**Determinar que hacen los siguientes fragmentos de código y calcular el número total de instrucciones**

**ejecutadas en el peor caso.**

Instrucción (1): Se define la función item1 que toma un parámetro a, que se espera que sea un arreglo.

Costo: Esta operación de definición no tiene un costo relevante en el análisis del rendimiento.

Inicialización de Variables:

Instrucción (2): let res = "Si se cumple"; inicializa res con un mensaje que indica que se cumple una cierta condición.

Instrucción (3): let final = a.length - 1; establece final como el índice del último elemento del arreglo a.

Costo: Ambas instrucciones son O(1), ya que son operaciones de asignación.

Inicialización del Bucle:

Instrucción (4): for (let i = 0; i < a.length / 2; i++) { inicia un bucle que itera desde i = 0 hasta i < a.length / 2. Esto significa que se comprueba solo la mitad del arreglo.

Costo: Se ejecutará a.length / 2 veces.

Comparación de Elementos:

Instrucción (5): if (a[i] != a[final]) { compara el elemento en la posición i con el elemento en la posición final. Si son diferentes, se realiza la siguiente acción.

Actualización de res y Salida del Bucle:

Instrucción (6): res = "no se cumple"; asigna un nuevo mensaje a res si los elementos comparados son diferentes.

Instrucción (7): break; se usa para salir del bucle si se encuentra que los elementos son diferentes.

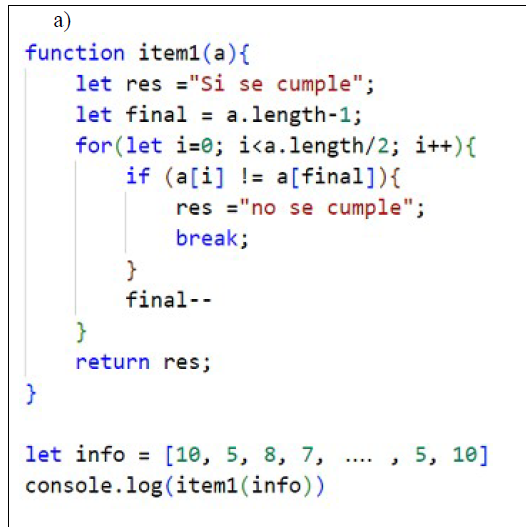
Costo: Ambas operaciones se realizan en el caso de que la condición sea verdadera, lo que puede suceder varias veces pero solo desencadenará una acción de salida del bucle.

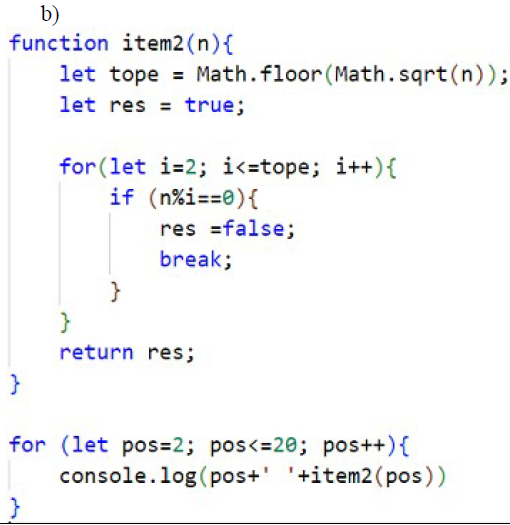
Decremento de final:

Instrucción (8): final--; reduce el índice final en uno después de cada iteración del bucle. Esto permite que la comparación se haga con el elemento correspondiente en la otra mitad del arreglo.

Instrucción (9): return res; devuelve el valor de res al finalizar la ejecución de la función.

Costo: Esta operación es O(1).





Instrucción (1): Se define la función item2 que toma un parámetro n, que es el número a verificar si es primo.

Costo: La definición de la función no tiene un costo relevante.

Cálculo del Límite Superior:

Instrucción (2): let tope = Math.floor(Math.sqrt(n)); calcula la raíz cuadrada de n y la redondea hacia abajo. Esto se usa para limitar el número de iteraciones al verificar factores de n.

Costo: Esta operación es O(1) porque se trata de una sola operación matemática.

Inicialización de res:

Instrucción (3): let res = true; inicializa res como true, asumiendo inicialmente que n es primo.

Costo: Esta operación es O(1).

Bucle de Verificación de Primalidad:

Instrucción (4): for (let i = 2; i <= tope; i++) { inicia un bucle que itera desde i = 2 hasta i igual a tope. Esto verifica si n tiene divisores en este rango.

Verificación de Divisibilidad:

Instrucción (5): if (n % i == 0) { verifica si n es divisible por i. Si es divisible, n no es primo.

Costo: Esta comparación se realiza en cada iteración del bucle, lo que tiene un costo O(1) en cada iteración.

Actualización de res y Salida del Bucle:

Instrucción (6): res = false; asigna false a res si n es divisible por i.

Instrucción (7): break; se usa para salir del bucle si se encuentra que n no es primo.

Costo: Ambas operaciones tienen un costo de O(1) si se ejecutan.

Retorno del Resultado:

Instrucción (8): return res; devuelve el valor de res al finalizar la ejecución de la función.

Costo: Esta operación es O(1).

Bucle para Probar Varios Números:

Instrucción (9): for (let pos = 2; pos <= 20; pos++) { inicia un bucle que itera desde 2 hasta 20 para verificar cada número.

Costo: Este bucle se ejecutará 19 veces, lo que tiene un costo O(1) por cada iteración.

Llamada a la Función y Registro en Consola:

Instrucción (10): console.log(pos + ' ' + item2(pos)); llama a item2 y muestra el resultado en la consola.

Costo: Esta operación es O(1) por cada iteración.Instrucción (6): res = "no se cumple"; asigna un nuevo mensaje a res si los elementos comparados son diferentes.

Instrucción (7): break; se usa para salir del bucle si se encuentra que los elementos son diferentes.

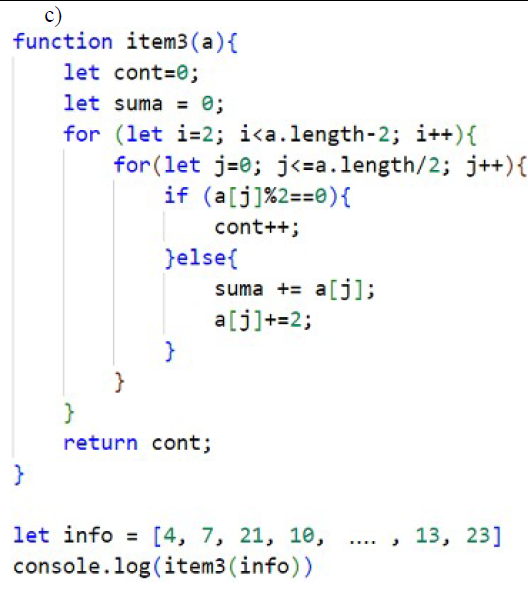
Costo: Ambas operaciones se realizan en el caso de que la condición sea verdadera, lo que puede suceder varias veces pero solo desencadenará una acción de salida del bucle.

Decremento de final:

Instrucción (8): final--; reduce el índice final en uno después de cada iteración del bucle. Esto permite que la comparación se haga con el elemento correspondiente en la otra mitad del arreglo.

Instrucción (9): return res; devuelve el valor de res al finalizar la ejecución de la función.

Costo: Esta operación es O(1).



R/ function item3(a) {

let cont = 0; // 1 instrucción

let suma = 0; // 1 instrucción

for (let i = 2; i < a.length - 2; i++) { // (n - 4) iteraciones

for (let j = 0; j <= a.length / 2; j++) { // (n / 2 + 1) iteraciones por cada i

if (a[j] % 2 == 0) { // 1 instrucción por cada j

cont++; // 1 instrucción si se cumple la condición

} else { // Si no se cumple la condición

suma += a[j]; // 1 instrucción

a[j] += 2; // 1 instrucción

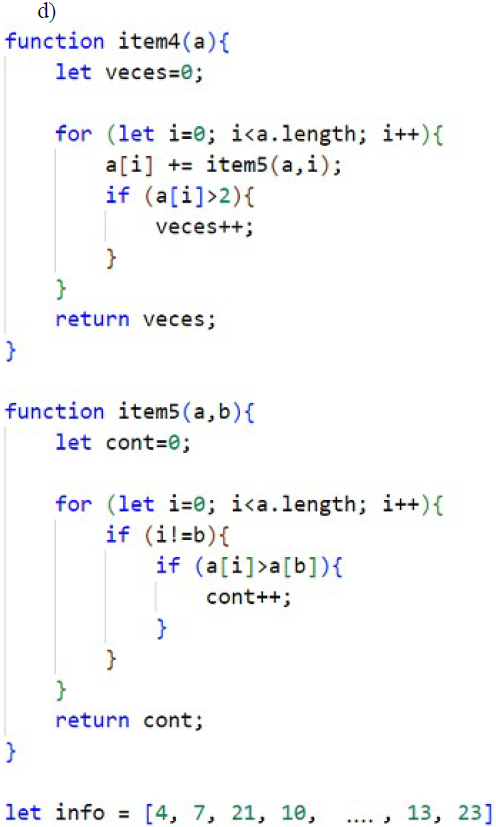
}

}

}

return cont; // 1 instrucción

}



function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

}

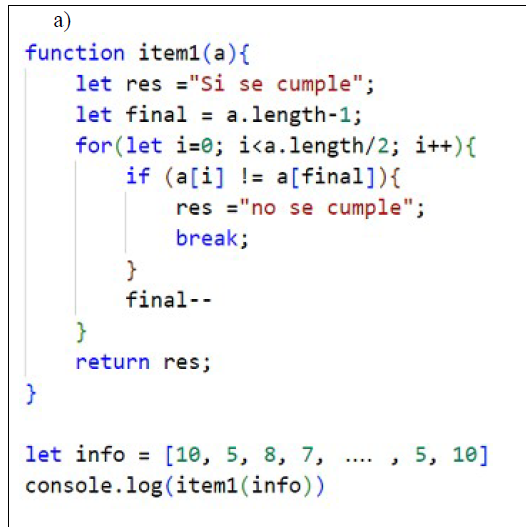
return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));

Determinar la complejidad Big O (O grande) de los siguientes fragmentos de código.



function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

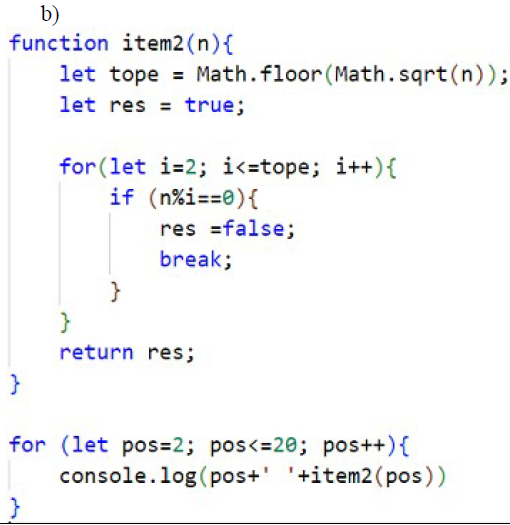
}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));



function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

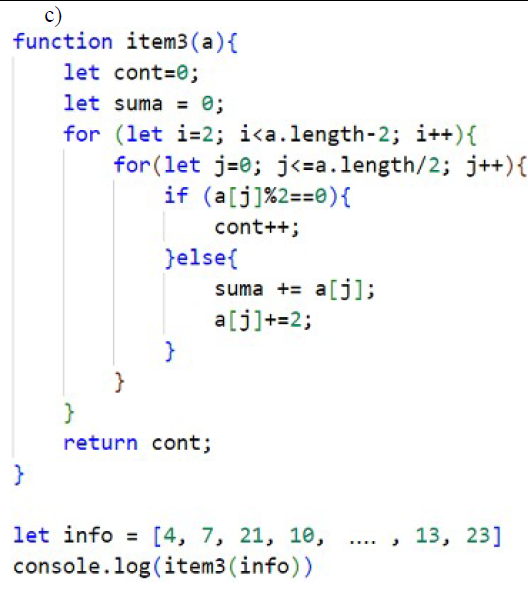
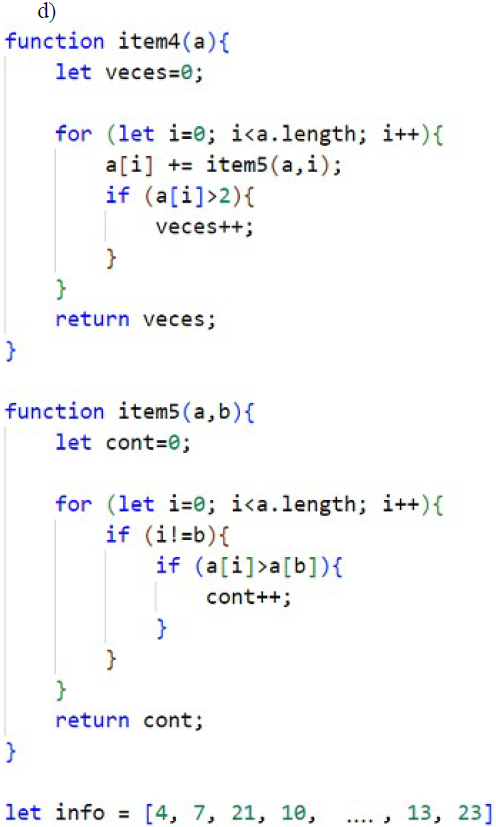
}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));



function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));

function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

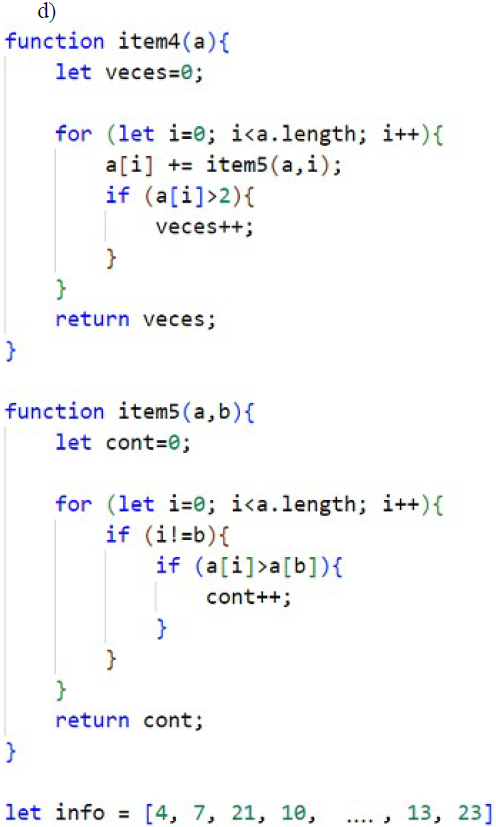
}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));



function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

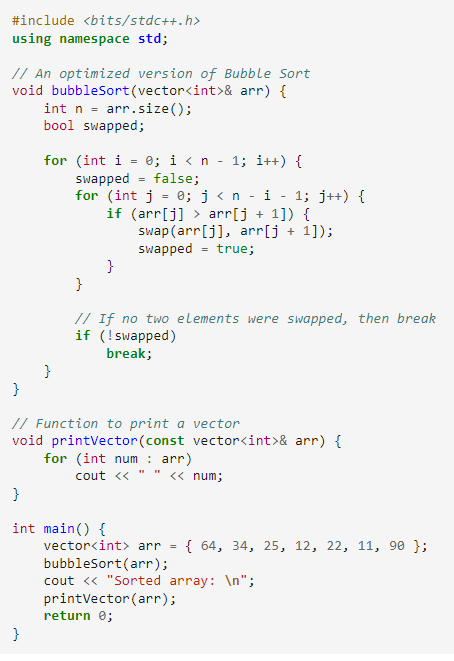
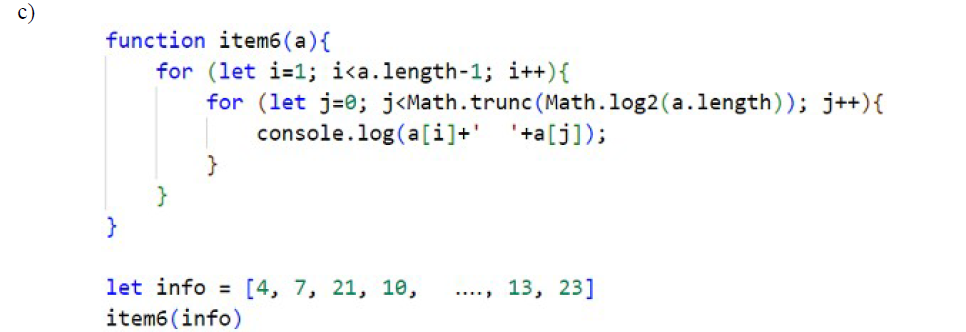
}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));

j

function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

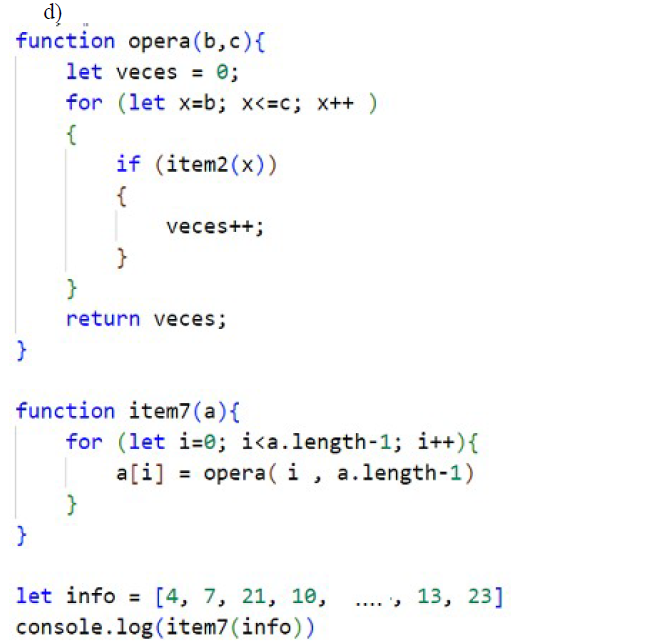
}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));

l

function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));

function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

}

return cont; // 1 instrucción

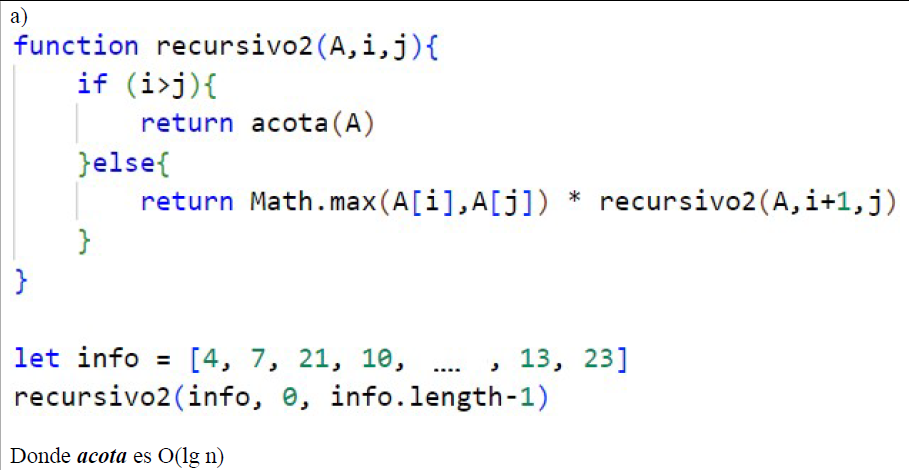
}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));

**Para cada uno de los siguientes algoritmos recursivos, calcular la función de recurrencia que define su tiempo**

**de ejecución y resolver para obtener su complejidad.**

****

f

function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

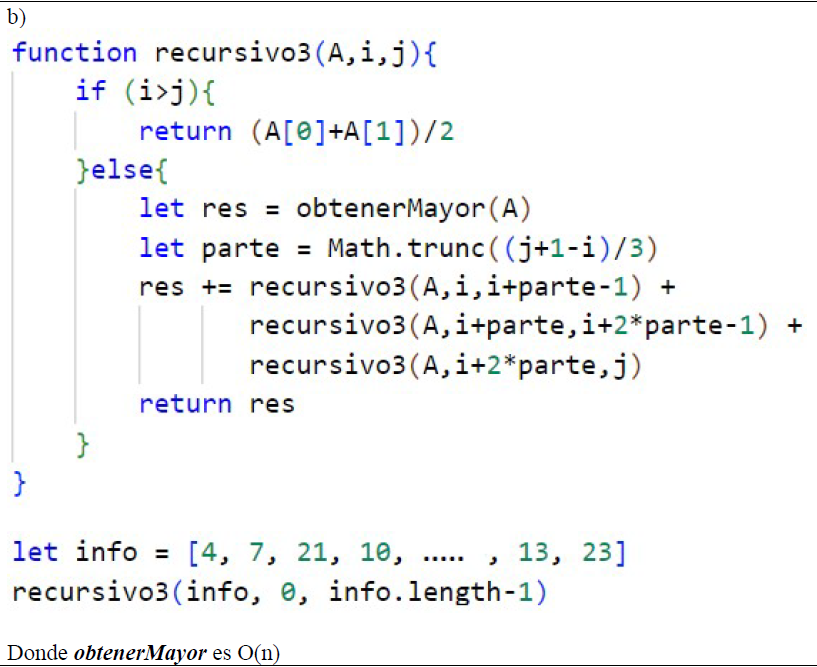
}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));



h

function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

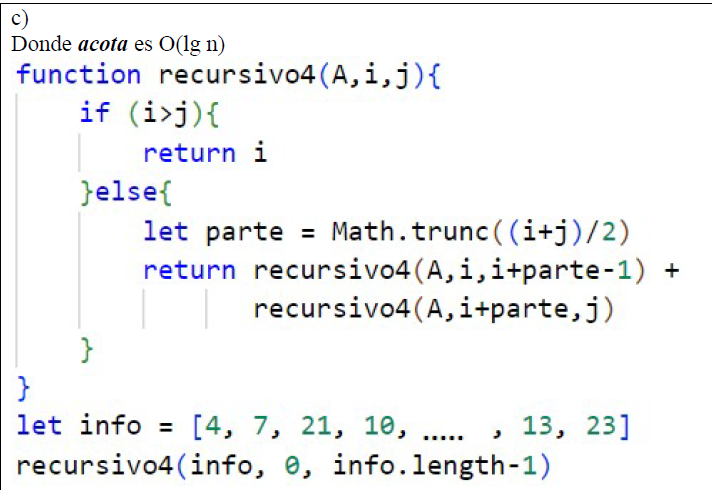
}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));



function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

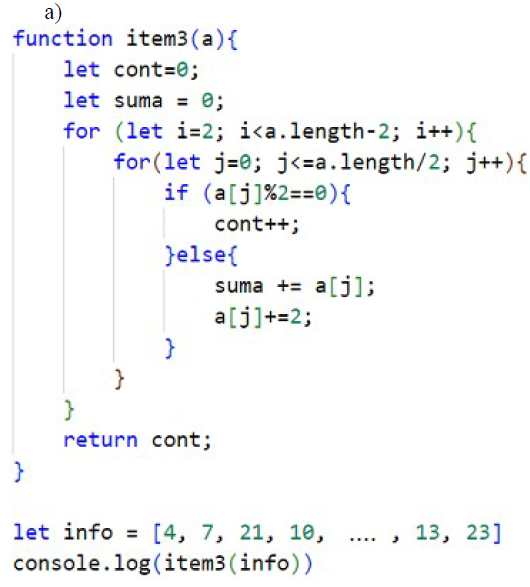
}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));

´l

function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

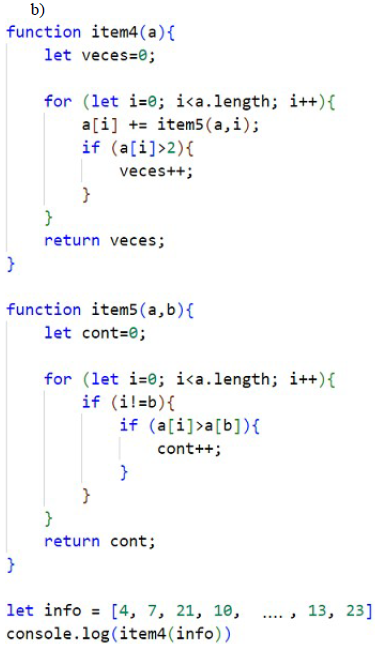
}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));



function item4(a) {

let veces = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

a[i] += item5(a, i); // Llama a item5

if (a[i] > 2) { // 1 instrucción por cada i

veces++; // 1 instrucción si la condición se cumple

}

}

return veces; // 1 instrucción

}

function item5(a, b) {

let cont = 0; // 1 instrucción

for (let i = 0; i < a.length; i++) { // n iteraciones

if (i != b) { // 1 instrucción por cada i

if (a[i] > a[b]) { // 1 instrucción si se cumple la condición

cont++; // 1 instrucción

}

}

}

return cont; // 1 instrucción

}

let info = [4, 7, 21, 10, ..., 13, 23];

console.log(item4(info));