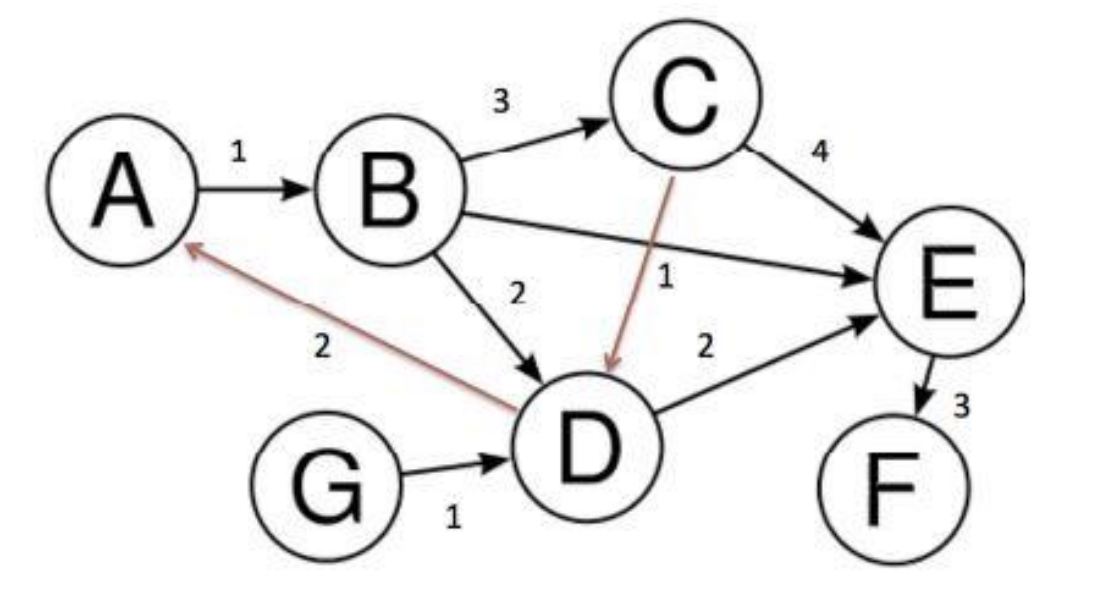
**Praktikum Struktur Data Modul 6**

1. Graph adalah Struktur data yang terdiri dari sekumpulan atau sekelompok simpul atau node yang yang saling berhubungan dan dihubungkan dengan garis. Representasi dalam graph, simpul atau node disebut dengan vertex dan garis yang menghubungkan disebut dengan edge.

Perbedaan Tree dan Graph

|  |  |  |
| --- | --- | --- |
|  | Tree | Graph |
| Alur pada node | Hanya sekali tidak bisa kembali ke node sebelumnya | Bisa bolak – balik antar node |
| Akar / Root | Hanya memiliki satu akar/root | Tidak memiliki akar/root |
| Model | Hirarki atau berurutan dari atas sampai bawah dan tidak bisa kembali lagi ke atas | Bebas, satu node bisa dilewati lebih dari satu kali |
| Kompleksitas | Tidak rumit karena hanya memiliki satu alur | Cukup rumit dan kompleks |
| Traversal | Pre-order, In-order, dan Post-order | DFS (Depth First Search) dan BFS (Breadth First Search) |
| Jumlah garis (edge) | n-1 (dimana n adalah jumlah simpul atau node) | Tidak terdefinisikan karena satu node bisa memiliki lebih dari satu garis dan satu garis bisa dilewati kembali |

1. Graph



1. Notasi
   * G = {V, E}
   * V = {A, B, C, D, E, F, G}
   * E = {(A, B), (B, C), (B, D), (B, E), (C, D), (C, E), (D, A), (D, E), (E, F), (G, D)}
2. Jumlah in-degree, out-degree, dan edge
   * in-degree :
     1. in-degree(A) = 1
     2. in-degree(B) = 1
     3. in-degree(C) = 1
     4. in-degree(D) = 3
     5. in-degree(E) = 3
     6. in-degree(F) = 1
     7. in-degree(G) = 0
   * out-degree :
     1. out-degree(A) = 1
     2. out-degree(B) = 3
     3. out-degree(C) = 2
     4. out-degree(D) = 2
     5. out-degree(E) = 1
     6. out-degree(F) = 0
     7. out-degree(G) = 1
   * edge = 10
3. Adjacency Matriks

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G |
| A | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| B | 0 | 0 | 3 | 2 | 1 | 0 | 0 |
| C | 0 | 0 | 0 | 4 | 1 | 0 | 0 |
| D | 2 | 0 | 0 | 0 | 2 | 0 | 0 |
| E | 0 | 0 | 0 | 0 | 0 | 3 | 0 |
| F | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 0 | 0 | 1 | 0 | 0 | 0 |

1. Weight tiap node
   * Weight = 1
     1. A → B
     2. B → E
     3. C → D
     4. G → D
   * Weight = 2
     1. B → D
     2. D → A
     3. D → E
   * Weight = 3
     1. B → C
     2. E → F
   * Weight = 4
     1. C → E
2. Program