Ikram Nurhadi D

222310013

2. b.Binary Tree adalah struktur data yang mirip dengan Linked List. Bila Linked List dianalogikan sebagai rantai yang linier maka Binary Tree dianalogikan sebagai pohon. Binary Tree dikelompokkan menjadi tree yang tidak berurut (unordered Binary Tree) dan tree yang terurut (ordered Binary Tree).

c. Saat membuat program tree pada C++, ada beberapa hal yang perlu diperhatikan

1. Struktur Data: Tentukan struktur data yang sesuai untuk merepresentasikan tree dalam program C++. Umumnya, menggunakan pendekatan pointer dengan menggunakan struktur atau kelas yang memiliki pointer ke anak kiri dan anak kanan.

2. Algoritma Penyisipan: Tentukan algoritma yang tepat untuk menyisipkan elemen baru ke dalam tree. Ini termasuk menentukan posisi yang sesuai untuk menyisipkan elemen dan menangani kondisi khusus seperti pengecekan kesamaan nilai atau pengurutan.

3. Algoritma Pencarian: Tentukan algoritma yang tepat untuk mencari elemen dalam tree. Biasanya digunakan algoritma pencarian biner (binary search) untuk mencari elemen dengan efisien dalam pohon pencarian biner.

4. Algoritma Penghapusan: Tentukan algoritma yang tepat untuk menghapus elemen dari tree. Ini melibatkan menemukan elemen yang akan dihapus, menangani kasus ketika elemen tersebut memiliki anak atau tidak, serta mengatur ulang tree setelah penghapusan.

5. Traversal: Tentukan metode traversal yang sesuai untuk melintasi seluruh simpul dalam tree. Misalnya, traversal inorder, preorder, atau postorder yang memungkinkan Anda mengakses elemen-elemen dalam urutan yang berbeda.

6. Manajemen Memori: Pastikan Anda memperhatikan manajemen memori dengan benar saat menggunakan pendekatan pointer dalam implementasi tree. Selalu dealokasikan memori yang dialokasikan secara dinamis dengan menggunakan operator delete atau pembebasan memori yang sesuai.

7. Validasi dan Penanganan Error: Lakukan validasi data yang masuk dan berikan penanganan yang tepat jika terjadi kesalahan atau kondisi tidak valid saat memanipulasi tree. Ini termasuk pengecekan input yang benar, penanganan kasus khusus, dan pemrosesan yang sesuai saat terjadi kesalahan.

8. Efisiensi: Pertimbangkan efisiensi dalam implementasi program tree. Pilih algoritma yang efisien untuk operasi penyisipan, pencarian, dan penghapusan. Juga, pastikan implementasi tidak membuang-buang memori atau sumber daya komputer yang tidak perlu.