

# BFS DFS

## Graph

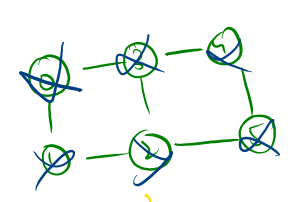
```
void dfs(ArrayList<ArrayList<Integer>> graph, boolean[] path, ArrayList<Integer> psf, int src) {
    //mark
    path[src] = true;
    psf.add(src);
    int totalVtx = graph.size()-1;
    if(psf.size() == totalVtx){
        //i have detected an hamiltonian path
        System.out.println(psf);
    }

    int start = psf.get(0);
    for(int nbr: graph.get(src)){
        if(nbr == start){
            System.out.println(" Cycle Found ");
            break;
        }
    }

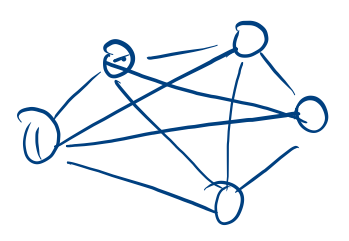
    path[src] = false;
    psf.remove(psf.size()-1);
    System.out.println();
    return;
}

for(int nbr: graph.get(src)){
    if(!path[nbr]) dfs(graph, path, nbr, psf);
}

path[src] = false;
psf.remove(psf.size()-1);
}
```



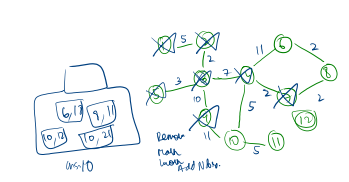
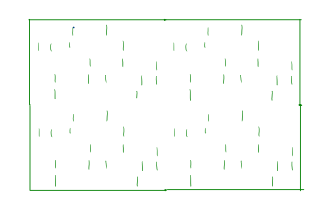
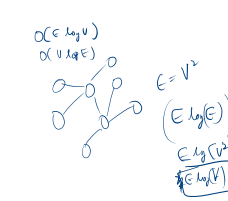
20, 23  
20, 3, 4, 5, 2, 1



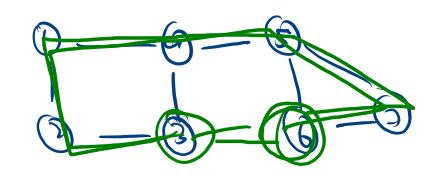
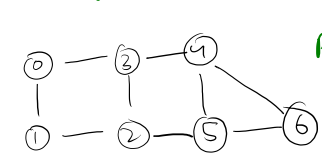
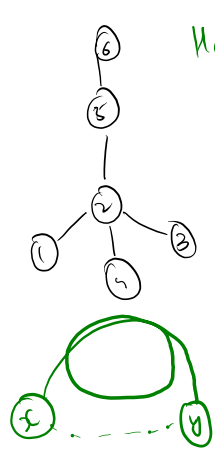
|   | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| 0 | 0 |   |   |   |   |   |
| 1 | 0 | 0 |   |   |   |   |
| 2 | 0 | 0 | 0 |   |   |   |
| 3 |   |   |   | 0 |   |   |

$10^1 \approx O(1)$  a few elements  
 $10^2 \approx O(n)$  a small array  
 $10^5 \approx O(n \log n)$  a sorting algo  
 $10^6 \approx O(n^2)$  a hashtable/array, 2D array  
 $10^7 \approx O(n^3)$  a brute force  
 $10^8 \approx O(n^4)$  a naive algo

$O(V+E)$   
 $O(V)$   
 $O(E)$



## Hamiltonian Path Hamiltonian Cycle



Path  $\rightarrow$  DFS

Algorithm  
 dfs(graph, path, src, psf) {  
 // Mark  
 path[src] = true;  
 psf.add(src);  
 // Work  
 if(psf.size() == graph.size()) {  
 // Hamiltonian  
 }  
 // Call dfs on unvisited nbr  
 for(int nbr: graph.get(src)) {  
 if(!path[nbr]) dfs(nbr, psf);  
 }  
 // Mark false  
 path[src] = false;  
 psf.remove(psf.size()-1);  
 }

