Graph algorithms

- **Adjacency**: A vertex $X$ is adjacent to a vertex $Y$ if and only if $(x, y) \in E$

- $E = (u, v, w)$, $u, v, w \in V$ is the weight of an edge in a weighted graph $G$

- **Directed vs. undirected graphs**

- $E$: set of edges

- $V$: set of vertices

- $G = (V, E)$
A path is a sequence of vertices \( V \rightarrow V \rightarrow \ldots \rightarrow V \) such that \( (V_i, V_{i+1}) \in E \) for all \( 1 \leq i < \text{length} - 1 \).

A simple path is a path where all vertices are distinct, except the first and the last may be the same.

Cycle:

\[ \text{cycle} \]

\[ \text{degree} = \text{outdegree} + \text{indegree} \]

\[ \text{in degree} = 2 \text{ } \text{out degree} = 3 \]
Represent a graph:

- A directed graph (digraph) is a graph in which the edges have a direction.
- A directed acyclic graph (DAG) is a directed graph with no directed cycles.

Formally:

1. **Nodes**: Represents entities or elements within the graph.
2. **Edges**: Directed edges indicate a one-way relationship between nodes.
3. **Cycle**: A cycle in a graph is a sequence of nodes and edges that starts and ends at the same node.
4. **Acyclic**: A graph is acyclic if it does not contain any cycles.

Mathematically:

- **Set of vertices**: \( V \)
- **Set of edges**: \( E \)
For all nodes \( v \) in the graph, for all adjacent nodes \( w \):

\[ |E| = 2 \times (\deg v) \]
If stuck, go back and repeat the process.

- Try all nodes going as deep as possible.

DFS: Start with a node

BFS: Breadth first search

DFS: Depth-first search

(Directed graphs)

\[|V| = \text{in degree} + \text{out degree} = |E|
\]

\[|E| = \text{in degree} = |E|
\]

\[|E| = \text{out degree} = |E|
\]
\text{dfs}(v): \\
\text{dfs}(v_3) = v \\
\text{dfs}(v_1) = v_2 \lor v_4 \lor v_5 \\
\text{dfs}(v_4) = v_5 \\
\text{dfs}(v_5) = \{ v, v_4 \} \\
\text{dfs}(v_2) = \{ v \} \\
\text{dfs}(v) = \{ v \} \\
\text{dfs}(v) = \{ v \} \\
\text{dfs}(v) = \{ v \} \\
\text{dfs}(v) = \{ v \} \\
\text{dfs}(v) = \{ v \}
In a DAG, if there is a path from \( v \) to \( v' \), then \( v \) appears before \( v' \) in the ordering.

Definition: A topological sort of a DAG is an ordering of its vertices such that if there is a path from \( v \) to \( v' \), then \( v \) appears before \( v' \) in the ordering.

Topological Sort

BFS: \( \text{dfs}(v) \)
If m. indegree = 0 then

lemma Snoo

for each u in ν. Add( ) \ ν \ x \ G

T. push back ( )
ν \ ν \ G. pop-back ( )

while ν, empty ( ) do

G. push-back ( )

if ν, indegree = 0 then

ν, indegree = Compute-Indegree ( ) \ 
ν, Adjacency Table

∀ for each u ∈ V, in G

O(\(|V|+\(|E|\))

Top-sort ( G, Graph )
```
if (T.size() == 1) {
  // push-back()
}
```