recurrencia es  $T(n) = c_0$  si  $n < 10$ ;  $T(n) = T(n/10) + c_1$  si  $n \geq 10$ , así que por el teorema de la división  $T(n) \in \theta(\log n)$ .
- d. Falso. La respuesta correcta es:  $\theta(\log n)$

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

**Pregunta 5**

Correcta


Se puntúa 1,00 sobre 1,00

Indica la complejidad del siguiente algoritmo

```
int f(int n, int m){
    int z = 0;
    for (int i = -10; i <= n - 9; i += 1)
        z -= 5;
    for (int j = 1; j < m; j *= 3)
        z -= -2;

    return z;
}
```

Seleccione una:

- ☒ a.  $\theta(n + \log m)$   Cierto. Hay dos bucles independientes. El primero da un número proporcional a  $n$  de vueltas, por lo que su coste es  $(n)$ . Mientras que el segundo da un número logarítmico de vueltas con respecto a  $m$ , por lo que su coste es  $(m)$ . Como no sabemos si  $n$  o  $m$  es mayor, el coste es  $(n + m)$ .
- ☐ b.  $\theta(n * m)$
- ☐ c.  $\theta(\log n * m)$
- ☐ d. Ninguna de las anteriores.
- a. Cierto. Hay dos bucles independientes. El primero da un número proporcional a  $n$  de vueltas, por lo que su coste es  $(n)$ . Mientras que el segundo da un número logarítmico de vueltas con respecto a  $m$ , por lo que su coste es  $(m)$ . Como no sabemos si  $n$  o  $m$  es mayor, el coste es  $(n + m)$ .
- b. Falso. Hay dos bucles independientes. El primero da un número proporcional a  $n$  de vueltas, por lo que su coste es  $(n)$ . Mientras que el segundo da un número logarítmico de vueltas con respecto a  $m$ , por lo que su coste es  $(m)$ . Al ser independientes el coste no es el producto sino la suma  $(n + m)$ , es decir, el máximo entre estas expresiones.
- c. Falso. Hay dos bucles independientes. El primero da un número proporcional a  $n$  de vueltas, por lo que su coste es  $(n)$ . Mientras que el segundo da un número logarítmico de vueltas con respecto a  $m$ , por lo que su coste es  $(m)$ . Al ser independientes el coste es la suma  $(n + m)$ .
- d. Falso. La respuesta correcta es  $\theta(n + \log m)$ .

La respuesta correcta es:  $\theta(n + \log m)$

**Pregunta 6**

Sin contestar

Se puntúa como 0 sobre 1,00

Dado el siguiente código,  $0 < r \leq v.size()$ ,  $0 \leq f \leq v.size()$  y  $v$  es un vector de enteros que cumple  $\forall i: 0 \leq i < r: v[i] = 0$ :

```
int n = 0, i = r, j = 0, a = r;
while (a < f){
    if(v[a]%2 == 0)
        ++i;
    else
        ++j;
    if(v[a-r] == 0)
        --i;
    else
        --j;
    if(i < j)
        ++n;
    ++a;
}
```

¿Cuál de estas propiedades es un invariante de este bucle?

Seleccione una:

- ☐ a.  $n = \#p, q: 0 \leq p < q \leq a \wedge q - p = r: P(v, p, q)$ , donde  $P(v, x, y) = (\#s: x \leq s \leq y: v[s]\%2 = 0) < (\#s: x \leq s \leq y: v[s]\%2 \neq 0)$
- ☐ b.  $n = \#p, q: 0 \leq p < q \leq a \wedge q - p = r: P(v, p, q)$ , donde  $P(v, x, y) = (\#s: x \leq s < y: v[s]\%2 = 0) < (\#s: x \leq s < y: v[s]\%2 \neq 0)$
- ☐ c.  $r \leq a < f$
- ☐ d. Ninguna de las anteriores.

a. Falso. Incorrecto. El bucle calcula el número de intervalos de longitud  $r$  tales que el número de valores impares es mayor que el número de valores pares en el vector  $v$  y ese valor se guarda en la variable  $n$ . Por lo tanto, en cada vuelta, en  $n$  se guarda el número de intervalos de longitud  $r$  desde el inicio del vector hasta el índice del bucle, es decir, la variable  $a$ . Aquí llegamos hasta  $a$ , al final del bucle  $a = f$  y  $f$  podría valer hasta  $v.size()$ , si es así el invariante está mal definido por culpa del predicado auxiliar  $P(v, x, y)$  que indica que en el intervalo  $[x, y]$  hay más impares que pares: si llamamos con  $q = a = f = v.size()$  accederíamos a posiciones erróneas del vector. Por eso es incorrecto.

b. Cierto. El bucle calcula el número de intervalos de longitud  $r$  tales que el número de valores impares es mayor que el número de valores pares en el vector  $v$  y ese valor se guarda en la variable  $n$ . Por lo tanto, en cada vuelta, en  $n$  se guarda el número de intervalos de longitud  $r$  desde el inicio del vector hasta el índice del bucle, es decir, la variable  $a$ . Además el predicado auxiliar  $P(v, x, y)$  indica que en el intervalo  $[x, y)$  hay más impares que pares. La respuesta correcta es  $n = \#p, q: 0 \leq p < q < a \wedge q - p = r: P(v, p, q)$  con  $P(v, x, y) = (\#s: x \leq s < y: v[s]\%2 = 0) < (\#s: x \leq s < y: v[s]\%2 \neq 0)$

c. Falso. Incorrecto. El invariante debe ser cierto al terminar el bucle y en ese momento  $a = f$

d. Falso. La respuesta correcta es:  $n = \#p, q: 0 \leq p < q \leq a \wedge q - p = r: P(v, p, q)$ , donde  $P(v, x, y) = (\#s: x \leq s < y: v[s]\%2 = 0) < (\#s: x \leq s < y: v[s]\%2 \neq 0)$

La respuesta correcta es:  $n = \#p, q: 0 \leq p < q \leq a \wedge q - p = r: P(v, p, q)$ , donde  $P(v, x, y) = (\#s: x \leq s < y: v[s]\%2 = 0) < (\#s: x \leq s < y: v[s]\%2 \neq 0)$

**Pregunta 7**

Correcta

Se puntúa 1,00 sobre 1,00

Indica cuales de las siguiente afirmaciones son ciertas con respecto a los algoritmos divide y vencerás:

Seleccione una o más de una:

- ☐ a. Sólo se implementan de forma recursiva
- ☐ b. Son más eficientes que los algoritmos no divide y vencerás que resuelven el mismo problema
- ☒ c. Dividen el problema original en subproblemas cuyo tamaño es una fracción del original ✓ Cierto. Cada subproblema debe tener como tamaño una fracción del original.
- ☒ d. Dividen el problema original en subproblemas que se pueden resolver en paralelo ✓ Cierto. Cada subproblema se debe poder resolver de forma independiente antes de combinar las soluciones para obtener la solución al problema original.

- a. Falso. Divide y vencerás es una estrategia de resolución de problemas. Por ejemplo, la búsqueda binaria se puede implementar de forma iterativa o recursiva.
- b. Falso. Usar divide y vencerás no garantiza mejorar la complejidad.
- c. Cierto. Cada subproblema debe tener como tamaño una fracción del original.
- d. Cierto. Cada subproblema se debe poder resolver de forma independiente antes de combinar las soluciones para obtener la solución al problema original.

Las respuestas correctas son: Dividen el problema original en subproblemas cuyo tamaño es una fracción del original, Dividen el problema original en subproblemas que se pueden resolver en paralelo

**Pregunta 8**

Correcta

Se puntúa 1,00 sobre 1,00

Cuando Lupin abrió la caja fuerte no se lo podía creer la cantidad de joyas (en concreto  $N$ ) de gran valor (en concreto  $v_i$  para cada joya) y tamaños variables (en concreto  $t_i \text{ cm}^3$  para cada joya) que tenía delante de sí. Todo estaba yendo según su plan salvo un pequeño detalle: se le había roto su bolsa de transporte y sólo puede llevarse aquellas joyas que le quepan en los bolsillos (en total  $B \text{ cm}^3$ ).

Lupin es bueno abriendo cajas fuertes y sabiendo el valor y espacio que ocupa cada joya a simple golpe de vista; pero no sabe cómo afrontar este problema. Tú, en cambio, sí sabes que puede resolverse con la estrategia de Vuelta Atrás. Ayuda a Lupin indicándole cual sería la forma más adecuada de representar la solución.

Seleccione una:

- ☐ a. Una variable booleana.
- ☐ b. Este problema no se puede resolver con vuelta atrás.
- ☐ c. Un vector de enteros de tamaño  $N$ .
- ☒ d. Ninguna de las anteriores. ✓ Cierto. La respuesta correcta es: Un vector de booleanos de tamaño  $N$ .

- a. Falso. Incorrecto. Para cada joya decidimos si ésta se coge (valor true) o no se coge (valor false), necesitamos un vector de booleanos y no sólo una variable.
- b. Falso. Incorrecta. Se resuelve eligiendo para cada joya qué pasa si se coge o si no se coge. La solución es un vector de tamaño  $N$  de booleanos que indica si se ha cogido o no la joya  $i$ -ésima.
- c. Falso. Incorrecto. Si bien es necesario un vector de tamaño  $N$ , cada joya sólo se puede o coger o no coger, para dos valores lo más adecuado es un booleano, no un entero. Debe ser un vector de booleanos de tamaño  $N$ .
- d. Cierto. La respuesta correcta es: Un vector de booleanos de tamaño  $N$ .

La respuesta correcta es: Ninguna de las anteriores.

**Pregunta 9**

Correcta

Se puntúa 1,00 sobre 1,00

Tenemos una función  $f$  que sabemos que satisface su especificación, dada por una precondition  $P$  y una postcondición  $Q$ .

¿Cuál de las siguientes afirmaciones es cierta?

Seleccione una:

- ☒ a. La salida de  $f$  está garantizada que cumple  $Q$  únicamente si la entrada cumple  $P$ . ✔ Cierto. Que  $f$  cumpla la especificación significa que si ejecutamos  $f$  en valores que cumplen  $P$  entonces la salida cumplirá  $Q$ . Si llamamos con una entrada que no cumpla  $P$  no podemos afirmar nada: puede suceder que se cumpla  $Q$ , puede ser que la función dé un error, etc.
- ☐ b.  $f$  está obligada a comprobar que la salida cumple  $Q$ .
- ☐ c. La salida de  $f$  siempre cumple  $Q$ .
- ☐ d. Ninguna de las anteriores.

- a. Cierto. Que  $f$  cumpla la especificación significa que si ejecutamos  $f$  en valores que cumplen  $P$  entonces la salida cumplirá  $Q$ . Si llamamos con una entrada que no cumpla  $P$  no podemos afirmar nada: puede suceder que se cumpla  $Q$ , puede ser que la función dé un error, etc.
- b. Falso. Que  $f$  cumpla la especificación significa que si ejecutamos  $f$  en valores que cumplen  $P$  entonces la salida cumplirá  $Q$ . Si bien la función  $f$  podría comprobar si la salida cumple la postcondición, no está obligada a ello.
- c. Falso. Que  $f$  cumpla la especificación significa que si ejecutamos  $f$  en valores que cumplen  $P$  entonces la salida cumplirá  $Q$ . Por lo tanto sólo está garantizado que la salida cumple  $Q$  en aquellas entradas que cumplen  $P$ , en el resto podría no cumplirse o  $f$  dar errores.
- d. Falso. La respuesta correcta es: La salida de  $f$  está garantizada que cumple  $Q$  únicamente si la entrada cumple  $P$ .

La respuesta correcta es: La salida de  $f$  está garantizada que cumple  $Q$  únicamente si la entrada cumple  $P$ .

**Pregunta 10**

Correcta

Se puntúa 1,00 sobre 1,00

Indica cuál de las siguientes afirmaciones es correcta

Seleccione una:

- ☒ a.  $\Omega(n \log n) \subseteq \Omega(n)$  ✔ Cierto. Por el teorema del límite, como  $\lim_{n \rightarrow \infty} \frac{n \log n}{n} = \infty$ ,  $\Omega(n \log n) \subseteq \Omega(n)$ .
- ☐ b.  $\Omega(n) \subseteq \Omega(n^2)$
- ☐ c.  $\theta(n^2) = \theta(n^3)$
- ☐ d. Ninguna de las anteriores.

- a. Cierto. Por el teorema del límite, como  $\lim_{n \rightarrow \infty} \frac{n \log n}{n} = \infty$ ,  $\Omega(n \log n) \subseteq \Omega(n)$ .
- b. Falso. Por el teorema del límite, como  $\lim_{n \rightarrow \infty} \frac{n}{n^2} = 0$ ,  $\Omega(n) \not\subseteq \Omega(n^2)$ .
- c. Falso. Por el teorema del límite, como  $\lim_{n \rightarrow \infty} \frac{n^2}{n^3} = 0$ ,  $\theta(n^2) \not\subseteq \theta(n^3)$ , por lo que no pueden ser iguales.
- d. Falso. La respuesta correcta es:  $\Omega(n \log n) \subseteq \Omega(n)$

La respuesta correcta es:  $\Omega(n \log n) \subseteq \Omega(n)$