Comenzado el	viernes, 10 de enero de 2025, 12:48
Estado	Finalizado
Finalizado en	viernes, 10 de enero de 2025, 13:06
Tiempo empleado	17 minutos 50 segundos
Calificación	**4,00** de 10,00 (**40**%)

Pregunta 1

Incorrecta

Se puntúa -0,33 sobre 1,00

La tía Josefina tiene un negocio de dulces navideños y necesita repartir P paquetes con sus exquisiteces usando el menor número de camiones de reparto (para ahorrar costes). Todos los camiones de reparto son idénticos: tienen la misma capacidad y su alquiler cuesta lo mismo. Cada paquete tiene un volumen v_i, peso w_i y características diferentes que impiden que podamos meter todos los dulces juntos: por ejemplo, no se puden apilar más de 3 cajas con dulces de hojaldre porque se estropean, etc.

Tu misión es minimizar el dinero que Josefina se gastará usando el menor número de camiones. Como ya sabes mucho de algoritmos te planteas si podrías hacer una poda del espacio de búsqueda. ¿Cuál de las siguientes sería una poda válida?

Seleccione una:

- a. Podar considerando que cada paquete de dulces que queda por repartir necesita un nuevo camión de reparto ✖ Falso. Eso no funcionaría como poda de minimización ya que estaríamos sobreaproximando la cantidad de caminones que necesitamos. Una poda de minimización debe ser una cota inferior al coste, no una superior
- b. Podar considerando la cantidad actual de caminones alquilados + 2
- c. Podar considerando la el peso mínimo de los paquetes multiplicado por el número de camiones ya alquilados
- d. Ninguna de las anteriores.

a. Falso. Eso no funcionaría como poda de minimización ya que estaríamos sobreaproximando la cantidad de caminones que necesitamos. Una poda de minimización debe ser una cota inferior al coste, no una superior

b. Falso. Eso no funcionaría como poda de minimización ya que estaríamos sobreaproximando la cantidad de caminones que necesitamos, puede ser que no necesitemos 2 camiones más sino sólo 1 o ninguno extra. Una poda de minimización debe ser una cota inferior al coste, no una superior

c. Falso. Ese valor no nos indica nada relevante. Buscamos minimizar el alquiler de camiones y nos basta podar considerando el valor de la solución parcial: el número de camiones que hemos alquilado por el momento

d. Cierto. La respuesta correcta es: Podar con la cantidad actual de camiones alquilados

La respuesta correcta es: Ninguna de las anteriores.

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 $0 \leq n < long(v)$:

```
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 ✔ . 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. $n - i$.
- c. $n - j$.
- d. Ninguna de las anteriores.

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 crece únicamente en algunas iteraciones, dependiendo de los valores de v, la función de cota debe decrecer en cada iteración. No puede ser función de cota
d. Falso. La respuesta correcta es: i.

La respuesta correcta es: i.

Pregunta 3

Correcta

Se puntúa 1,00 sobre 1,00

Indica cuál de las siguientes afirmaciones es incorrecta

Seleccione una:

- a. $\Omega(n^5) \subseteq \Omega(n \log n)$
- b. $\Omega(n^3) \supseteq \Omega(n^4)$
- c. $O(\log n) \subseteq O(n^4)$
- d. Ninguna de las anteriores. ✔ Cierto. La respuesta correcta es: $\Omega(\log n) \subseteq \Omega(n)$

a. Falso. Por el teorema del límite, como $\lim_{n \to \infty} \frac{n^5}{n \log n} = \infty$, $\Omega(n^5) \subseteq \Omega(n \log n)$.
b. Falso. Por el teorema del límite, como $\lim_{n \to \infty} \frac{n^4}{n^3} = \infty$, $\Omega(n^4) \subseteq \Omega(n^3)$.
c. Falso. Por el teorema del límite, como $\lim_{n \to \infty} \frac{n^4}{\log n} = \infty$, por lo que $O(\log n) \subseteq O(n^4)$.
d. Cierto. La respuesta correcta es: $\Omega(\log n) \subseteq \Omega(n)$

La respuesta correcta es: Ninguna de las anteriores.

Pregunta 4

Correcta

Se puntúa 1,00 sobre 1,00

Indica la complejidad del siguiente algoritmo

```
int f(int n, int m){
  int x = 0;
  for (int i = -10; i <= n - 8; i += 1)
    x += 3;
  for (int j = 1; j < m; j *= 4)
    x -= 7;

 return x;
}
```

Seleccione una:

- a. $\Theta(n*m)$
- b. $\Theta(n+\log m)$ ✔ 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).
- c. $\Theta(n*\log m)$
- d. Ninguna de las anteriores.

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 no es el producto sino la suma (n +m), es decir, el máximo entre estas expresiones.
b. 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).
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 no es el producto sino la suma (n +m), es decir, el máximo entre estas expresiones.
d. Falso. La respuesta correcta es $\Theta(n+\log m)$.

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

Pregunta 5

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 $\forall i \colon 0 \leq i < r \colon v[i] = 0$:

```
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. $n = \#p, q \colon 0 \leq p < q < a \wedge q - p = r \colon P\left(v, p, q\right)$, donde $P\left(v, x, y\right) = \left(\#s \colon x \leq s < y \colon v[s]\%2 = 0\right) > \left(\#s \colon x \leq s < y \colon v[s]\%2 \neq 0\right)$

✖ 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 índide del bucle, es decir, la variable a. Aquí la variable q como máximo tiene el valor $a - 1$ y al final del bucle $a = f$, por lo que el invariante cuenta mal, no considera el último intervalo. Esto es así por culpa del predicado auxiliar $P\left(v, x, y\right)$ que indica que en el intervalo $[x, y)$ hay más pares que impares. Si llamamos con $q = a - 1 = f - 1$ nunca comprobaríamos el intervalo [f-r, f). Por eso es incorrecto.

b. $\left(i = \#k \colon a - r \leq k < a \colon v[s]\%2 \neq 0\right) \wedge \left(j = \#k \colon a - r \leq k < a \colon v[s]\%2 = 0\right)$

c. $\left(i = \#k \colon 0 \leq k < a \colon v[s]\%2 \neq 0\right) \wedge \left(j = \#k \colon 0 \leq k < a \colon v[s]\%2 = 0\right)$

d. Ninguna de las anteriores.

a. 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 índide del bucle, es decir, la variable a. Aquí la variable q como máximo tiene el valor $a - 1$ y al final del bucle $a = f$, por lo que el invariante cuenta mal, no considera el último intervalo. Esto es así por culpa del predicado auxiliar $P\left(v, x, y\right)$ que indica que en el intervalo $[x, y)$ hay más pares que impares. Si llamamos con $q = a - 1 = f - 1$ nunca comprobaríamos el intervalo [f-r, f). Por eso es incorrecto.

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. Incorecto. 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 variables pares. El intervalo corresponde a los índices $[a - r, a)$ y no $[0, a)$ como indican las expresiones. Por eso no es un invariante del bucle.

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

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

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:

- [x] 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. Sólo se implementan de forma recursiva
- [x] 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. Son más eficientes que los algoritmos no divide y vencerás que resuelven el mismo problema

a. Cierto. Cada subproblema debe tener como tamaño una fracción del original.
b. Falso. Divide y vencerás es una estrategia de resolución de problemas. Por ejemplo, la búsqueda binaria se puede implementar de forma iterativa o recursiva.
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. Usar divide y vencerás no garantiza mejorar la complejidad.

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

Sin contestar

Se puntúa como 0 sobre 1,00

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

Seleccione una:

- a. Su coste en el caso peor es $\Theta(n^2)$.
- b. Su coste es en promedio $\Theta(n^2)$.
- c. Su coste en el mejor de los casos es $\Theta(1)$.
- d. Ninguna de las anteriores.

a. Falso. Incorrecta. Mergesort siempre hace dos llamadas recursivas con la mitad del tamaño y después hace una mezcla ordenada de lo obtenido. Por ello su coste en el peor de los casos también será $\Theta(n\log n)$.
b. Falso. Incorrecta. Mergesort siempre hace dos llamadas recursivas con la mitad del tamaño y después hace una mezcla ordenada de lo obtenido. Por ello su coste en promedio también será $\Theta(n\log n)$.
c. Falso. Incorrecta. Mergesort siempre hace dos llamadas recursivas con la mitad del tamaño y después hace una mezcla ordenada de lo obtenido. Por ello su coste no puede ser nunca constante sino $\Theta(n\log n)$.
d. Cierto. La respuesta correcta es: Su coste es siempre $\Theta(n\log n)$.

La respuesta correcta es: Ninguna de las anteriores.

Pregunta 8

Incorrecta

Se puntúa -0,33 sobre 1,00

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

¿Cuál de las siguientes afirmaciones es cierta?

Seleccione una:

- a. Si llamamos con un valor que no cumple P, habría que comprobar a mano si el resultado cumple Q.
- b. f está obligada a comprobar que la entrada cumple P.
- c. Si llamamos con un valor que no cumple P, la función f dará un error. ✖ Falso. 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 incluso suceder que f se ejecute correctamente y se cumpla Q
- d. Ninguna de las anteriores.

a. Falso. 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. f podría dar error sobre esa entrada por lo que no habría resultado.
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 precondición, no está obligada a ello, la precondición se da para indicar sobre qué entradas la función al ejecutarse devuelve salidas que cumplen Q.
c. Falso. 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 incluso suceder que f se ejecute correctamente y se cumpla Q
d. Cierto. La respuesta correcta es: Si llamamos con un valor que no cumple P, el comportamiento de f no está garantizado.

La respuesta correcta es: Ninguna de las anteriores.

Pregunta 9

Sin contestar

Se puntúa como 0 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. $\Theta(\log n)$
- b. $\Theta(k)$
- c. $\Theta(n)$
- 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 k no influye en el número de llamadas que se hacen, así que el coste asintótico no depende de dicho parámetro.
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 10

Correcta

Se puntúa 1,00 sobre 1,00

Dado la siguiente función recursiva:

```
int fun(const vector<int>&v, int i, int f){
    if ( i+1 == f && v[i]%2 != 0)
       return 1;
    else if ( i+1 == f && v[i]%2 == 0)
       return 0;
    else{
      int m = (f+i)/2;
      return fun(v, i, m) + fun(v, m, f);
   }
}
```

Si $v = 11, 0, 2, 5, 1, 9$, ¿Qué devuelve la llamada $fun\,(\,v, 0, 5\,)$?

Seleccione una:

- a. 3 ✔ Cierto. La función devuelve el número de valores impares que hay en el intervalo [i, f), al llamar con $i = 0$ y $f = 5$. Hay 3 valores impares en ese intervalo: 11, 5 y 1
- b. 2
- c. 5
- d. Ninguna de las anteriores.

a. Cierto. La función devuelve el número de valores impares que hay en el intervalo [i, f), al llamar con $i = 0$ y $f = 5$. Hay 3 valores impares en ese intervalo: 11, 5 y 1
b. Falso. La función devuelve el número de valores impares que hay en el intervalo [i, f), al llamar con $i = 0$ y $f = 5$. Hay 3 valores impares en ese intervalo: 11, 5 y 1
c. Falso. La función devuelve el número de valores impares que hay en el intervalo [i, f), al llamar con $i = 0$ y $f = 5$. Hay 3 valores impares en ese intervalo: 11, 5 y 1
d. Falso. La respuesta correcta es: 3

La respuesta correcta es: 3