**Jawaban Pertanyaan**

1. Dalam Binary search tree proses pencarian data lebih efektif dibanding binary tree biasa karena Binary Search Tree memiliki sifat bahwa semua left-child harus lebih kecil daripada right child dan parentnya. Jadi data terurut dari data terkecil di left-child hingga data terbesar berada di right child
2. Pada class Node, fungsi atribut left adalah sebagai pointer yang menunjuk ke left-child, sedangkan fungsi dari atribut right adalah sebagai pointer yang menunjuk ke right-child.
3. a. Fungsi atribut root adalah sebagai node awal pada binary tree dan tidak memiliki predessor maupun parent. b. Ketika objek tree pertama kali dibuat, nilai dari root adalah null yang berarti kosong atau tidak ada data.
4. Ketika tree masih kosong atau bernilai null, dan akan ditambahkan sebuah node baru, maka fungsi add akan dijalankan dan node awal yang bernilai null akan diubah dengan data baru.
5. Fungsi kode program tersebut adalah melakukan pengecekan pada data baru yang ditambahkan pada method add, dengan data pada current yaitu data yang terdapat pada root. Jika data yang ditambahkan bernilai kurang dari data root maka dilakukan pengecekan pada child sebelah kiri root (current.left). Setelah itu dilakukan pengecekan, apakah current.left!=null, yaitu apabila child kiri bernilai tidak null maka current akan diubah menjadi current.left, dan proses ini terus berulang hingga proses masuk ke percabangan dari pengecekan current.left!=null, apabila child kiri sudah bernilai null maka current.left akan ditambahkan data baru yang diinputkan pada parameter method add, dan program akan melakukan break atau menghentikan proses perulangan yang sedang berlangsung.
6. Perbedaan antara traverse mode pre-order, in-order, dan post-order adalah pada letak baris program pencetakan data yang dikunjungi. Pada mode pre-order, pencetakan data dilakukan di awal, yaitu sebelum mengunjungi left dan right child. Pada mode in-order, data akan dicetak di tengah,, yaitu setelah mengunjungi left child, tetapi juga sebelum mengunjungi right child. Pada mode post-order data dicetak paling akhir, yaitu setelah mengunjungi left dan rigth child.
7. Menurut saya mengetahui parent dari node yang akan dihapus diperlukan karena dengan mengetahui parentnya kita dapat mengetahui letak node yang akan dihapus. Jika data node yang akan dihapus nilainya lebih kecil dari parent, berarti letak node yang akan dihapus berada di child sebelah kiri parent. Dan begitu pula jika data node yang akan dihapus nilai lebih besar dari parent, berarti letak node yang akan dihapus berada di child sebelah kanan parent.
8. Dalam method delete(), variabel isLeftChild yang bertipe data boolean digunakan untuk mengetahui letak dari node yang akan dihapus. Jika data berada pada child sebelah kiri parent, maka variabel isLeftChild akan bernilai true, dan data berada pada child sebelah kanan parent, maka variabel isLeftChild akan bernilai false.
9. Method getSuccessor digunakan untuk mendapatkan successor dari node yang akan dihapus yang nantinya akan digunakan ketika menghapus node yang memiliki 2 child.
10. Dalam method getSuccessor() di program yang dilakukan pada percobaan cara yang diimplementasikan adalah dengan cara mencari nilai terkecil dari subtree sebelah kanannya, pada statement Node successor = del.right dapat diketahui bahwa successor yang dicari dimulai dari subtree sebelah kanan node yang akan dihapus.
11. Pada class BinaryTreeArray kegunaan dari atribut data adalah sebagai array yang menyimpan data bertipe data integer dan atribut idxLast digunakan untuk menyimpan data indeks terakhir data array yang merupakan leaf paling kanan binary tree dengan tipe data integer.
12. Kegunaan dari method populateData() adalah untuk mengisi atribut data dan idxLast pada class BinaryTreeArray. Method traverseInOrder digunakan untuk mencetak data secara in-order yaitu data dicetak ditengah proses, yaitu setelah mengunjungi left child dan sebelum mengunjungi right child.
13. Suatu node binary tree disimpan dalam array indeks 2, maka posisi left-child berada pada indeks: 2*indeks+1 = 2*2+1 = 5 dan posisi right-child berada pada indeks: 2*indeks+2 = 2*2+2 = 6. Jadi posisi indeks left-child dan right child masing-masing adalah 5 dan 6.