

<b>Comenzado el</b>	viernes, 10 de enero de 2025, 12:38
<b>Estado</b>	Finalizado
<b>Finalizado en</b>	viernes, 10 de enero de 2025, 12:46
<b>Tiempo empleado</b>	8 minutos 7 segundos
<b>Calificación</b>	5,33 de 10,00 (53,33%)

**Pregunta 1**

Incorrecta

Se puntúa -0,33 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 = n, j = 1, s = 0;
while (i > 0){
    if (v[i] > v[j] && j <= n){
        s += v[i];
        j++;
    }
    i--;
}
```

Seleccione una:

- ☐ a.  $i$ .
- ☐ b.  $n - i$ .
- ☐ c.  $j$ .
- ☒ d. Ninguna de las anteriores. ✖ Falso. La respuesta correcta es:  $i$ .

- a. Cierto. La variable  $i$  empieza en  $n$  y disminuye en cada vuelta donde además está garantizado que  $i > 0$ . Por eso es una función de cota válida
- b. Falso. La variable  $i$  decrece en cada iteración, por lo que la expresión  $n - i$  aumenta de valor en cada iteración. No puede ser función de cota
- c. Falso. La variable  $j$  no decrece sino que, según algunos valores del vector, aumenta. No puede ser una función de cota
- d. Falso. La respuesta correcta es:  $i$ .

La respuesta correcta es:  $i$ .

**Pregunta 2**

Correcta


Se puntúa 1,00 sobre 1,00

Indica la complejidad del siguiente algoritmo

```
int f(int n, int m){
    int y = 0;
    for (int i = -10; i <= n - 8; i += 2)
        y -= 4;
    for (int j = 1; j < m; j *= 3)
        y += 4;

    return y;
}
```

Seleccione una:

- ☐ a.  $\theta(n + m)$
- ☐ b.  $\theta(n * \log m)$
- ☐ c.  $\theta(\log n * m)$
- ☒ d. Ninguna de las anteriores.  Cierto. La respuesta correcta es  $\theta(n + \log m)$ .

- a. 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  $(n + m)$ .
- 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. Cierto. La respuesta correcta es  $\theta(n + \log m)$ .

La respuesta correcta es: Ninguna de las anteriores.

**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. No termina únicamente para los valores de  $n$  mayores que 10
- ☐ c. Termina únicamente para los valores de  $n$  entre 0 y 9
- ☐ 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. El caso base es cuando el parámetro  $n > 10$ , por lo tanto para esos casos termina siempre.
- c. 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.
- 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

Indica cuál de las siguientes afirmaciones es correcta

Seleccione una:

- ☐ a.  $O(n^{20}) \supseteq O(2^n)$
- ☐ b.  $\Omega(n) \subseteq \Omega(n^2)$
- ☐ c.  $\theta(n^2) = \theta(n^3)$
- ☒ d. Ninguna de las anteriores.  Cierto. La respuesta correcta es:  $\Omega(n \log n) \subseteq \Omega(n)$

- a. Falso. Por el teorema del límite, como  $\lim_{n \rightarrow \infty} \frac{2^n}{n^{20}} = \infty$ ,  $O(2^n) \not\subseteq O(n^{20})$ .
- 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. Cierto. La respuesta correcta es:  $\Omega(n \log n) \subseteq \Omega(n)$

La respuesta correcta es: Ninguna de las anteriores.

**Pregunta 5**


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. Si llamamos con un valor que no cumple  $P$ , el comportamiento de  $f$  no está garantizado.  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 entrada cumple  $P$ .
- ☐ c. Si llamamos con un valor que cumple  $P$ , hay que comprobar si el resultado 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 entrada cumple la precondition, no está obligada a ello, la precondition se da para indicar sobre qué entradas la función al ejecutarse devuelve salidas que cumplen  $Q$ .
- c. Falso. PQue  $f$  cumpla la especificación significa que si ejecutamos  $f$  en valores que cumplen  $P$  entonces la salida cumplirá  $Q$ . Por lo tanto no hay que comprobarlo.
- d. Falso. La respuesta correcta es: Si llamamos con un valor que no cumple  $P$ , el comportamiento de  $f$  no está garantizado.

La respuesta correcta es: Si llamamos con un valor que no cumple  $P$ , el comportamiento de  $f$  no está garantizado.

**Pregunta 6**

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. Dividen el problema original en subproblemas cuyo tamaño es una fracción del original ✓ Cierto. Cada subproblema debe tener como tamaño una fracción del original.
- ☐ b. Son más eficientes que los algoritmos no divide y vencerás que resuelven el mismo problema
- ☒ c. 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.
- ☐ d. Sólo se implementan de forma recursiva

a. Cierto. Cada subproblema debe tener como tamaño una fracción del original.

b. Falso. Usar divide y vencerás no garantiza mejorar la complejidad.

c. Cierto. Cada subproblema se debe poder resolver de forma independiente antes de combinar las soluciones para obtener la solución al problema original.

d. Falso. Divide y vencerás es una estrategia de resolución de problemas. Por ejemplo, la búsqueda binaria se puede implementar de forma iterativa o recursiva.

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 7

Incorrecta

Se puntúa -0,33 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 = 0, j = r, a = r;
while (a < f){
    if(v[a]%2 == 0)
        ++j;
    else
        ++i;
    if(v[a-r] == 0)
        --j;
    else
        --i;
    if(i < j)
        ++n;
    ++a;
}
```

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

Seleccione una:

- ☐ a.  $r < a < f$
- ☐ b.  $(i = \#k: a - r \leq k < a: v[s]\%2 \neq 0) \wedge (j = \#k: a - r \leq k < a: v[s]\%2 = 0)$
- ☒ c.  $n = \#p, q: 0 \leq p < q \leq a \wedge q - p = r: P(v, p, q)$ , donde  $P(v, x, y) = (\#s: x \leq s < y: v[s]\%2 = 0) < (\#s: x \leq s < y: v[s]\%2 \neq 0)$

✗ Falso. Incorrecto. El bucle calcula el número de intervalos de longitud  $r$  tales que el número de valores pares es mayor que el número de valores impares en el vector  $v$  y ese se guarda en la variable  $n$ . Por lo tanto, en cada vuelta se guarda el número de intervalos de longitud  $r$  desde inicio del vector hasta el índice del bucle, es decir, la  $v$   $a$ . El problema es que el predicado auxiliar  $P(v, x, y)$  que en el intervalo  $[x, y)$  hay más impares que pares y  $(\#s: x \leq s < y: v[s]\%2 = 0) < (\#s: x \leq s < y: v[s]\%2 \neq 0)$ . Por eso es incorrecto.

- ☐ d. Ninguna de las anteriores.

- a. Falso. Incorrecto. El invariante debe ser cierto antes de comenzar el bucle y al terminar el bucle. Al comienzo  $a = r$  y al final  $a = f$
- b. Cierto. El bucle calcula el número de intervalos de longitud  $r$  tales que el número de valores pares es mayor que el número de valores impares en el vector  $v$  y ese valor se guarda en la variable  $n$ . Para llevar la cuenta del número de valores pares del intervalo usamos la variable  $j$ , mientras que  $i$  cuenta el número de valores pares. Cada intervalo corresponde a los índices  $[a - r, a)$ , ya que los índices van de 0 a  $v.size()$ . Por ejemplo, al final del bucle  $a = f$  lo que corresponde al último intervalo posible de longitud  $r$   $[f - r, f)$ .
- c. Falso. Incorrecto. El bucle calcula el número de intervalos de longitud  $r$  tales que el número de valores pares es mayor que el número de valores impares 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  $a$ . El problema es que el predicado auxiliar  $P(v, x, y)$  indica que en el intervalo  $[x, y)$  hay más impares que pares ya que  $(\#s: x \leq s < y: v[s]\%2 = 0) < (\#s: x \leq s < y: v[s]\%2 \neq 0)$ . Por eso es incorrecto.
- d. Falso. La respuesta correcta es:  $(i = \#k: a - r \leq k < a: v[s]\%2 \neq 0) \wedge (j = \#k: a - r \leq k < a: v[s]\%2 = 0)$

La respuesta correcta es:  $(i = \#k: a - r \leq k < a: v[s]\%2 \neq 0) \wedge (j = \#k: a - r \leq k < a: v[s]\%2 = 0)$

**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. Un vector de enteros de tamaño  $B$ .
  - ☒ b. Un vector de booleanos de tamaño  $N$ .  Cierto. Para cada joya decidimos si ésta se coge (valor true) o no se coge (valor false).
  - ☐ c. Una variable booleana.
  - ☐ d. Ninguna de las anteriores.
- a. Falso. Incorrecta.  $B$  es el límite de joyas que podemos llevar, lo que necesitamos es mirar con 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.
- b. Cierto. Para cada joya decidimos si ésta se coge (valor true) o no se coge (valor false).
- c. 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.
- d. Falso. La respuesta correcta es: Un vector de booleanos de tamaño  $N$ .

La respuesta correcta es: Un vector de booleanos de tamaño  $N$ .

**Pregunta 9**

Sin contestar

Se puntúa como 0 sobre 1,00

¿Cuál es el coste de la siguiente función recursiva?

```
int f(const vector<int>&v, int c, int a, int b){
    if ( b == a + 1 ) {
        return v[a]*c - 3*c;

    } else{
        int m = (a + b)/2;
        int x = f(v, c*2, a, m);
        int y = f(v, c*2+1, m, b);
        for(int i = a; i <= b; ++i)
            x+= v[a];
        return x - y + c;
    }
}
```

Seleccione una:

- ☐ a.  $\theta(N)$  con  $N = v.size()$ .
- ☐ b.  $\theta(k * N)$  con  $N = v.size()$ .
- ☐ c.  $\theta(k \log n)$  con  $n = b - a$ .
- ☐ d. Ninguna de las anteriores.

- a. Falso. El tamaño de la entrada en la función  $f$  es la longitud del intervalo  $[a, b]$ , es decir,  $n = b - a$ , y no la longitud del vector. Así la recurrencia es  $T(n) = c_0$  si  $n \leq 1$ ;  $T(n) = 2 * T(n/2) + c_1 * n$  si  $n > 1$ , así que por el teorema de la división  $T(n) \in \theta(n \log n)$ .
- b. Falso. El parámetro  $k$  no influye en el número de llamadas que se hacen, así que el coste asintótico no depende de dicho parámetro. Además, el tamaño de la entrada en la función  $f$  es la longitud del intervalo  $[a, b]$ , es decir,  $n = b - a$ , y no la longitud del vector.  $T(n) = c_0$  si  $n \leq 1$ ;  $T(n) = 2 * T(n/2) + c_1 * n$  si  $n > 1$ , así que por el teorema de la división  $T(n) \in \theta(n \log n)$ .
- c. Falso. El parámetro  $k$  no influye en el número de llamadas que se hacen, así que el coste asintótico no depende de dicho parámetro. La recurrencia es  $T(n) = c_0$  si  $n \leq 1$ ;  $T(n) = 2 * T(n/2) + c_1 * n$  si  $n > 1$ , así que por el teorema de la división  $T(n) \in \theta(n \log n)$ .
- d. Cierto. La respuesta correcta es:  $\theta(n \log n)$  con  $n = b - a$ .

La respuesta correcta es: Ninguna de las anteriores.




**Pregunta 10**

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. 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  $\theta(n^2)$ .
- ☐ b. Sólo se puede implementar de manera recursiva.
- ☐ c. Su coste en el mejor de los casos es  $\theta(1)$ .
- ☐ d. Ninguna de las anteriores.

- a. 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  $\theta(n^2)$ .
- b. Falso. Incorrecta. Quicksort es un algoritmo divide y vencerás que se puede implementar entero iterativamente como cualquier otro algoritmo Divide y Vencerás.
- c. 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  $\theta(1)$ .
- 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  $\theta(n^2)$ .