**LAPORAN PRAKTIKUM**

**MATA KULIAH ALGORITMA DAN STRUKTUR DATA**

Dosen Pengampu : Triana Fatmawati, S.T, M.T

**PERTEMUAN - 13 - Tree**

****

**Nama : M. Zidna Billah Faza**

**NIM : 2341760030**

**Prodi : D-IV Sistem Informasi Bisnis**

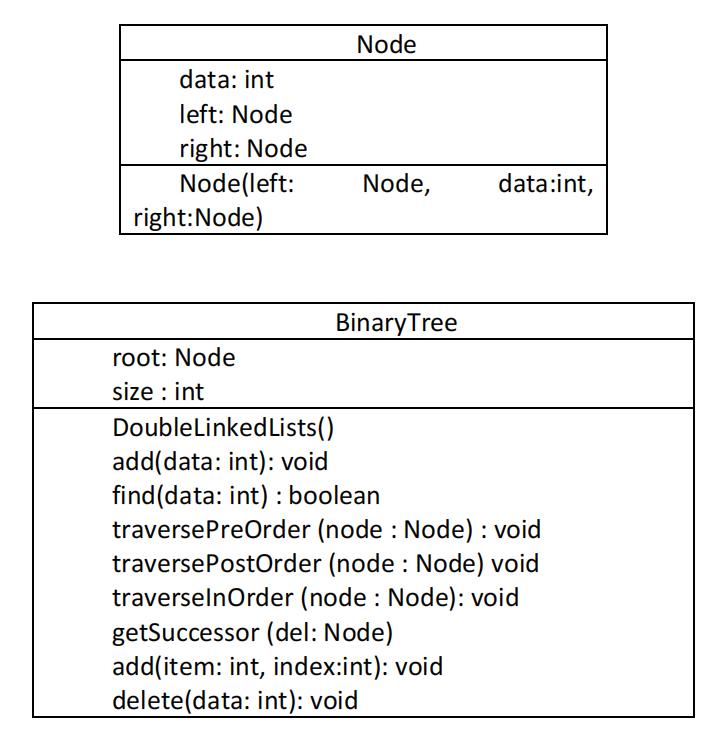
**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**2024**

**Percobaan 1**

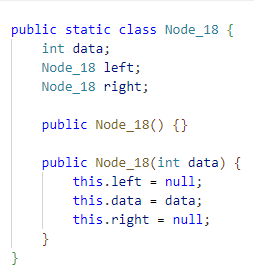
Pada percobaan ini akan diimplementasikan Binary Search Tree dengan operasi dasar, dengan menggunakan array (praktikum 2) dan linked list (praktikum 1). Sebelumnya, akan dibuat class Node, dan Class BinaryTree



1. Buatlah class NodeNoAbsen, BinaryTreeNoAbsen dan BinaryTreeMainNoAbsen



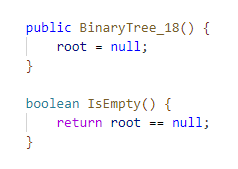
1. Di dalam class Node, tambahkan atribut data, left dan right, serta konstruktor default dan berparameter.



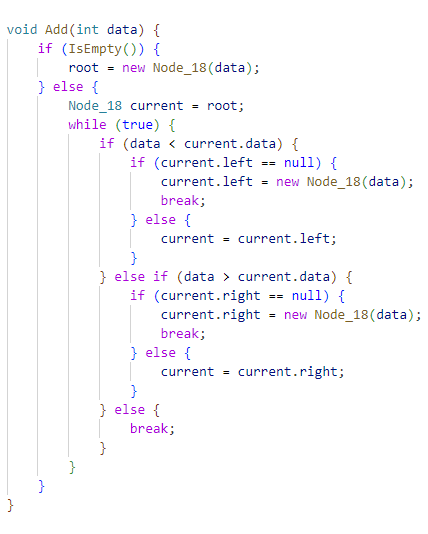
1. Di dalam class BinaryTreeNoAbsen, tambahkan atribut root.



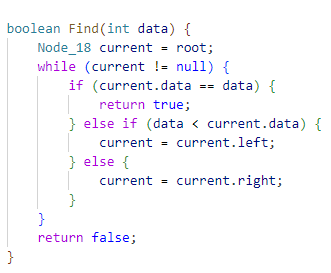
1. Tambahkan konstruktor default dan method isEmpty() di dalam class BinaryTreeNoAbsen



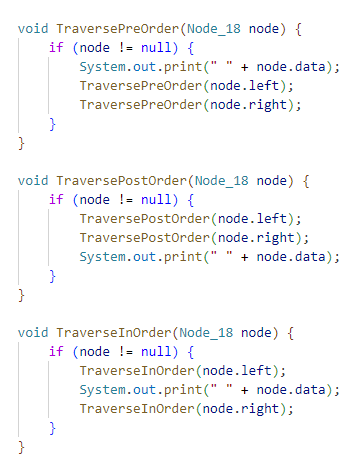
1. Tambahkan method add() di dalam class BinaryTreeNoAbsen. Di bawah ini proses penambahan node tidak dilakukan secara rekursif, agar lebih mudah dilihat alur proses penambahan node dalam tree. Sebenarnya, jika dilakukan dengan proses rekursif, penulisan kode akan lebih efisien.



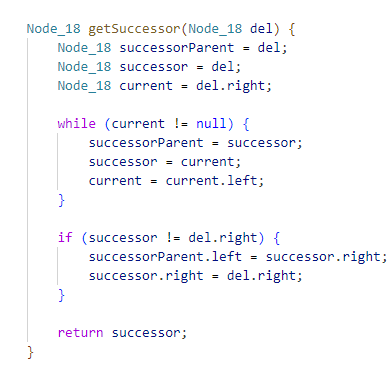
1. Tambahkan method find()



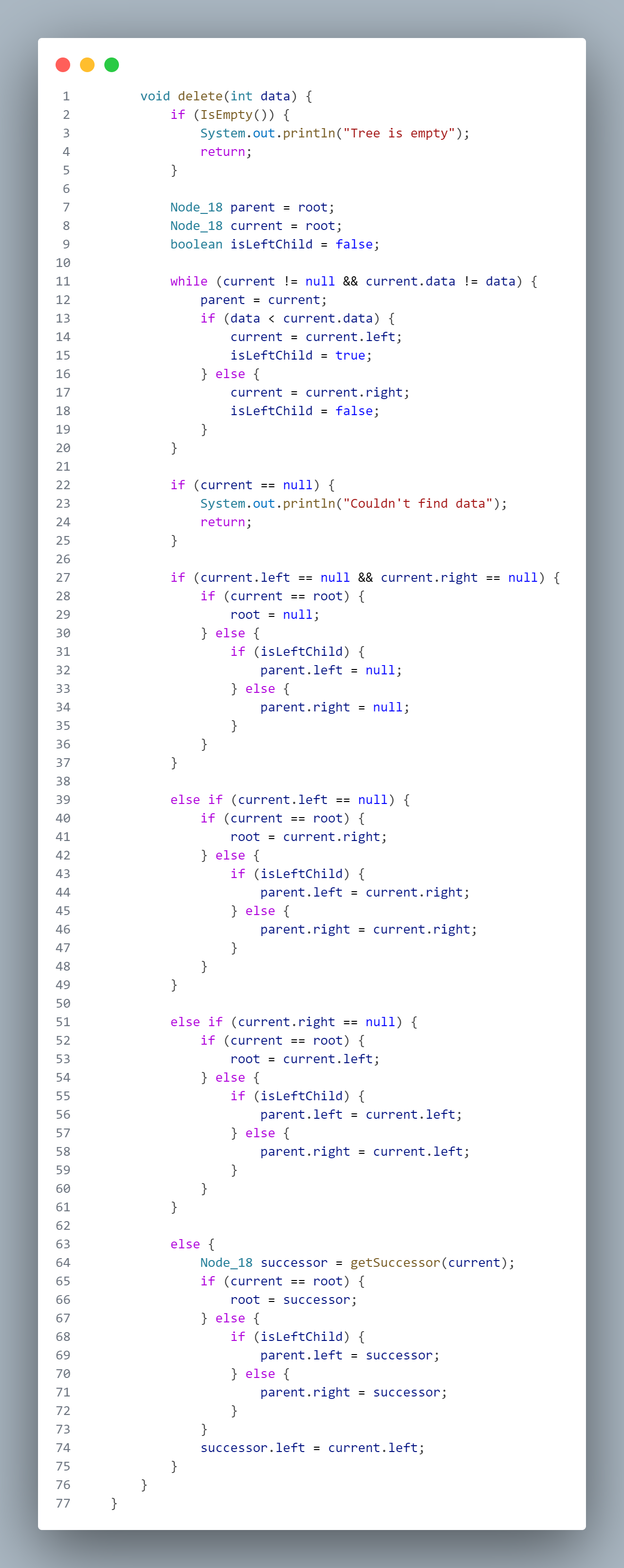
1. Tambahkan method traversePreOrder(), traverseInOrder() dan traversePostOrder(). Method traverse digunakan untuk mengunjungi dan menampilkan node-node dalam tree, baik dalam mode pre-order, in-order maupun post-order.



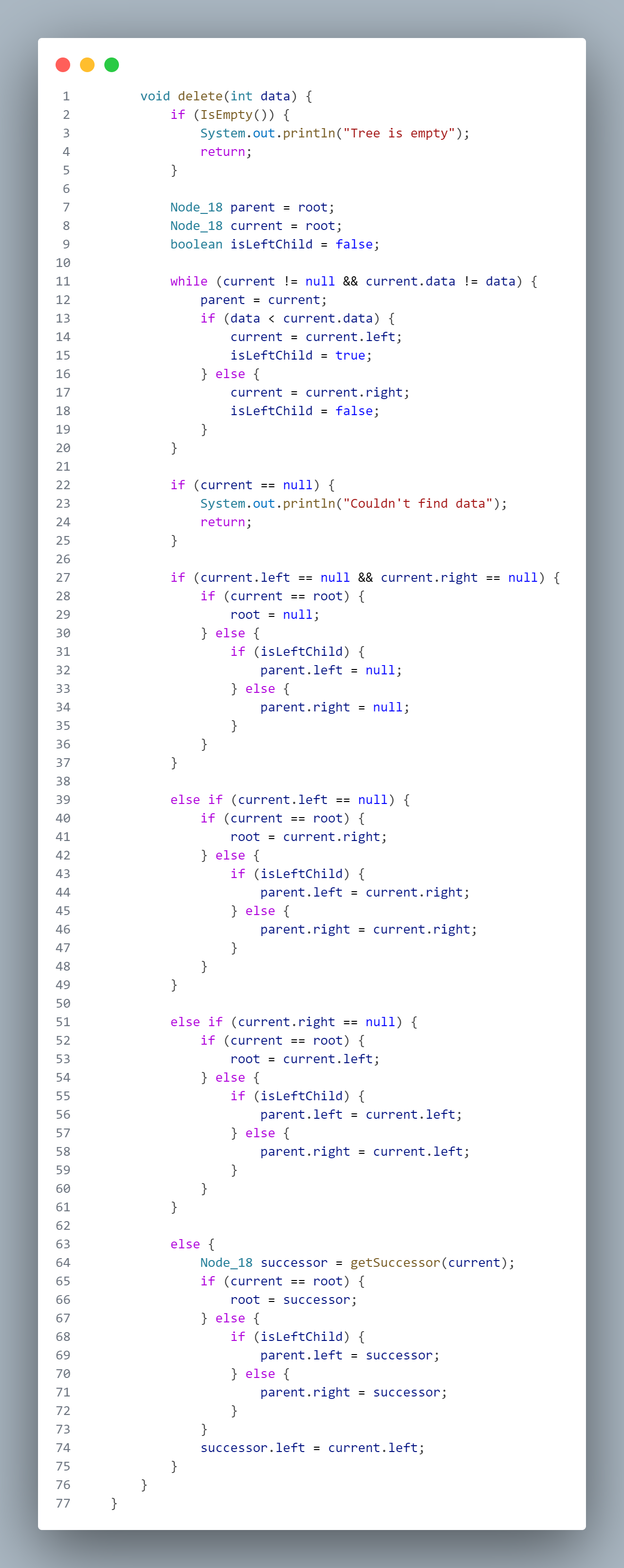
1. Tambahkan method getSuccessor(). Method ini akan digunakan ketika proses penghapusan node yang memiliki 2 child.



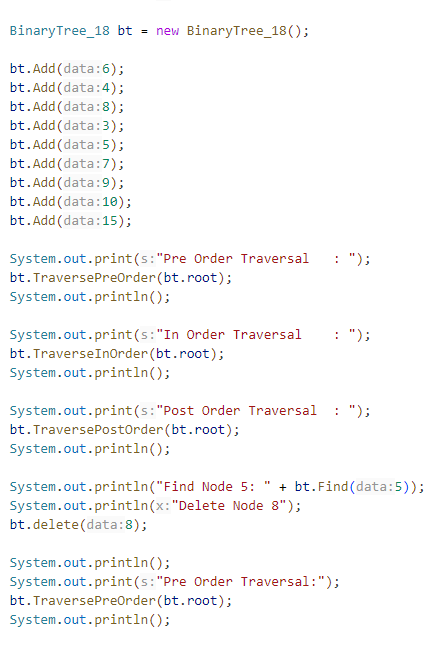
1. Tambahkan method delete(). Di dalam method delete tambahkan pengecekan apakah tree kosong, dan jika tidak cari posisi node yang akan di hapus.



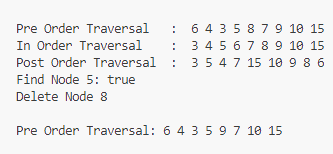
1. Kemudian tambahkan proses penghapusan didalam method delete() terhadap node current yang telah ditemukan.



1. Buka class BinaryTreeMainNoAbsen dan tambahkan method main() kemudian tambahkan kode berikut ini



1. Compile dan jalankan class BinaryTreeMain untuk mendapatkan simulasi jalannya program tree yang telah dibuat.
2. Amati hasil running tersebut.



**Pertanyaan Percobaan 1**

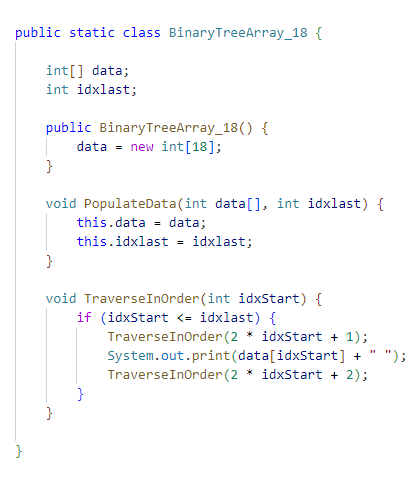
1. Mengapa dalam binary search tree proses pencarian data bisa lebih efektif dilakukan dibanding binary tree biasa?
2. Untuk apakah di class Node, kegunaan dari atribut left dan right?
3. a) Untuk apakah kegunaan dari atribut root di dalam class BinaryTree?

b) Ketika objek tree pertama kali dibuat, apakah nilai dari root?

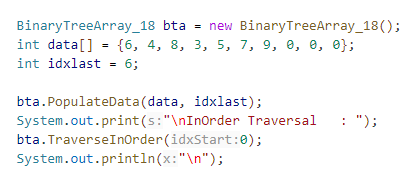
1. Ketika tree masih kosong, dan akan ditambahkan sebuah node baru, proses apa yang akan terjadi?
2. Perhatikan method add(), di dalamnya terdapat baris program seperti di bawah ini. Jelaskan secara detil untuk apa baris program tersebut?

**Percobaan 2**

1. Di dalam percobaan implementasi binary tree dengan array ini, data tree disimpan dalam array dan langsung dimasukan dari method main(), dan selanjutnya akan disimulasikan proses traversal secara inOrder.
2. Buatlah class BinaryTreeArrayNoAbsen dan BinaryTreeArrayMainNoAbsen
3. Buat atribut data dan idxLast di dalam class BinaryTreeArrayNoAbsen. Buat juga method populateData() dan traverseInOrder().



1. Kemudian dalam class BinaryTreeArrayMainNoAbsen buat method main() dan tambahkan kode seperti gambar berikut ini di dalam method Main



1. Jalankan class BinaryTreeArrayMain dan amati hasilnya!



**Pertanyaan Percobaan 2**

1. Apakah kegunaan dari atribut data dan idxLast yang ada di class BinaryTreeArray?
2. Apakah kegunaan dari method populateData()?
3. Apakah kegunaan dari method traverseInOrder()?
4. Jika suatu node binary tree disimpan dalam array indeks 2, maka di indeks berapakah posisi left child dan rigth child masin-masing?
5. Apa kegunaan statement int idxLast = 6 pada praktikum 2 percobaan nomor 4?

**Tugas Praktikum 1**

1. Buat method di dalam class BinaryTree yang akan menambahkan node dengan cara rekursif.
2. Buat method di dalam class BinaryTree untuk menampilkan nilai paling kecil dan yang paling besar yang ada di dalam tree.
3. Buat method di dalam class BinaryTree untuk menampilkan data yang ada di leaf
4. Buat method di dalam class BinaryTree untuk menampilkan berapa jumlah leaf yang ada di dalam tree.
5. Modifikasi class BinaryTreeArray, dan tambahkan

* method add(int data) untuk memasukan data ke dalam tree
* method traversePreOrder() dan traversePostOrder()

Repository : <https://github.com/zidnafaz/Praktikum-Algoritma-Struktur-Data>