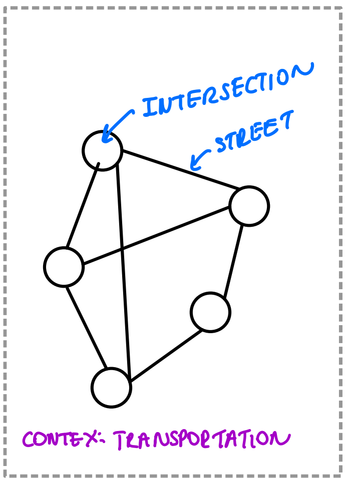
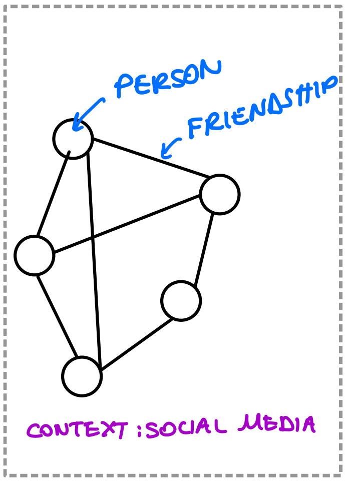
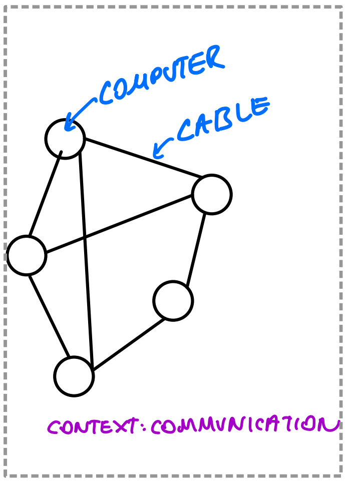
GRAPHS

AN ABSTRACT WAY OF REPRESENTING CONNECTIVITY USING NODES (ALSO CALLED VERTICES) AND EDGES

LOTS OF PROBLEMS ARE FORMULATED AND SOLVED IN TERMS OF GRAPHS

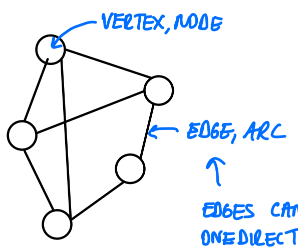






GRAPH TERMINOLOGY

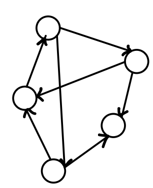
- SET OF **VERTICES** CONNECTED PAIRWISE BY **EDGES**



A GRAPH $G = (V, E)$

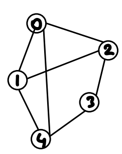
- A SET OF VERTICES, V
- A SET OF EDGES, E, WHERE EACH PAIR IS A PAIR OF (v, w) s.t. $v, w \in V$

EDGES CAN BE
ONEDIRECTIONAL (DIRECTED)
OR
BIDIRECTIONAL (UNDIRECTED)



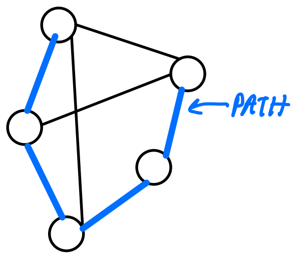
- TWO VERTICES OF A GRAPH ARE **ADJACENT** IF THEY ARE JOINED BY AN EDGE.

  VERTEX w IS ADJACENT TO v IFF $(v, w) \in E$



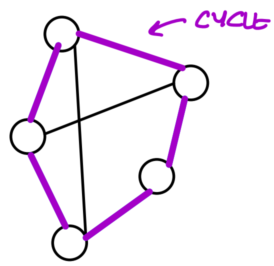
- 0 and 2 ARE ADJACENT $(0, 2)$ is an edge in set E.
- 0 and 3 ARE NOT ADJACENT

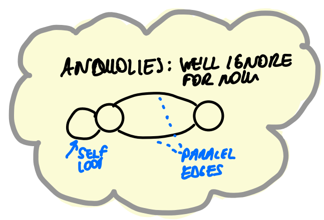
- ☐ **PATH**: SEQUENCE OF VERTICES CONNECTED BY EDGES



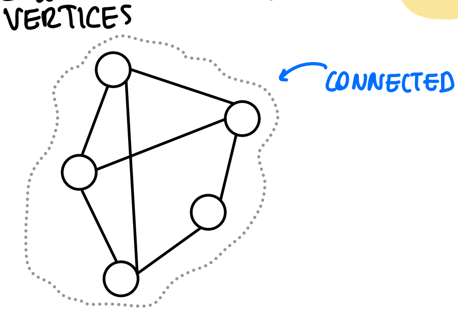
TWO VERTICES ARE CONNECTED IF THERE IS A PATH BETWEEN THEM.

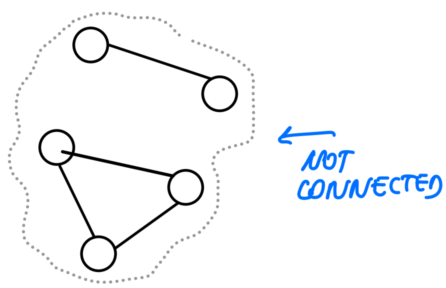
- ☐ **CYCLE**: PATH WHOSE FIRST AND LAST VERTICES ARE THE SAME



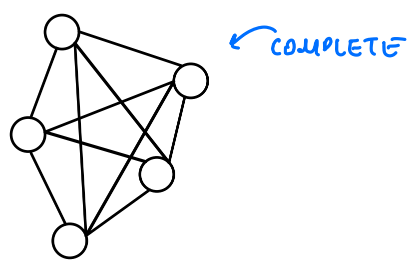


- ☐ **CONNECTED GRAPH**: HAS A PATH BETWEEN EACH PAIR OF DISTINCT VERTICES

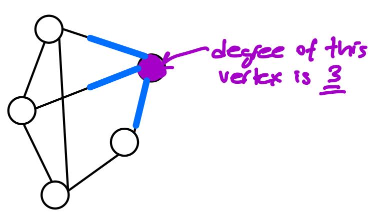




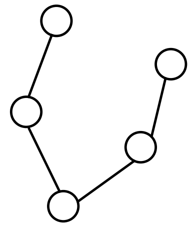
- ☐ **COMPLETE GRAPH**: HAS A EDGE BETWEEN EACH PAIR OF DISTINCT VERTICES



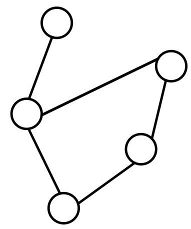
- ☐ **DEGREE OF A VERTEX**: # OF EDGES CONNECTED TO IT



- **AN ACYCLIC GRAPH:** A GRAPH WITH NO CYCLES

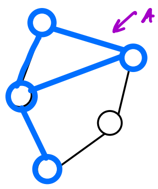


ACYCLIC GRAPH



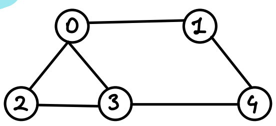
NOT AN ACYCLIC GRAPH

- A **SUBGRAPH** IS A SUBSET OF A GRAPH'S EDGES (AND VERTICES) THAT CONSTITUTES A GRAPH



A SUBGRAPH

EXAMPLE:



→ UNDIRECTED GRAPH

→ $V = \{0,1,2,3,4\}$

→ $E = \{(0,1),(1,0),(0,2),(2,0),(0,3),(3,0)$
$(1,4),(4,1),(2,3),(3,2),(3,4),(4,3)\}$

→ degree(3) = 3
degree(1) = 2

→ 0 and 1 ARE ADJACENT
→ PATH : 0,1,4 (A SIMPLE PATH)
→ CYCLE : 0,1,4,3,0 (A CYCLE)
→ CONNECTED GRAPH

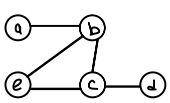
→ NOT ACYCLIC

EXAMPLE:

LET G BE THE GRAPH WITH THE FOLLOWING ADJACENCY LIST

a: b
b: a, c, e
c: b, d, e
d: c
e: b, c

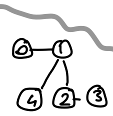
DRAW THE GRAPH



CONVENTION

WE USE INDEXES FROM 0 TO V-1 TO REPRESENT VERTICES

SYMBOL TABLE
a → 0
b → 1
c → 2
d → 3
e → 4



STORING GRAPHS

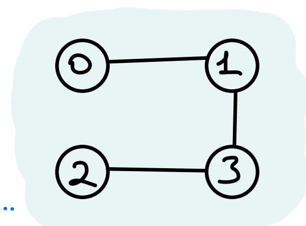
NEED TO STORE BOTH SET OF VERTICES $\underline{\underline{V}}$
AND THE SET OF EDGES, $\underline{\underline{E}}$

CAN BE STORED IN AN ARRAY

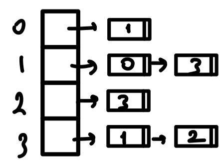
EDGES MUST BE STORED IN SOME OTHER WAY

ADJACENCY MATRIX

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 1 \\ 3 & 0 & 1 & 1 & 0 \end{array}$$



ADJACENCY LIST



$n \times n$ MATRIX

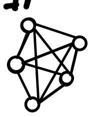
$a_{ij} = 1$ IF THERE IS AN EDGE FROM i to j

$a_{ij} = 0$ IF THERE IS NO EDGE FROM i to j.

OR BOOLEAN MATRIX
TRUE → 1 FALSE → 0

$O(V^2)$ MEMORY USAGE

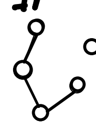
PREFERRED IF GRAPH IS DENSE



WE KEEP TRACK OF ALL THE VERTICES ADJACENT TO EACH VERTEX ON A LINKED LIST THAT IS ASSOCIATED WITH THAT VERTEX.

MEMORY USAGE ⇒ $O(E+V)$

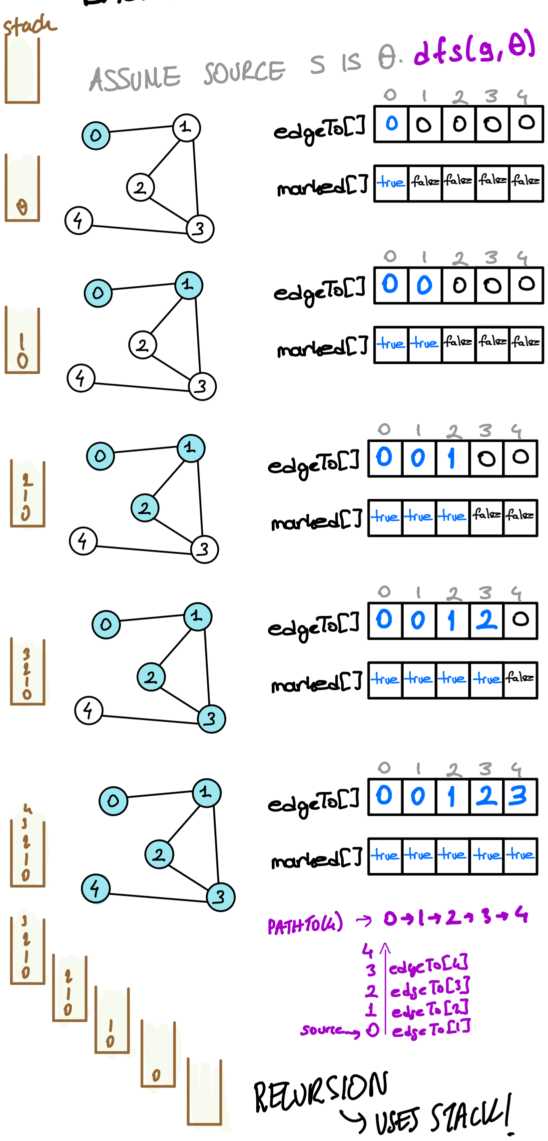
PREFERRED IF GRAPH IS SPARSE



GRAPH TRAVERSALS

DEPTH-FIRST SEARCH

EXPLORE AS FAR AS POSSIBLE ALONG EACH BRANCH BEFORE BACKTRACKING



BREADTH-FIRST SEARCH

VISIT A VERTEX, THEN VISIT ALL VERTICES ADJACENT TO THAT VERTEX.