



# Graph Theory 2

...

Graph traversal :  
BFS and DFS

# Graph traversal

What is it ?

-> A method to explore a graph from a given node (every node accessible)

Purpose ?

- Quickly explore the graph (useful for some problems).
- Foundation for more advanced algorithms (we'll see this in the following weeks).



# Big notions used



- Visited/unvisited:

To explore the graph optimally (a node is explored only once) we need to keep track of which node has been already visited

- Main data structure:

The data structure used to keep track on which node will be visited next.

- Additional data structure:

May be used to keep track of various informations depending on the context

# Two algorithm for the price of one:

BFS

Breadth-first search

-

Explore first the nearest nodes from the  
current position

FIFO - Queue

DFS

Depth-first search

-

Go as deep as you can and backtrack when  
you're in a dead end

LIFO - Stack

# A small demonstration

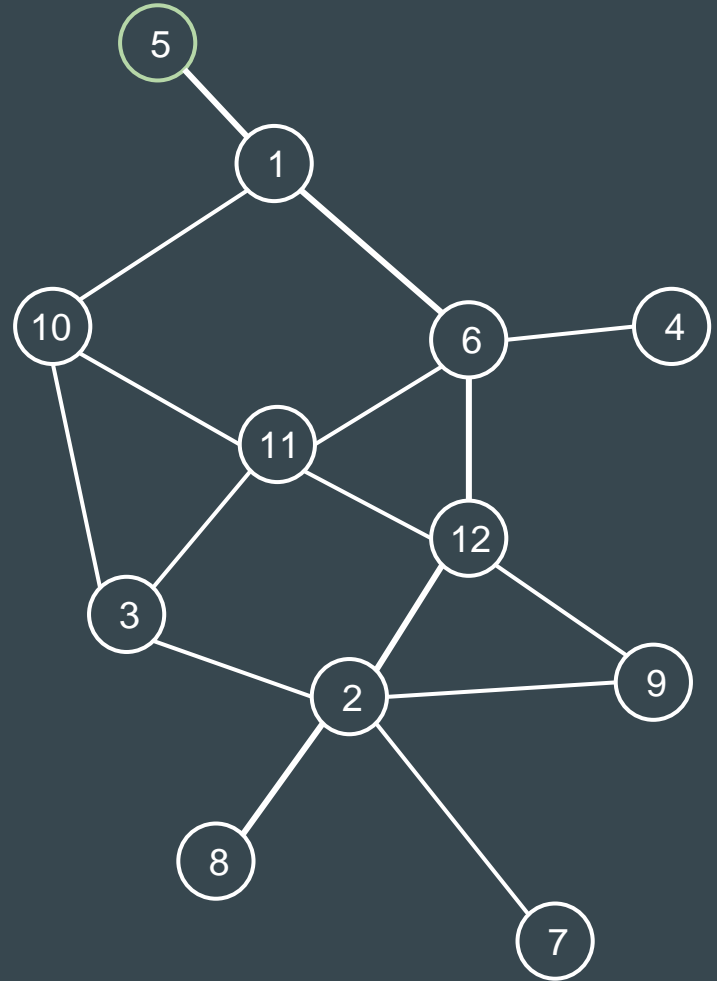
BFS

DFS

# BFS

```
BFS(G, start):  
    let q be a queue  
    add start to the queue  
    set every node of G to unvisited  
    set start to visited  
  
    while q is not empty:  
        current is the next node from q  
  
        for every node  $a_i$  adjacent to current:  
            if  $a_i$  is visited: skip  
            set  $a_i$  to visited  
            put  $a_i$  in q
```

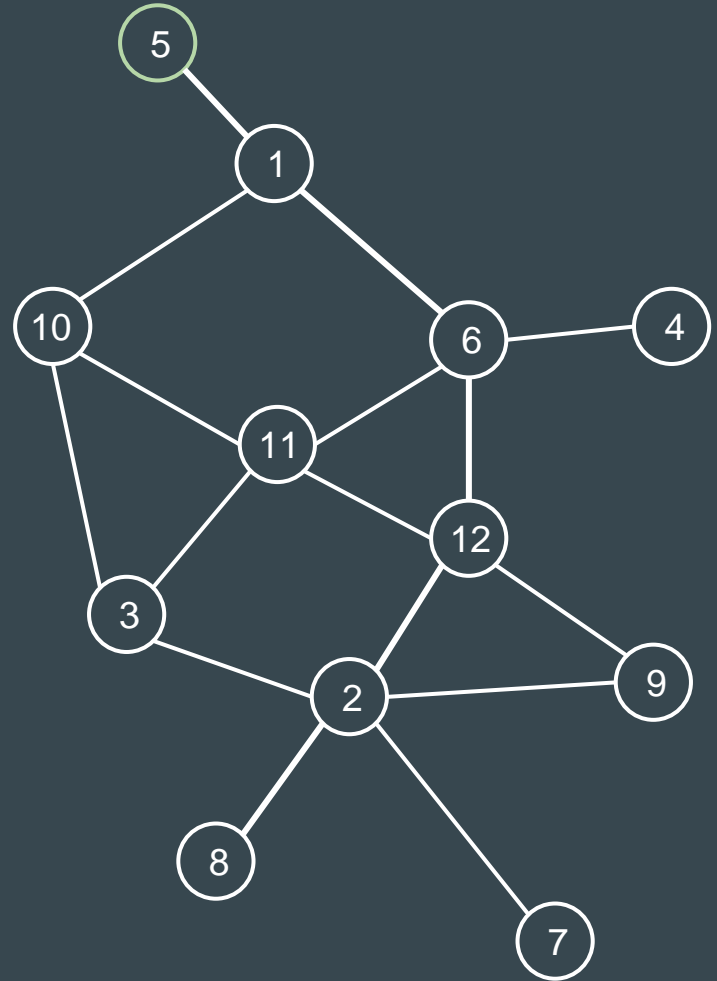
Complexity :  $O(E+V)$



# DFS

```
DFS(G, start):  
    let s be a stack  
    add start to the stack  
    set every node of G to unvisited  
    set start to visited  
  
    while q is not empty:  
        current is the next node from s  
  
        for every node  $a_i$  adjacent to current:  
            if  $a_i$  is visited: skip  
            set  $a_i$  to visited  
            put  $a_i$  in s
```

Complexity :  $O(E+V)$



# Why two variants ?

## BFS

- Find the shortest path between two nodes when the edges' weights are equals

## DFS

- Possibility to implement a recursive version
- Give the topological order of the node in an oriented graph (which can be useful for some shortest path application)



# Credits

More resources:

- DFS : [https://en.wikipedia.org/wiki/Depth-first\\_search](https://en.wikipedia.org/wiki/Depth-first_search)
- BFS : [https://en.wikipedia.org/wiki/Breadth-first\\_search](https://en.wikipedia.org/wiki/Breadth-first_search)
- Cours de 3IF : [https://moodle.insa-lyon.fr/pluginfile.php/317641/mod\\_resource/content/1/cours.pdf](https://moodle.insa-lyon.fr/pluginfile.php/317641/mod_resource/content/1/cours.pdf)

Slides: Sebastien Goll for INSAIgo