**ASSESSMENT FORM**

**Course: COMP6048001 - Data Structure**

**Method of Assessment:** **Case Study**

**Semester/Academic Year : 2/2021-2022**

**Name of Student : Stanley saputra**

**NIM : 2501961754**

**Name of Lecturer : Irene Anindaputri Iswanto**

**Date : 30/06/2022**

**Class : LA01**

**Topic : Review II**

**4.** What are the difference between binary tree and b-tree?

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Perbedaan** | **B - Tree** | **Binary Tree** |
| **1** | Jumlah ChildNode | Pada B-Tree , sebuah node dapat memiliki sejumlah n- childnode maksimal. | Pada Binary Tree hanya dapat memiliki maksimal 2 childnode. |
| **2** | Time Complexity | B – tree memiliki time complexity dimana M adalah jumlah node yang dimiliki oleh B-tree | Pada Binary Tree memiliki time complexity dengan N merupakan jumlah node dan 2 merupakan batas maksimal dari childnode. |
| **3** | Dari segi penggunaan | B – Tree biasanya digunakan untuk pengindeksan kode kode pada manajemen system database (DBMS) | Binary Tree pada umumnya digunakan untuk pengoptimalisasian kode. |
| **4** | Penyimpanan | B – Tree digunakan ketika data disimpan pada bentuk hardware seperti harddisk, ssd dan sebagainnya. | Binary Tree digunakan jika data tersimpan pada software seperti RAM. |

b. What are the difference between AVL Tree and Red Black Tree? In what case would you want to use a red black tree over an AVL tree and vice versa?

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Perbedaan** | **AVL - Tree** | **Red Black Tree** |
| **1** | Struktur Pohon | Pada AVL Tree memiliki struktur pohon yang seimbang. Hal ini dapat terjadi karena pada AVL terdapat Balance Factor. | Sedangkan pada Red Black Tree terkadang masih bisa terjadi ketimpangan. Hal ini dapat terjadi karena pada Red Black Tree tidak terdapat Balance Factor |
| **2** | Proses Insertion dan Deletion | Pada AVL Tree proses insertion dan deletion memiliki waktu yang lebih lambat, karena melewati banyak proses sepertu balance, dan rotation. | Pada Red Black Tree memiliki proses insertion dan deletion yang lebih cepat karena operasinya lebih sederhana |
| **3** | Warna Node | AVL tree ridak memiliki warna. | Pada Red Black Tree memiliki warna node merah dan hitam. |
| **4** | Proses Searching | AVL memiliki waktu searching yang lebih cepat karena lebih balance. | Sedangkan pada Red Black Tree memiliki waktu insertion yang lebih lama. |
| **5** | Pengimplementasian dan penggunaannya | AVL Tree lebih sulit untuk diimplementasikan karena lebih kompleks, pada AVL juga memiliki beberapa mekanisme seperti height,balance, rotate dan sebaginya. AVL Tree biasanya digunakan dalam system data base karena memiliki time complexity yang rendah | Pada Red Black Tree lebih mudah untuk diimplementasikan karena lebih sederhana. Pada umumnnya Red Black Tree digunakan pada system yang sederhana seperti system perpustakaan yang cenderung hanya melakukan insertion. |

Jika sata akan membuat suatu database atau suatu system yang membutuhkan searching. Saya akan menggunakan AVL Tree karena memiliki time-complexity yang lebih cepat.

Namun jika saya disuruh untuk membuat system sederhana yang hanya mengandalkan insertion dan deletion, maka saya akan menggunakan Red Black Tree, karena dalam hal ini tentunya Red Black Tree lebih unggul dibandingkan AVL.