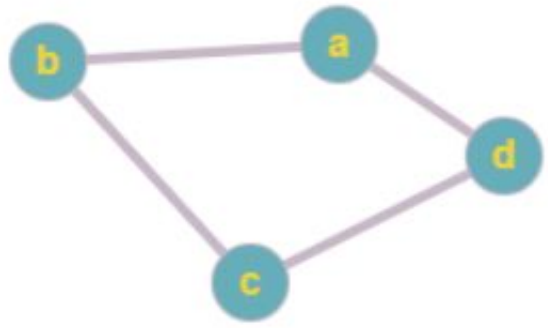
8_3-GraphTraversal

1 Graph Traversal

Graph Traversal merupakan proses mengunjungi setiap *vertex/node* . *Graph Traversal* digunakan untuk mencari jalur dalam suatu *graph* dari titik asal ke titik tujuan, mencari jalur terpendek antara dua *node/vertex*, menemukan semua jalur yang bisa dilalui dari titik asal ke titik tujuan.

Berikut adalah contoh graph untuk proses *graph traversal*.



Gambar 1. ***Graph***

Struktur data *graph* tersebut dengan menggunakan dictionary adalah sebagai berikut: graph = {'a': ['b', 'd'],'b': ['a', 'd'], 'c': ['d'], 'd': ['a', 'b', 'c']}

Dictionary tersebut berisi semua node/simpul serta list tetangganya, dimana setiap key suatu simpul akan berkorespondensi dengan semua tetangganya yang terhubung dengan simpul key tersebut. Berikut ini adalah implementasi mencari jalur dari simpul asal ke simpul tujuan, jika ada jalur maka nilai yang dikembalikan adalah list path jika tidak ada jalur maka nilai yang dikembalikan none. Algoritma yang digunakan adalah *backtracking*, yaitu mencoba setiap kemungkinan secara bergantian sampai menemukan solusi.

1.1 Code

Berikut adalah implementasi *graph traversal*

```
In [6]: graph = {'a': ['b', 'd'],'b': ['a', 'd'], 'c': ['d'], 'd': ['a', 'b', 'c']}
        def find_path(graph, start, end, path=[]):
            path = path + [start]
            for node in graph[start]:
                if not node in path:
                    newpath = find_path(graph, node, end, path)
```

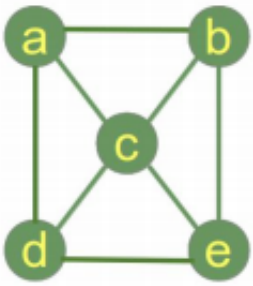
```
                if newpath:
                    return newpath
        if start == end:
            return path
        if not start in graph:
            return None
        return None

print(find_path(graph,'a','d'))
```

```
['a', 'b', 'd']
```

1.2 Latihan - 2

Buatlah program untuk mendapatkan path untuk jalur terpendek dan terpanjang dari sebuah graph. (jalur terpendek / terpanjang HARUS lebih dari 1 dan ditampilkan semua. Semisalkan ada graph seperti berikut ini



Gambar 2. *Graph*