**LAPORAN PRAKTIKUM**

**STRUKTUR DATA DAN ALGORITMA**

**Judul: Tree**



**DISUSUN OLEH**

**Ilham Nur Romdoni M0520038**

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

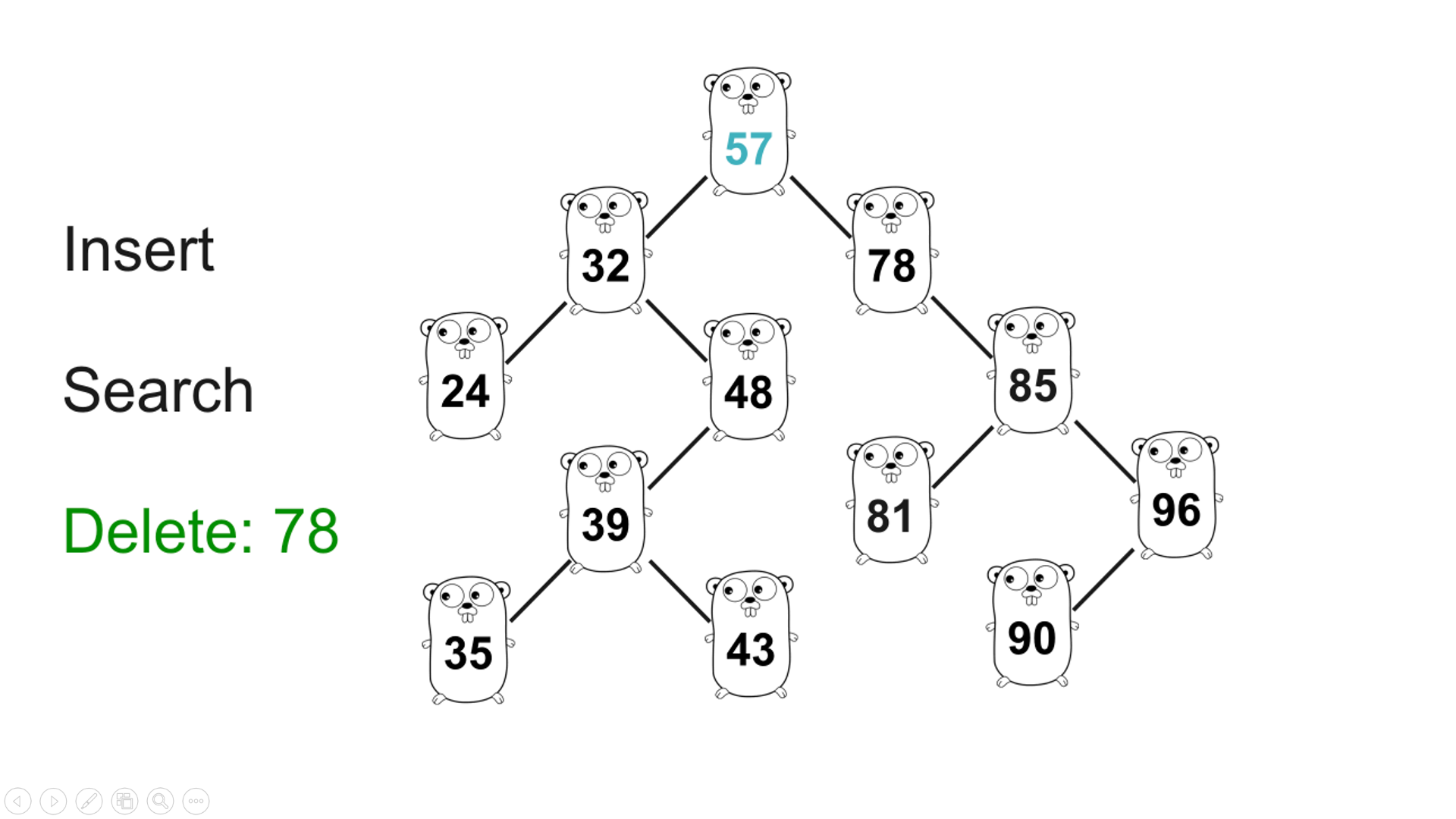
**UNIVERSITAS SEBELAS MARET**

**2021**

**Program Binary Tree dengan Tree Traversal Postorder**

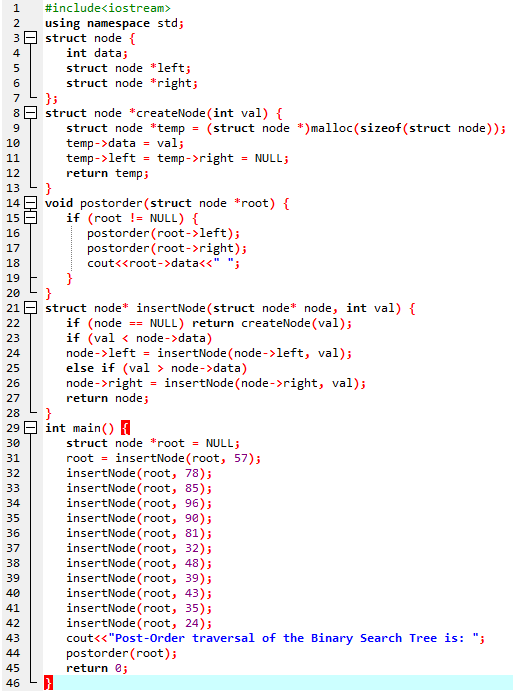
*Tree traversal* adalah bentuk grafik *traversal*. Ini melakukam pengecekan atau pencetakan setiap *node* di *tree*. *Traversal* *postorder* dari *binary* *search* *tree* melakukan kunjungan terhadap masing-masing *node* di *tree* dalam urutan (*Left*, *Right*, *Root*).

*Traversal* *postorder* dari *binary* *tree* adalah sebagai berikut.

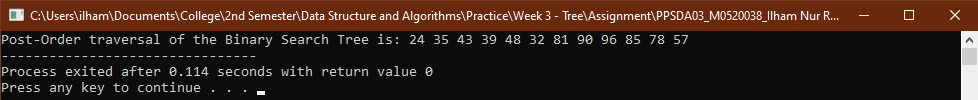


*Postorder* *Traversal* adalah: 24 35 43 39 48 32 81 90 96 85 78 57

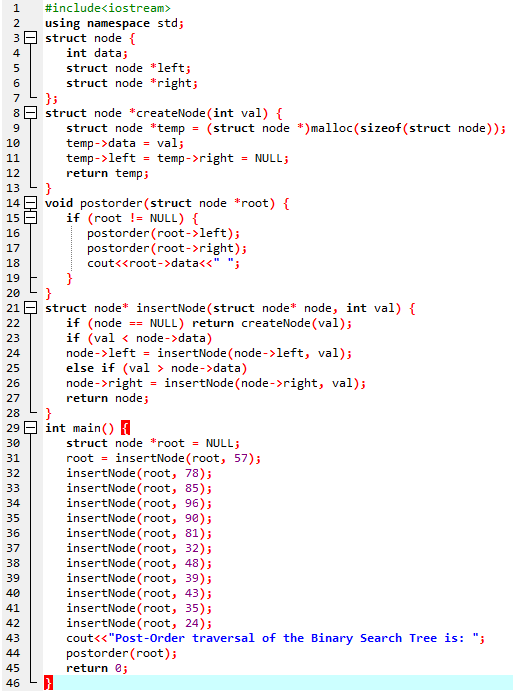
Program untuk melakukan *postorder* *recursive* *traversal* diberikan sebagai berikut.



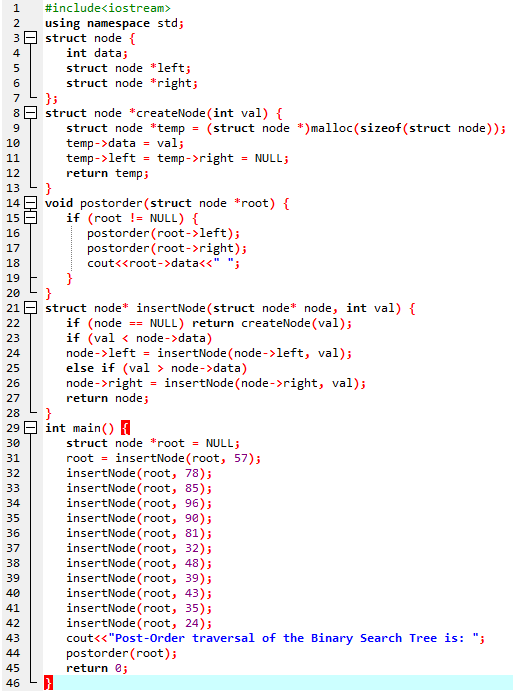
**Output**

****

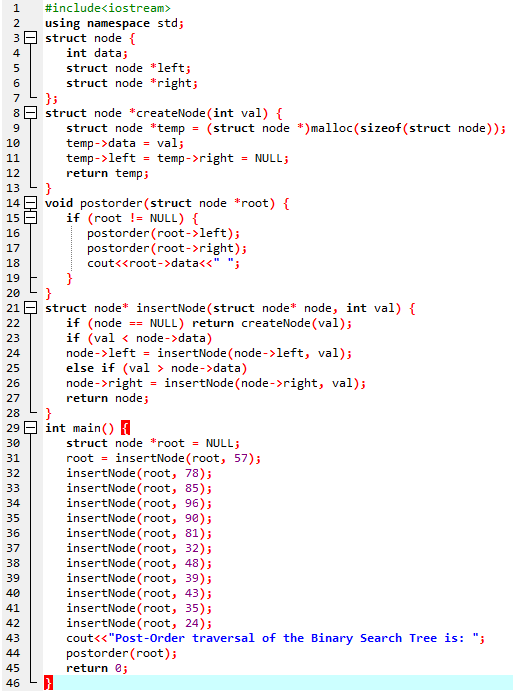
Dalam program di atas, struktur node membuat node dari sebuah *tree*. Struktur ini adalah *self* *referential* *structure* karena berisi *pointers* dari tipe **struct** node. Struktur ini ditunjukkan sebagai berikut.



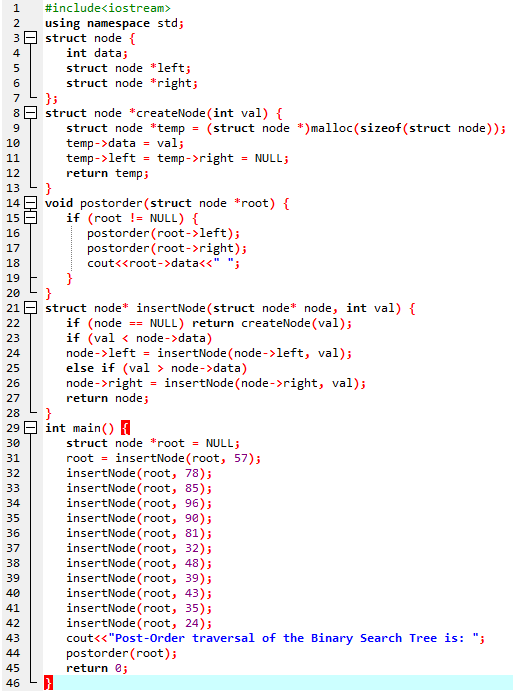
Fungsi createNode() membuat temp node dan mengalokasikan memori menggunakan malloc. Nilai data val disimpan dalam data temp. NULL disimpan di *pointers* temp left dan right. Ini ditunjukkan dengan cuplikan kode berikut.



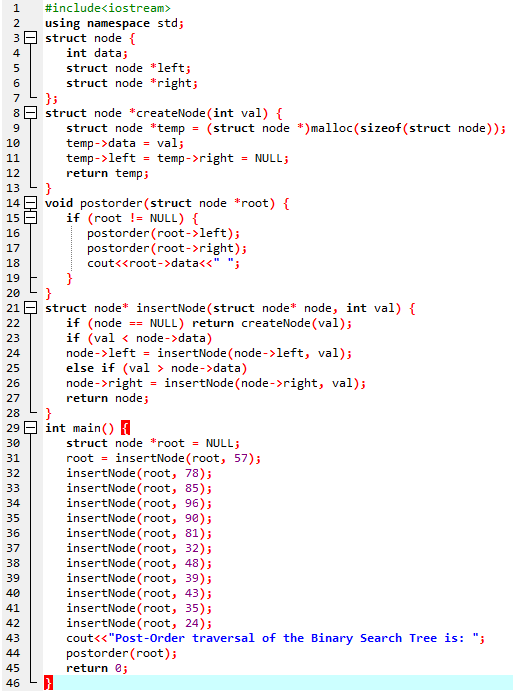
Fungsi postorder() mengambil root dari *binary tree* sebagai argumen dan mencetak elemen *tree* dalam postorder. Ini adalah fungsi *recursive*. Ini ditunjukkan menggunakan kode berikut.



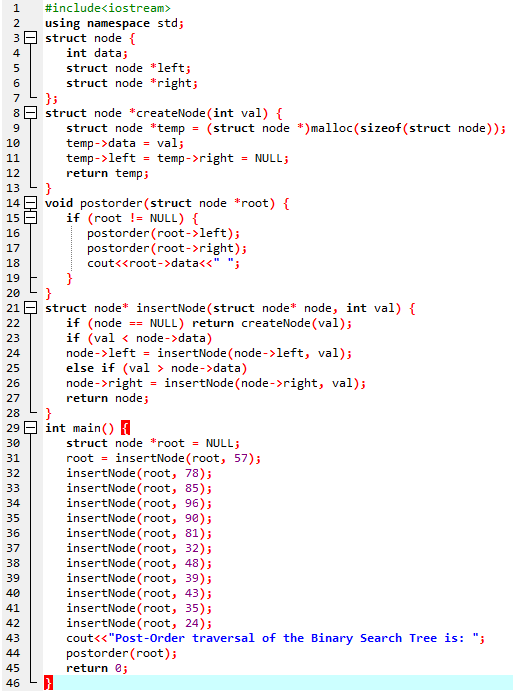
Fungsi insertNode() menyisipkan nilai yang diperlukan dalam *binary tree* pada posisi yang benar. Jika node NULL, maka createNode dipanggil. Jika tidak, posisi yang benar untuk node ditemukan di *tree*. Ini dapat diamati dalam cuplikan kode berikut.



Dalam fungsi main(), node root pertama kali didefinisikan sebagai NULL. Kemudian semua node dengan nilai yang diperlukan dimasukkan ke dalam *binary search tree*. Hal ini ditunjukkan di bawah ini.



Akhirnya, fungsi postorder() dipanggil menggunakan node root dari *tree* dan semua nilai *tree* ditampilkan di postorder. Ini diberikan di bawah ini.



**Referensi**

Atqia, Nuha Lina.2021.“[Week 3 Tree (google.com)](https://classroom.google.com/u/1/c/MzExMjIzMTIxODA4/m/MzE1MjQ4ODUxODQ0/details)”, diakses pada 13 April 2021.

Yadav, Chandu.2018.“[C++ Program to Perform Postorder Recursive Traversal of a Given Binary Tree (tutorialspoint.com)](https://www.tutorialspoint.com/cplusplus-program-to-perform-postorder-recursive-traversal-of-a-given-binary-tree#:~:text=C%2B%2BProgrammingServer%20Side%20Programming%20Tree%20traversal%20is%20a%20form,the%20tree%20in%20the%20order%20%28Left%2C%20Right%2C%20Root%29.)”, diakses pada19 April 2021.