The ADT Graph.

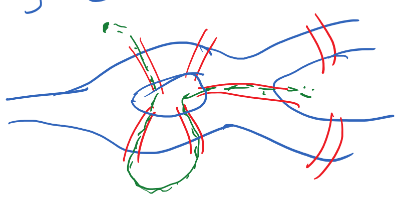
Wednesday, November 20, 2019 5:00 PM

Graph : is a collection of objects with a relationship among the objects.

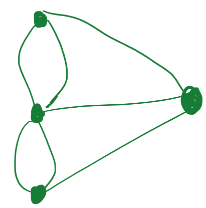
≈ 16th Century

Leonard Euler

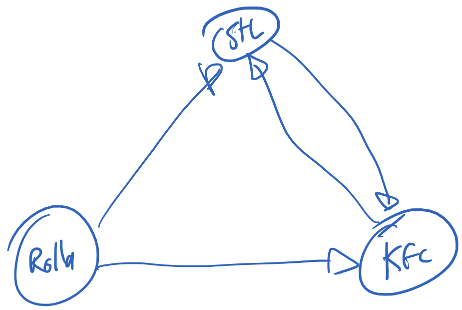
Konigsberg

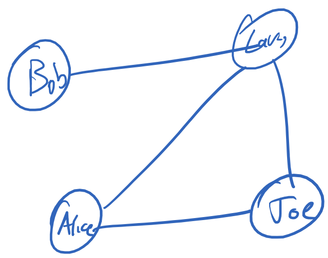


Tour the city by using each bridge once.



Graph theory





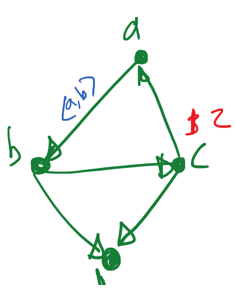
DEF: Graph = $\langle V, E \rangle$

V: set of objects = "Vertices"

E: set of pairs $\langle v, w \rangle$ where $v \in V$, $w \in V$
"edges"

if E is reflexive: $\langle v, w \rangle \in E$ iff $\langle w, v \rangle \in E$
the graph is "undirected"
else the graph is call "directed" or "digraph"

Example:



$V = \{a, b, c, d\}$

Vertices

$E = \{(a,b), (b,c), \langle c, a \rangle, \langle b, d \rangle, \langle c, d \rangle\}$

Edges

$\langle c, D \rangle$

Vertices

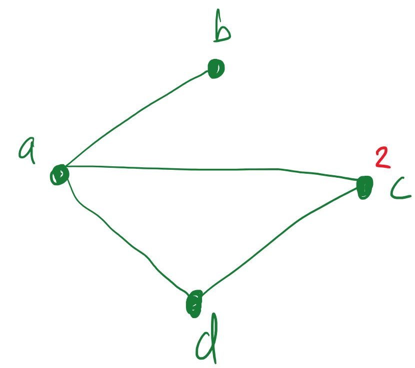
Edges

Example:

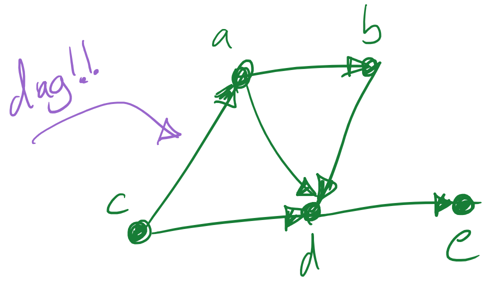
$V = \{a, b, c, d\}$

$$E \begin{cases} \langle a,b \rangle \ \langle b,a \rangle \\ \langle c,d \rangle \ \langle d,c \rangle \\ \langle a,d \rangle \ \langle d,a \rangle \\ \langle c,a \rangle \ \langle a,c \rangle \end{cases}$$

reflexive



- the edge $\langle v, v \rangle$ is called a loop.
- A graph without loops is called simple.
- w is adjacent to v if $\langle v, w \rangle \in E$
- the degree of a node v is the number of vertices adjacent to v
- the degree of a graph is the highest degree of a node in the graph.
- A path is a sequence of vertices $\langle V_0, V_1, V_2, \dots V_n \rangle$ such that $\langle V_i, V_{i+1} \rangle \in E$
  - $n-1$ is the lenght of a path.
  - a path can be empty
  - a path that does not contains a loop is called a "simple" path



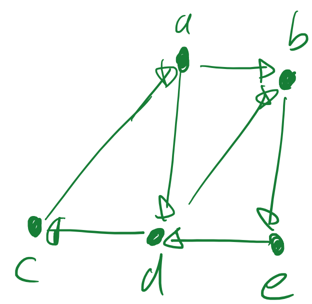
$\langle a, b, c \rangle$ ⊗

$\langle a, b, d, e \rangle$ ✓

$\langle e, d, c \rangle$ ⊗

$\langle e, d \rangle \notin E$

- A **cycle** is a path $\langle v_0, v_1, \ldots, v_n \rangle$ where $v_0 = v_n$



$\langle a, d, c, a \rangle$ = cycle.
$\langle d, b, e, d \rangle$ = cycle.

• not a dag

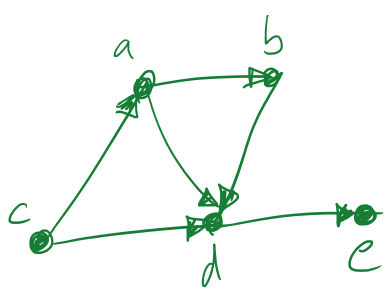
$E \{ (a,b) (b,a)$

$\langle a, b, a \rangle$ = cycle.

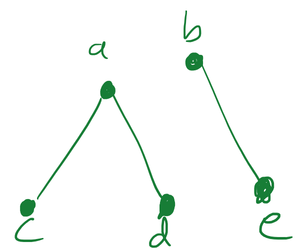
• not a dag

- **"Dag"**: Directed Acyclic Graph.

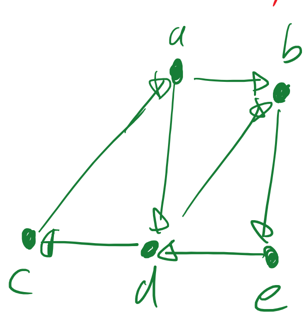
- a graph is **strongly connected** if for every two different vertices v, w there is a path $\langle v, \ldots, w \rangle$



not strongly connected
but weakly connected



not strongly connected
not weakly connected.



strongly connected

- a graph is weakly connected, if for every two different vertices v, w either there is a path $\langle v, \dots w \rangle$ or a path $\langle w, \dots v \rangle$

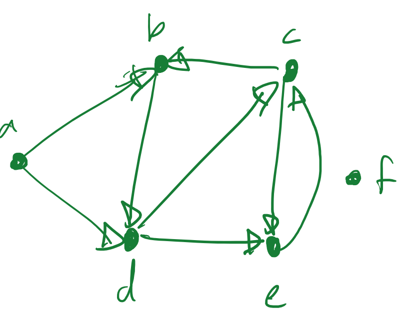
≡ OPERATIONS ≡

- addVertex (G, v)
- addEdge (G, e) $\quad e = \langle v, w \rangle$
- neighbors (G, v) = List of vertices adjacent to v

• Data Structures for Graphs:

"the Adjacency Matrix"

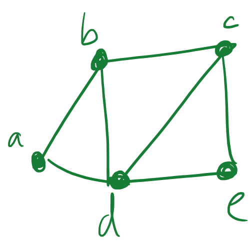
directed



From \ to	a	b	c	d	e	f
a	0	1	0	1	0	
b	0	0	0	1	0	
c	0	1	0	0	1	
d	0	0	1	0	1	
e	0	0	1	0	0	
f						

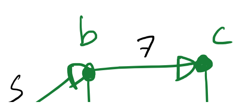
add Edge = cheap
add Node = Medium
neighbors: = Medium

undirected

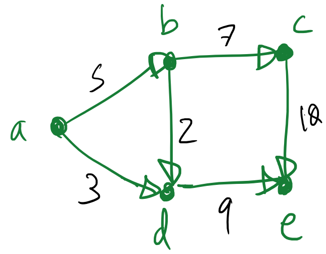


	a	b	c	d	e
a	0	1	0	1	0
b	1	0	1	1	0
c	0	1	0	1	1
d	1	1	1	0	1
e	0	0	1	1	0

weighted



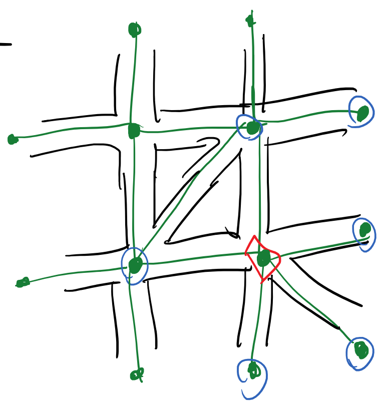
	a	b	c	d	e
a	0	5	0	3	0



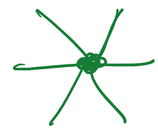
	a	b	c	d	e
a	0	5	0	3	0
b	0	0	7	2	0
c	0	0	0	0	10
d	0	0	0	0	9
e	0	0	0	0	0

Pro: Simple, Edd Edge is cheap.
Cons: Memory Expensive. (for the zeroes) Neighbors is expensive

Example



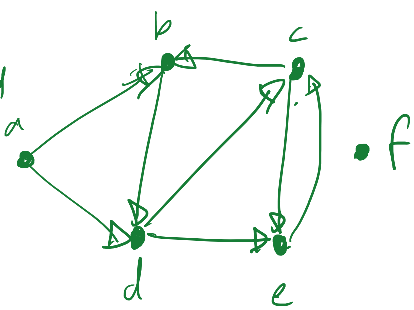
- highest degree of a vertex 6

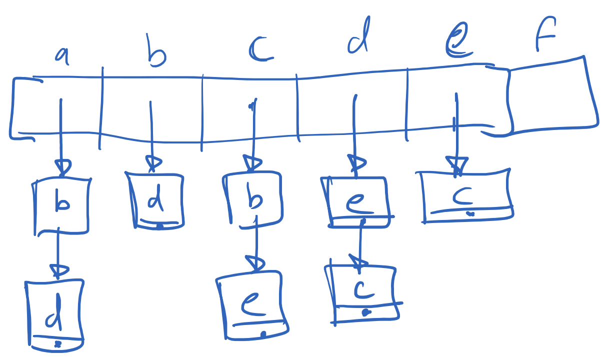


- A city graph can have 10,000 vertices.

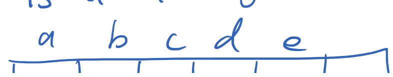
- Adjacency List Data structure

directed

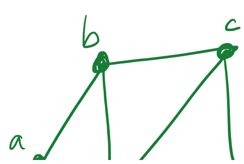




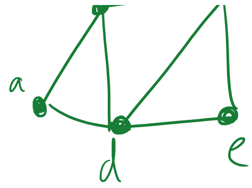
add Node cheap
add vertex cheap
Neighbors cheap.
"Is a a neighbor of b?" = expensive

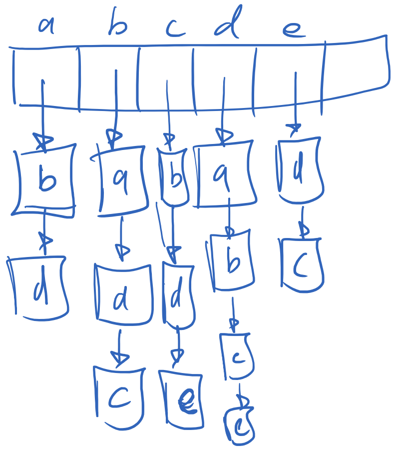


undirected



Is a a neighbor of b?





weighted