Pregunta 10/0

Antes de comenzar este examen, lee y completa la siguiente declaración jurada. Un examen cuya declaración no tenga la opción "De acuerdo" marcada, no será evaluado.

"Declaro que realicé este examen por mi propia cuenta, sin la ayuda de terceros, y que utilicé únicamente las siguientes ayudas autorizadas para este examen: la bibliografía de la materia 25.03 Algoritmos y Estructuras de Datos y documentos publicados en Internet. Declaro que no compartiré este examen ni sus respuestas a terceros. Comprendo que estoy sujeto al código disciplinario del ITBA, que, en caso de engaño o fraude, contempla sanciones que abarcan desde un llamado de atención hasta la expulsión".

Mostrar opciones de respuesta

- De acuerdo
- En desacuerdo

Pregunta 2

10/10

¿Cómo paralelizarías el algoritmo de ordenamiento mergesort a través del enfoque de computación distribuida? ¿Qué cuestiones particulares deberías atender? Justifica brevemente.

Su respuesta

Con el enfoque de computación distribuida, paralelizaría el algoritmo mergesort dividiendo el conjunto de datos en subconjuntos y asignarle cada subconjunto a un nodo de procesamiento diferente en el sistema distribuido. De esta manera, cada nodo realizaría el ordenamiento en su subconjunto de manera paralela y luego se combinarían esos subconjuntos para devolver la lista ordenada. Hay que tener en cuenta la distribución de los subconjuntos con la cantidad de nodos disponibles, y también el hecho de que habría que establecer mecanismos de sincronización para que se combinen los nodos en el mismo momento.

Pregunta 3

10/10

En el lenguaje SQL, ¿cuál es la diferencia entre un campo marcado `PRIMARY KEY` y uno que no lo está?

Su respuesta

Un campo marcado como Primary Key identifica de manera única cada registro (fila) en la tabla. Solo puede haber un solo campo marcado Primary Key, y este no puede contener valores duplicados ni nulos. Por el otro lado, si un campo no tiene la marca, entonces puede contener valores duplicados o nulos.

Pregunta 4

0/10

En relación a las tablas hash, describe en palabras el algoritmo de rehashing. Justifica brevemente.

Su respuesta

El algoritmo de rehashing se lleva a cabo cuando una función hash devuelve el mismo valor hash para dos elementos distintos. Es una herramienta de resolución de colisiones que consiste en calcular una nueva posición de índice utilizando una función de hash secundaria para devolver dos valores distintos para los diferentes elementos.

Pregunta 5

10/10

En el contexto de la búsqueda con hashing, ¿cuáles son las ventajas y desventajas de utilizar una función hash criptográfica en lugar de una función hash más simple? Justifica brevemente.

Su respuesta

Una ventaja de utilizar una función hash criptográfica es el hecho de que están diseñadas para enfrentar hackeos o ataques maliciosos, lo cual asegura una mayor seguridad en el la búsqueda y almecenamiento de los datos. Por el otro lado, las funciones hash criptográficas pueden tener una mayor complejidad de tiempo en comparación a las funciones de hash más simples, ya que necesitan mayores recursos para obtener valores hash únicos y uniformemente distribuidos.

Pregunta 6

10/10

La función a continuación encuentra el índice del elemento máximo del arreglo `array`. Siendo que dispones de una máquina con múltiples cores, describe en palabras qué modificaciones realizarías sobre la función para mejorar su eficiencia.

```
int getMaxIndex(vector<int> array)
{
    int max = -1;
    int maxIndex = 0;
```

```
    for (int i = 0; i < array.size(); i++)
    {
        if (array[i] > max)
        {
            max = array[i];
            maxIndex = i;
        }
    }

    return maxIndex;
}
```

Su respuesta

Habria que subdividir el arreglo en distintos subconjuntos y ordenar cada subconjunto en un core distinto y luego volverlos a juntar asi quedaria ordenado de una manera mas rapida.

Pregunta 7

10/10

En un arreglo ordenado de números enteros, ¿cómo puedes determinar, en la forma más eficiente, si un número específico se repite más de una vez utilizando búsqueda binaria? Describe el algoritmo en palabras y analiza su complejidad computacional.

Su respuesta

Utilizando búsqueda binaria, se busca la aparición de este número específico. Primero, comparamos el número K con el número del medio del arreglo. Si coincide, entonces se analiza a ambos lados, atrás y adelante a ver si se encuentra ese mismo número. Si no es encontrado en la mitad, entonces lleva a cabo el algoritmo: si K es menor al número del medio, repetimos la búsqueda en la mitad inferior del arreglo. Si K es mayor al número del medio, repetimos la búsqueda en la mitad superior del arreglo. Es un algoritmo recursivo que cuando encuentra el número deseado, se comparan los elementos adyacentes a ver si está repetido. La complejidad computacional de este algoritmo es O(log(n)) y la verificación de los elementos adyacentes tiene una complejidad de O(1).

Pregunta 8

10/10

Explica brevemente el concepto de interfaz en programación, y relaciónalo con la idea de abstracción.

Su respuesta

Una interfaz en programación define un conjunto de métodos o propiedades que una clase debe implementar, sin especificar cómo deben ser implementados. Son los aspectos públicos del programa. Esto permite la abstracción, ya que un código que utiliza una interfaz no necesita conocer detalles internos de la implementación de un objeto.

Pregunta 9

5/10

¿Cuál es la complejidad computacional del recorrido de una tabla hash? Justifica brevemente.

Su respuesta

La complejidad computacional del recorrido de una tabla hash es de O(n), ya que se debe visitar una n cantidad de elementos almacenados para encontrar cada posición en la tabla.

Pregunta 10

3/10

En un sistema de gestión de inventario de una tienda en línea, se desea ordenar los productos en función de su disponibilidad. ¿Qué algoritmo de ordenamiento sería más apropiado en este caso? Justifica brevemente.

Su respuesta

Podria utilizarse selection sort, donde se posicionarían los elementos con más disponibilidad en frente.

Pregunta 11

0/10

Escribe una función en C++ que reciba un objeto de tipo `list<int>` por referencia (con `list` de STL), y devuelva el elemento central en la forma más eficientemente posible. Por ejemplo, si la lista contiene los elementos {4, 9, 3, 1, 6}, debe devolver el elemento 3; y si contiene {4, 9, 3, 1}, el elemento 9. Analiza la complejidad computacional de tu función. Justifica brevemente.

Su respuesta