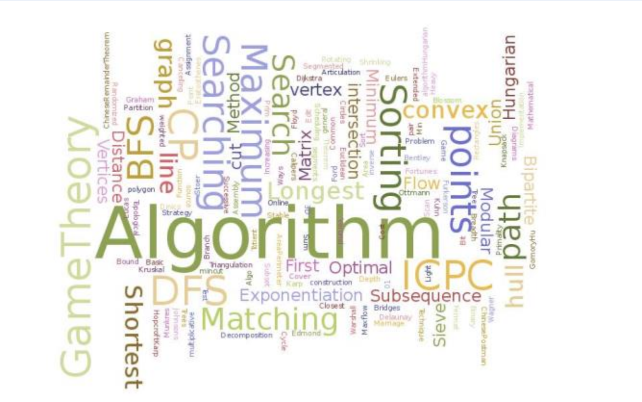
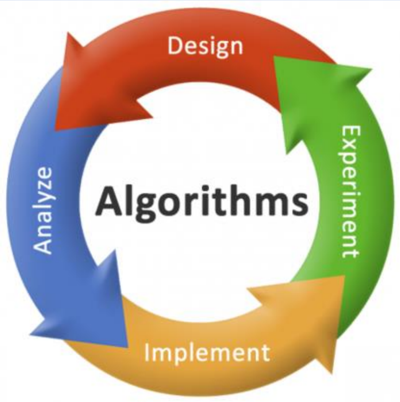
# Graph Algorithms Introduction

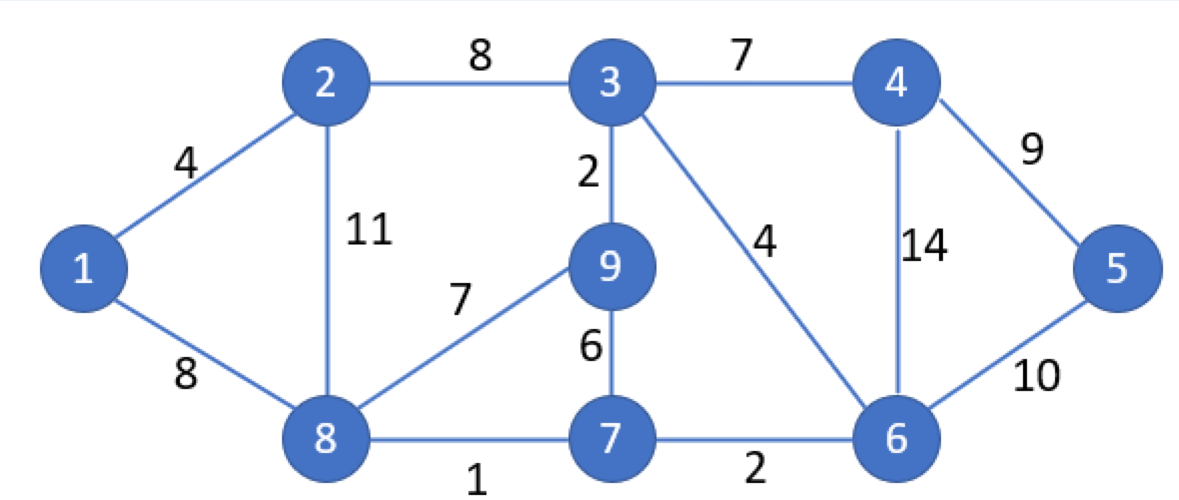
COP 3503
Fall 2021
Department of Computer Science
University of Central Florida
Dr. Steinberg



# Introduction

- Graphs are fundamental in the field of computer science
- There are a variety of problems that involves the use of graphs.
- Before we dive into the problems we need to understand some terminology and notations.

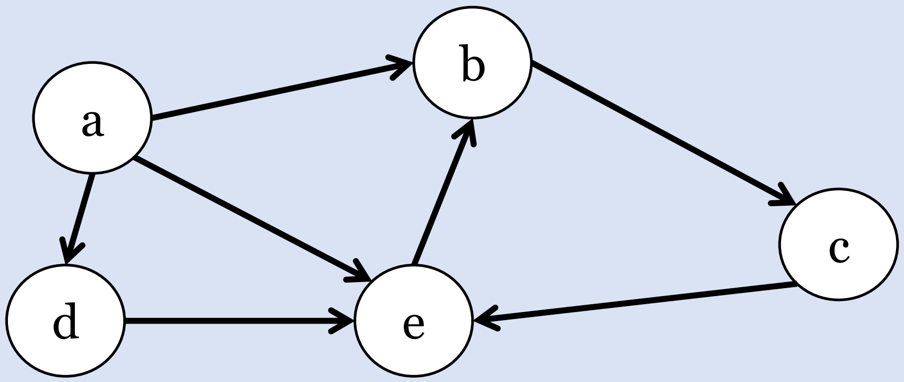


# Graph Representation (Some Notation)

- $G = (V, E)$
- $V$ represents a set of vertices in graph $G$
  - In the set contains a symbol denoting the vertex of a graph
- $E$ represents a set of edges $G$
  - In the set contains a tuple (a,b) denoting two vertices that are connected in the graph
- $|E|$ represents the number of edges in $G$
- $|V|$ represents the number of vertices in $G$

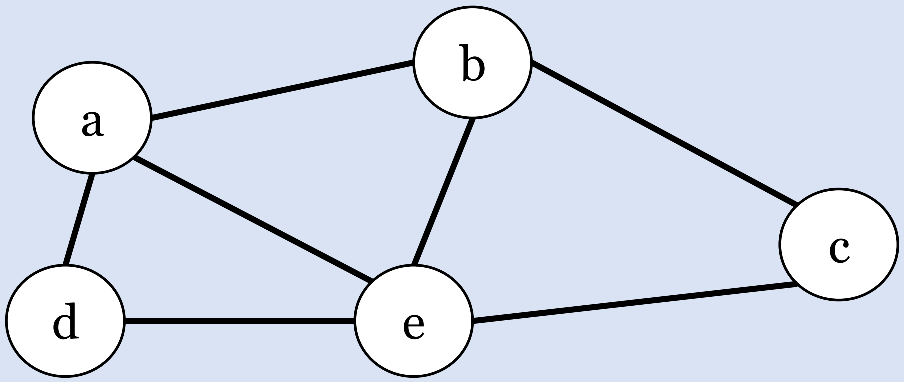
# Directed Graphs

- A graph is considered directed when the edges in $E$ only go in one direction.
- In figures, this is usually denoted by using arrows when connecting two vertices.



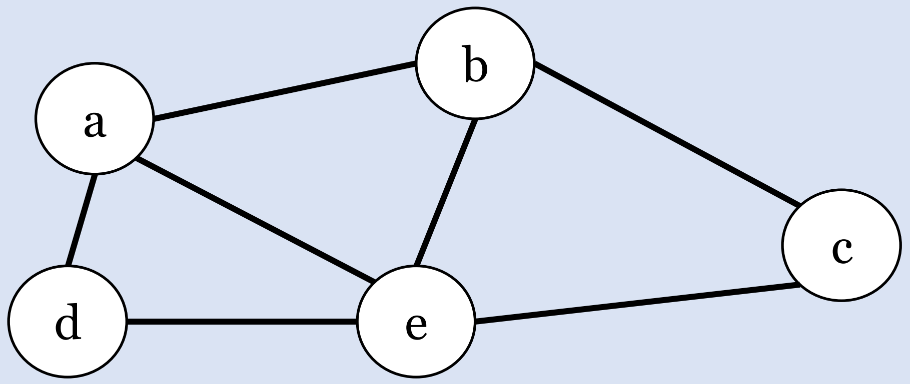
# Undirected Graphs

- A graph is considered undirected when the edges in $E$ only go in both direction.
- In figures, this is usually denoted by using <u>solid lines</u> when connecting two vertices.



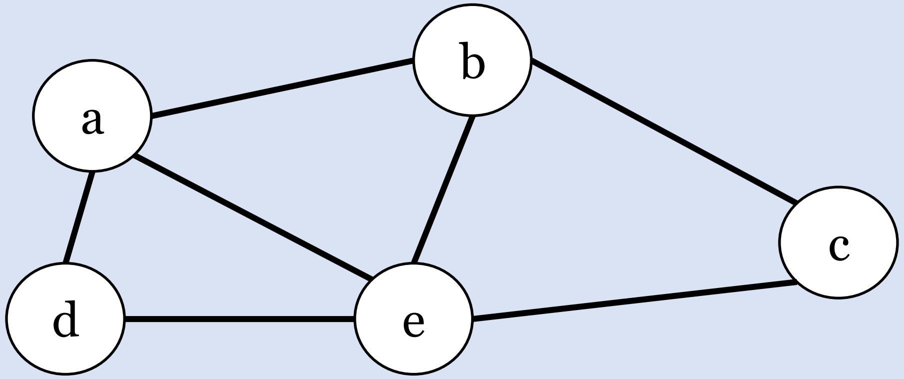
# Representing a Graph

- There are two ways we can represent graphs when writing code.
  - Matrix (2D Array)
  - Adjacency List (Array of LinkedList)



# Matrix for an undirected graph

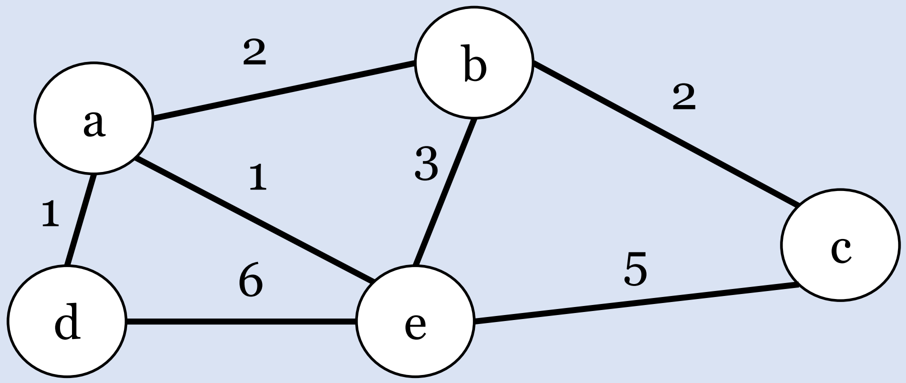
| | a | b | c | d | e |
|---|---|---|---|---|---|
| a | 0 | 1 | 0 | 1 | 1 |
| b | 1 | 0 | 1 | 0 | 1 |
| c | 0 | 1 | 0 | 0 | 1 |
| d | 1 | 0 | 0 | 0 | 1 |
| e | 1 | 1 | 1 | 1 | 0 |



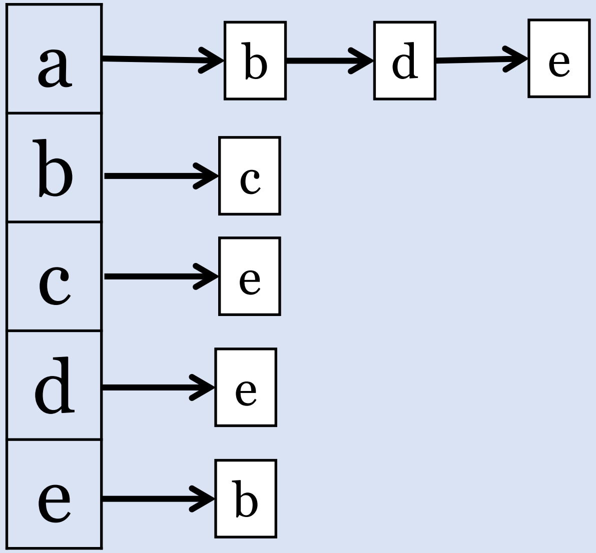

$$a_{ij} = \begin{cases} 1, & if\,(i,j) \in E \\ 0, & otherwise \end{cases}$$

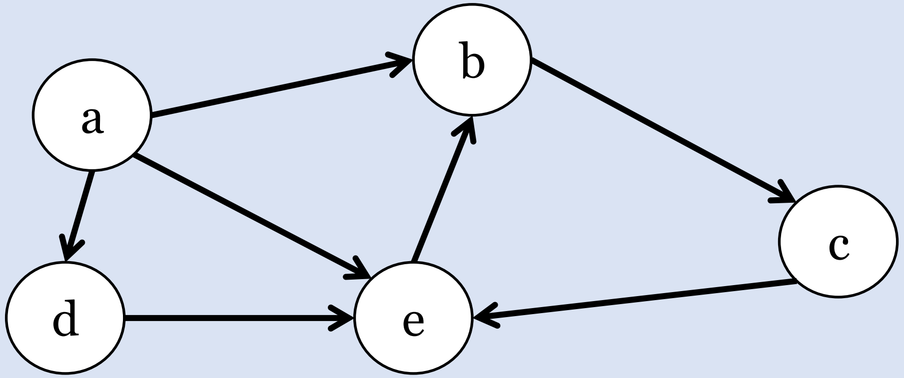
# Matrix for a directed weighted graph

| | a | b | c | d | e |
|---|---|---|---|---|---|
| a | 0 | 2 | 0 | 1 | 1 |
| b | 2 | 0 | 2 | 0 | 3 |
| c | 0 | 2 | 0 | 0 | 5 |
| d | 1 | 0 | 0 | 0 | 6 |
| e | 1 | 3 | 5 | 6 | 0 |



# Adjacency List for a directed graph



# Matrix or Adjacency List

- Both representations work for both directed and undirected graphs
- Sparse graphs (graphs where $|E|$ is much less than $|V|^2$)
  - Adjacency List is preferred
- Dense graphs (graphs where $|E|$ is close to $|V|^2$)
  - Matrix is preferred
- Adjacency List Note
  - The sum of the lengths of an adjacency list
    - $|E|$ for directed graphs
    - $|2 * E|$ for undirected graphs

# Complexity of Adjacency Lists

- Space required is $O(V + E)$
- Time needed to lists every vertex adjacent to vertex v. O(degree of v)
- Time needed to determine if $(v, u) \in E$ is O(degree of v)

# Complexity of Matrix

- Space required is $O(V^2)$
- Time needed to lists every vertex adjacent to vertex v. O(V)
- Time needed to determine if $(v, u) \in E$ is O(1)