Protokoll: Beschreibung der rekursiven Funktionen und Aufwandsabschätzung

1. Rekursive Funktionen

a) insert

Aufbau: Die Funktion fügt einen neuen Knoten in den binären Suchbaum ein.

Abbruchbedingung: Wenn der aktuelle Knoten NULL ist, wird ein neuer Knoten mit dem übergebenen Schlüsselwert erstellt und zurückgegeben.

Parameter: struct node *node ist der aktuelle Knoten, in den der neue Knoten eingefügt werden soll, und int key ist der Schlüsselwert des neuen Knotens.

Rückgabewert: Die Funktion gibt den aktualisierten Knoten zurück.

b) deleteNode

Aufbau: Die Funktion löscht einen Knoten mit einem bestimmten Schlüsselwert aus dem binären Suchbaum.

Abbruchbedingung: Wenn der aktuelle Knoten NULL ist, wird der aktuelle Knoten zurückgegeben.

Parameter: struct node *root ist der aktuelle Knoten, in dem der zu löschende Knoten gesucht wird, und int key ist der Schlüsselwert des zu löschenden Knotens.

Rückgabewert: Die Funktion gibt den aktualisierten Knoten zurück.

c) height

Aufbau: Die Funktion berechnet die Höhe des binären Baums.

Abbruchbedingung: Wenn der aktuelle Knoten NULL ist, wird -1 zurückgegeben.

Parameter: struct node *root ist der aktuelle Knoten.

Rückgabewert: Die Funktion gibt die Höhe des Baumes zurück.

d) balanceFactor

Aufbau: Die Funktion berechnet den Balancefaktor für jeden Knoten im Baum.

Abbruchbedingung: Wenn der aktuelle Knoten NULL ist, wird die Funktion beendet.

Parameter: struct node *root ist der aktuelle Knoten.

Rückgabewert: Die Funktion hat keinen Rückgabewert.

e) isAVL

Aufbau: Die Funktion überprüft, ob der gegebene Baum ein AVL-Baum ist.

Abbruchbedingung: Wenn der aktuelle Knoten NULL ist, wird 1 zurückgegeben.

Parameter: struct node *root ist der aktuelle Knoten.

Rückgabewert: Die Funktion gibt 1 zurück, wenn der Baum ein AVL-Baum ist, andernfalls 0.

f) calculateStatistics

Aufbau: Die Funktion berechnet Statistiken wie Min, Max, Summe und Anzahl der Knoten im Baum.

Abbruchbedingung: Wenn der aktuelle Knoten NULL ist, wird die Funktion beendet.

Parameter: struct node *root ist der aktuelle Knoten, und int *min, int *max, int *sum, int *count sind Zeiger auf Variablen, um die Statistiken zu speichern.

Rückgabewert: Die Funktion hat keinen Rückgabewert.

2. Aufwandsabschätzung mittels O-Notation

Die rekursiven Funktionen insert, deleteNode, isAVL, calculateStatistics haben im Durchschnitt eine Laufzeitkomplexität von O(log N) und im schlimmsten Fall von O(N), wobei N die Anzahl der Knoten im Baum ist.

Die Funktionen height und balanceFactor haben eine Laufzeitkomplexität von O(N), da sie den gesamten Baum traversieren müssen, um ihre Aufgabe zu erfüllen.