Linked list as a priority queue

1) Insert ———— $\Theta(1)$

2) Maximum ———— $O(n)$

3) Extract_max ———— $O(n)$

4) Increase key ———— $O(n)$

Heap data structure as a priority queue

2) Maximum (A)

    return A[1] $\}$ —— $\theta(1)$

3) Extract_max (A)

    tmp = A[1]

    Swap A[1] and A[heapsize(A)]

    heapsize(A) = heapsize(A) − 1

    maxheapify (A, 1) —— $O(\log n)$

    return tmp

Total time:

$O(\log n)$

4) Increase-key (A, i, key)

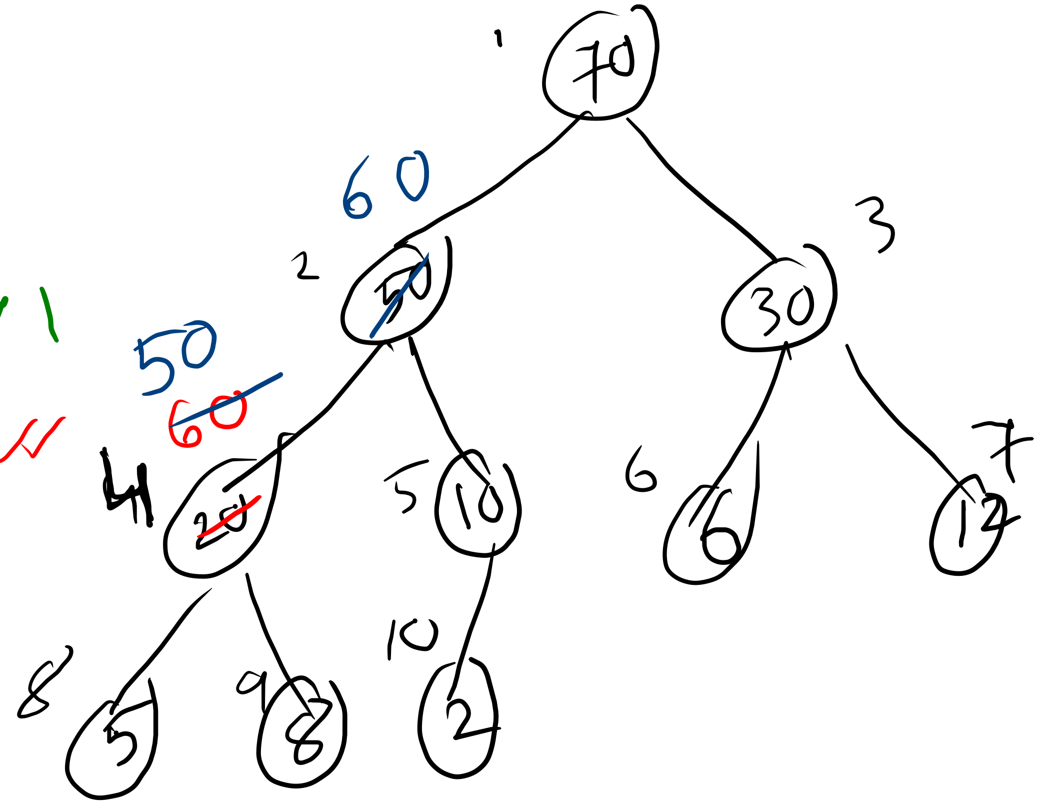
if A[i] > key

do nothing.

While A[i] > A[Parent(i)] and i > 1

swap A[i] and A[Parent(i)]

i = Parent(i)



Increase-key (A, 4, 60)

Running time:

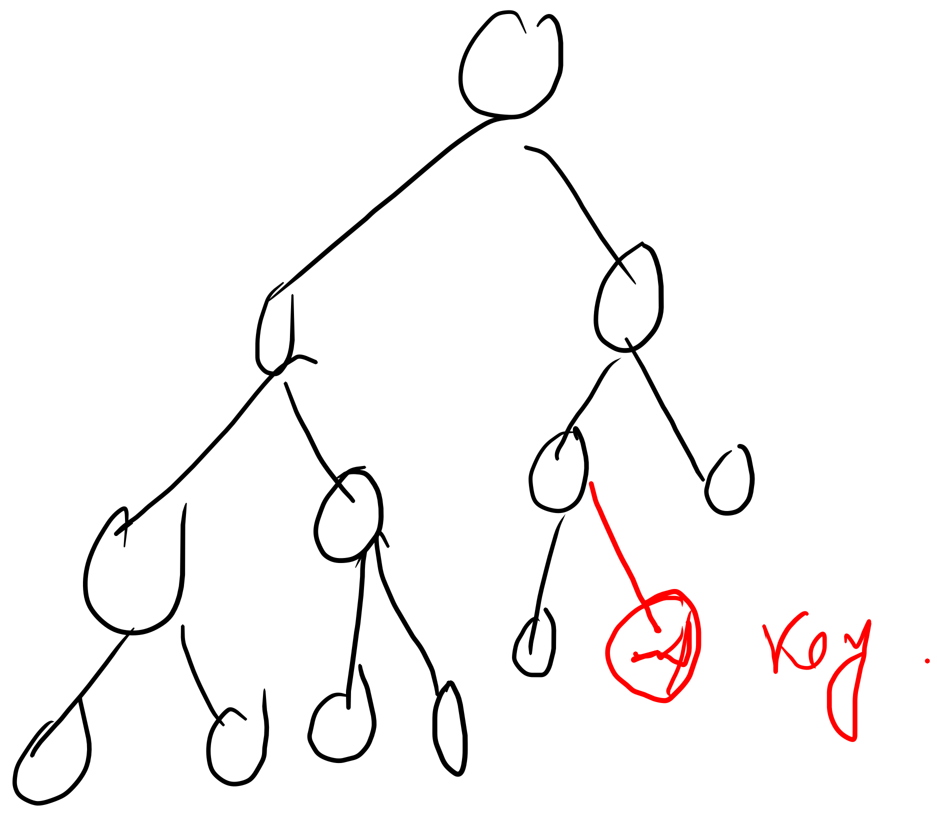
O(log n)

1) Insert (A, key)

heapsize(A) = heapsize(A) + 1

A[heapsize(A)] = $-\infty$

Increase-key(A, heapsize(A), key)



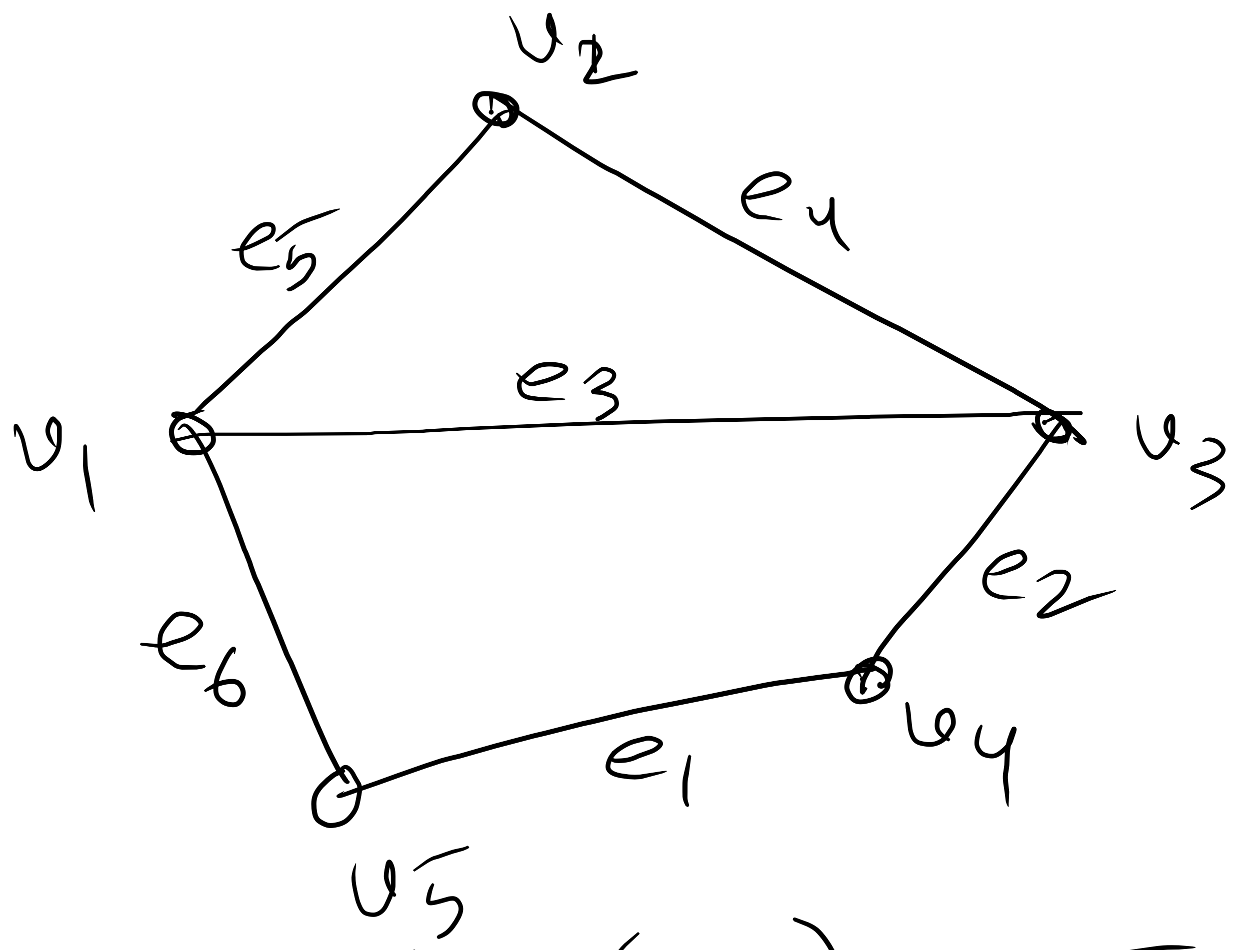
Running time:

$O(\lg n)$

Graphs G

collection of vertices and edges.

V (vertices), E (edges)



$G(V, E)$

$V = \{v_1, v_2, \ldots, v_5\}$

$E = \{e_1, e_2, \ldots, e_6\}$

$e_1 = \{v_5, v_4\}$

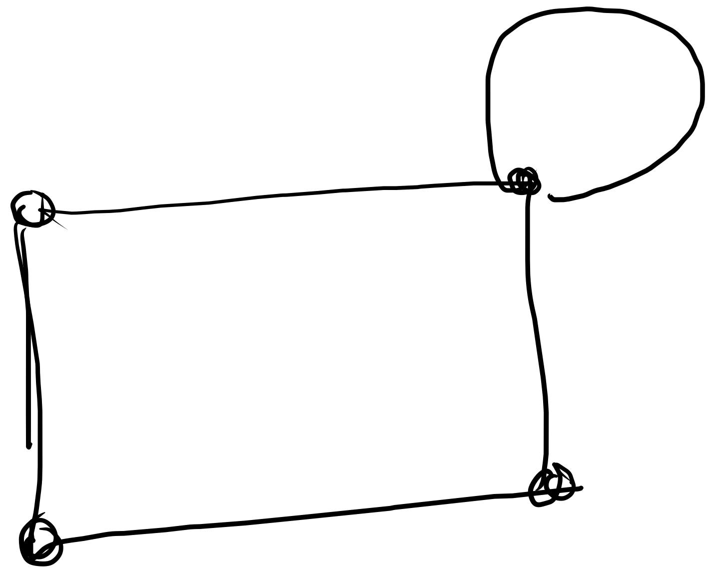
$e_2 = \{v_4, v_3\}$

$\vdots$

# of vertices $(|V|) = 5$

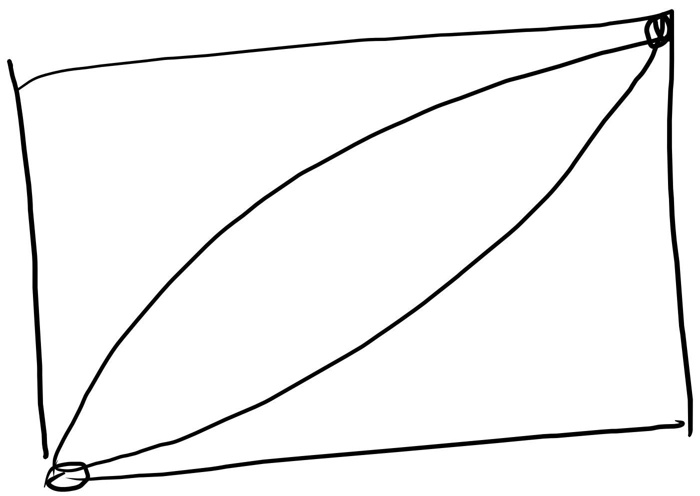
# of edges $(|E|) = 6$

Self loop



start and end of an edge is same.

multiedges



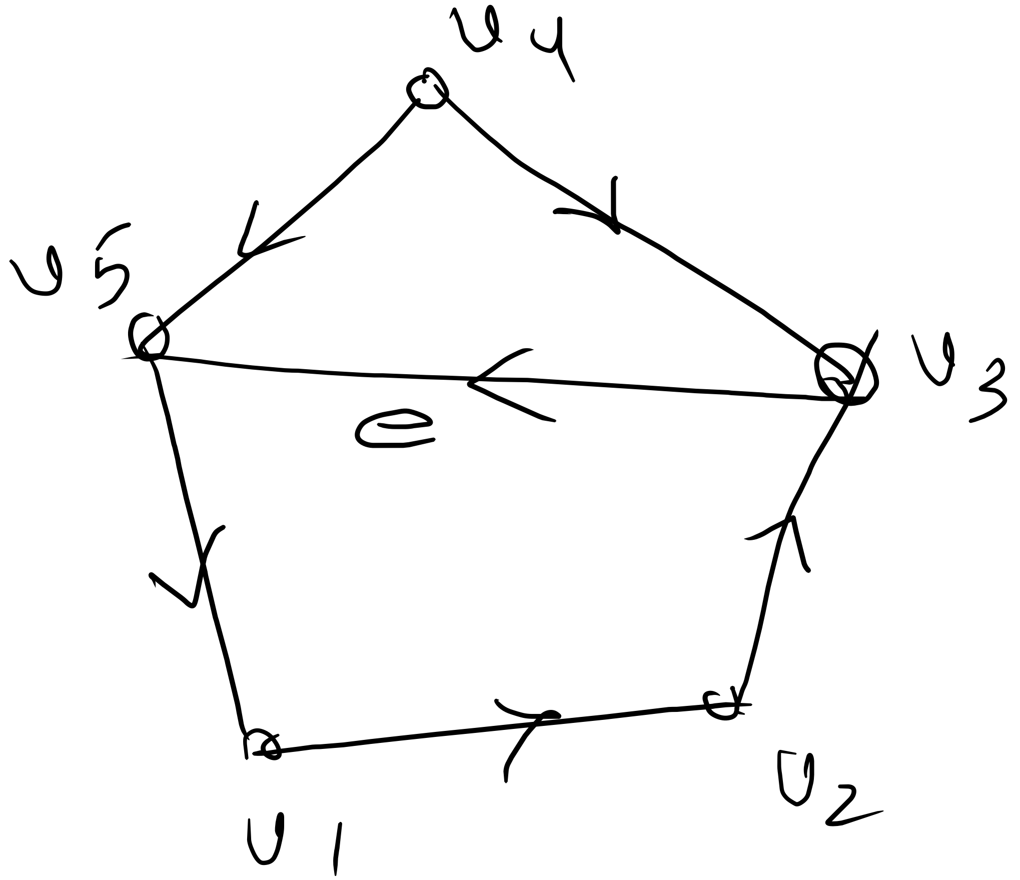
more than one edge
between a pair of vertices.

Siple graph

A graph that have no self loop and no multiedges.

Directed graph

A graph $G(V, E)$ where each edge has a direction.



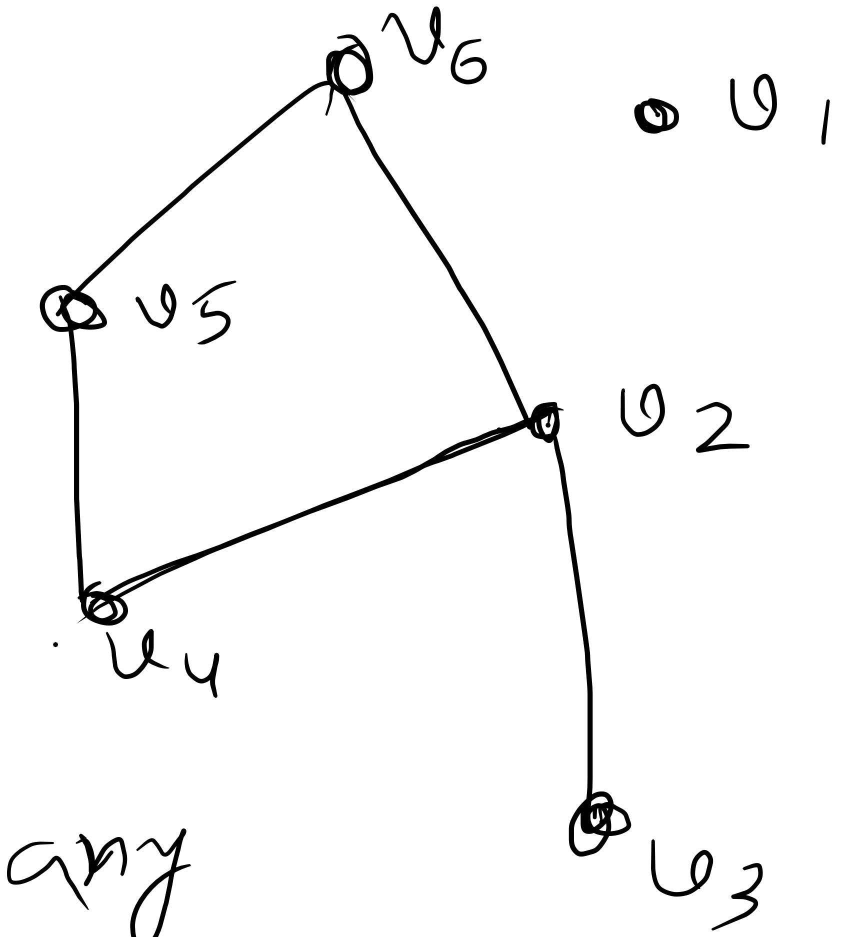
$e = (u, v)$

$u \leftarrow$ initial/start vertex.

$v \leftarrow$ end/terminal vertex

Degree (Undirected)

degree of a vertex is the number of edges incident on that vertex.



$degree(v_1) = 0$ ← isolated vertex.

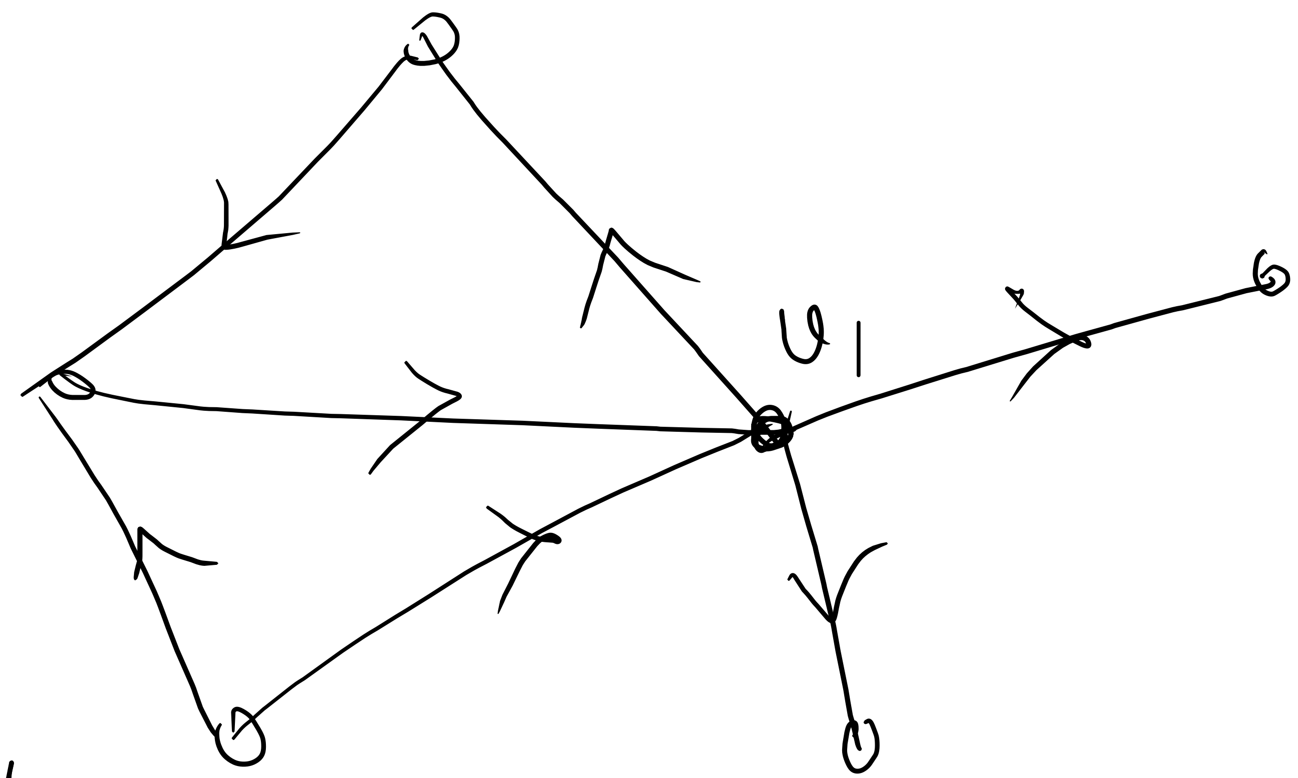
$degree(v_2) = 3$

$degree(v_3) = 1$ ← pendant vertex.

degree of the graph: maximum degree of any vertex.

$degree(G) = 3$

Degree (directed)



In-degree: # edges directed towards the vertex

out-degree: # edges directed outwards the vertex.

$\text{Indegree}(v_1) = 2$

$\text{out-degree}(v_1) = 3$

<u>Result</u> (Handshaking theorem)

$$\sum_{v \in V} \deg(v) = 2|E|$$

each edge increases the degree count by 2

$e = (u, v)$

1 $\leftarrow$ count in u

1 $\leftarrow$ count in v.

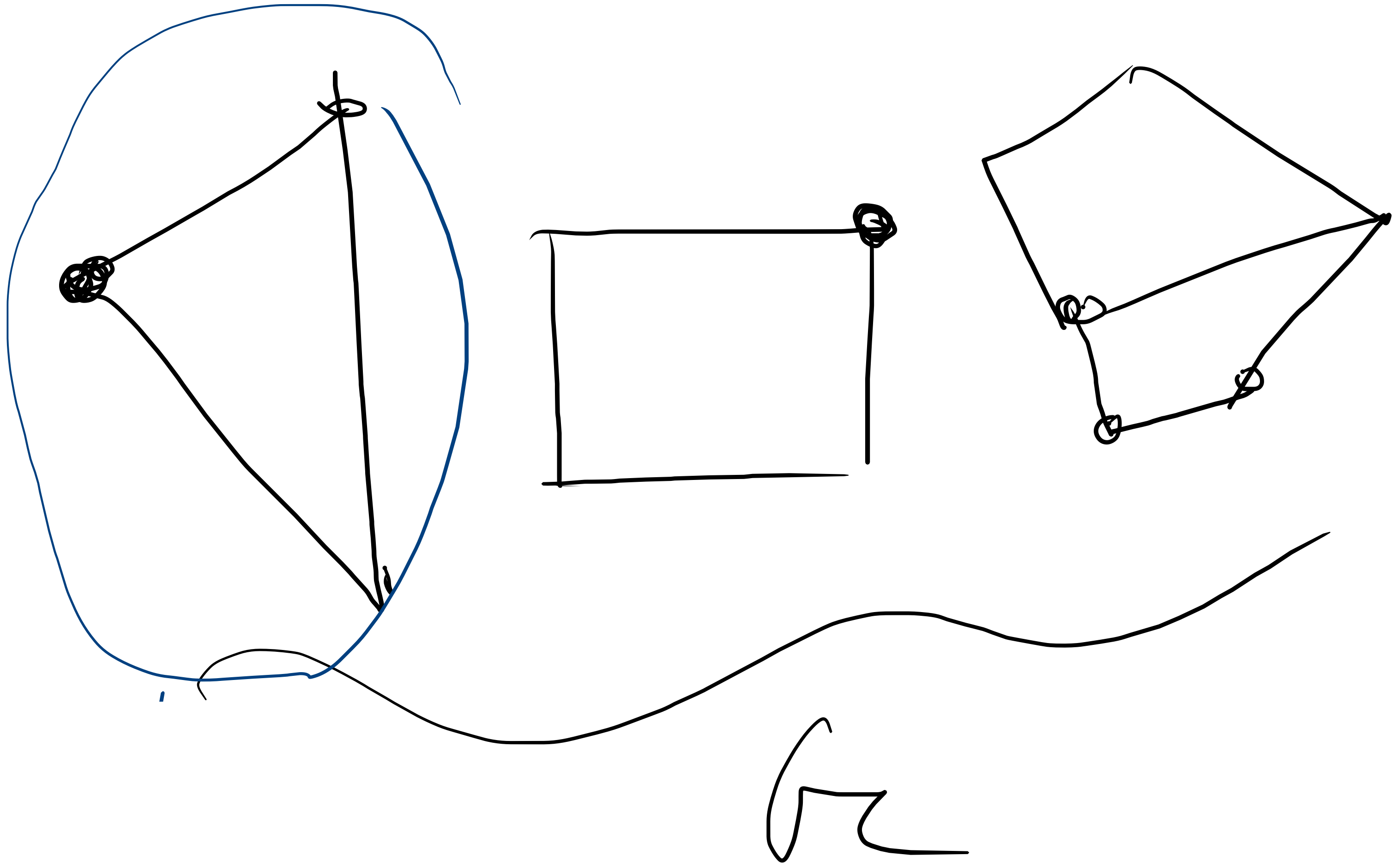
Result

$$\sum_{v \in V} \text{indeg}(v) = \sum_{v \in V} \text{outdeg}(v) = |E|$$

Types of graphs

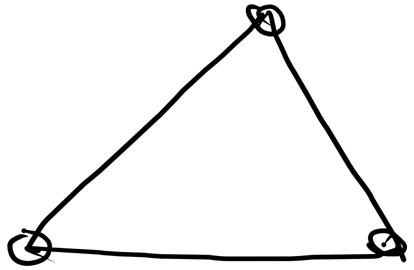
i) connected graph:

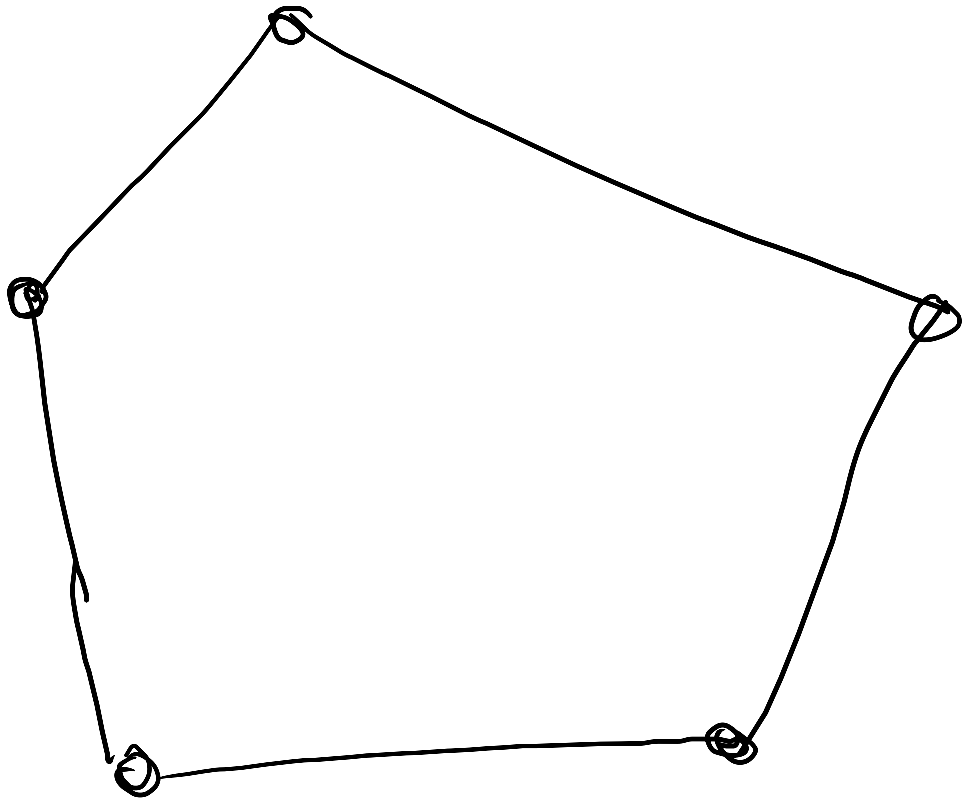
For every pair of vertices there is a path.



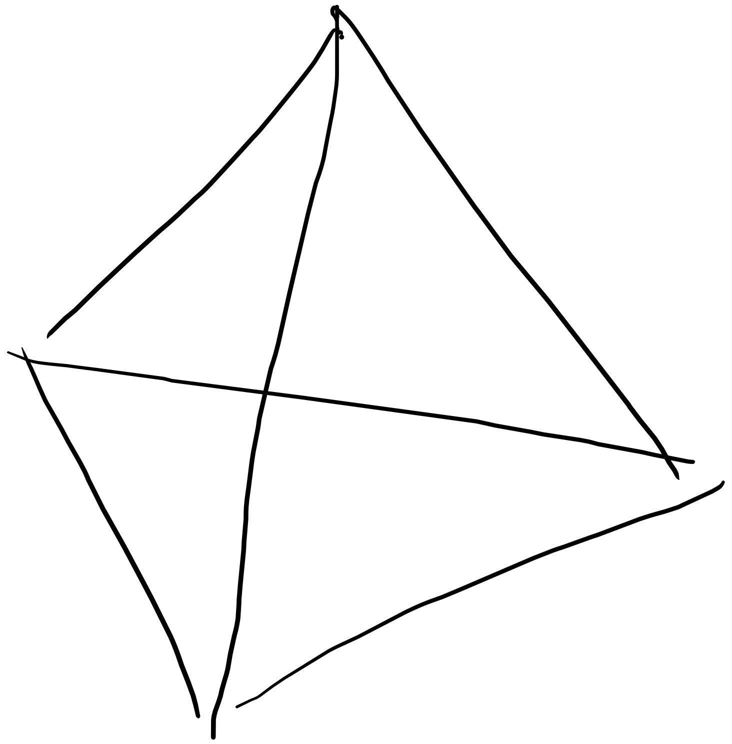
Regular graph

Each vertex is of same degree.





complete graph



K_4

Between each pair of vertices there is an edge.

(K_n)

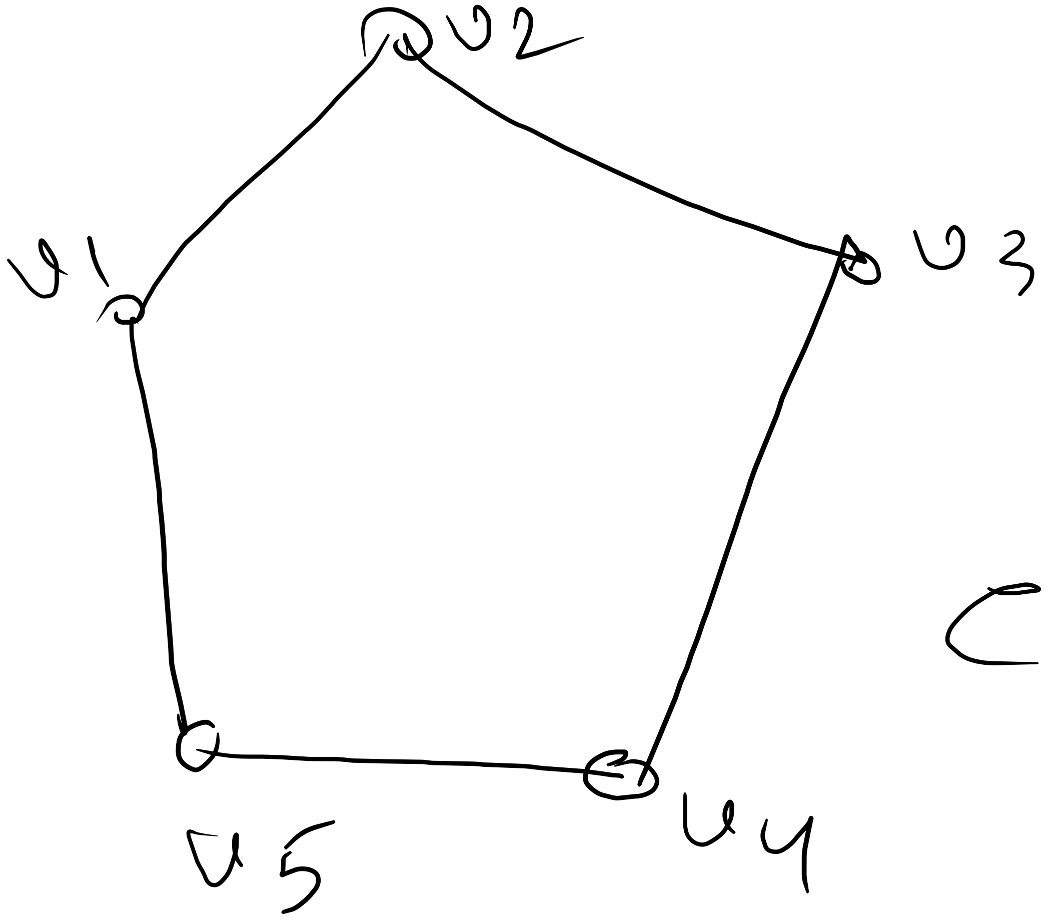
$(n-1)$- regular

cycle graph

a set of $n \geq 3$ vertices $v_1, v_2, \ldots, v_n$ forms a cycle graph containing edges.

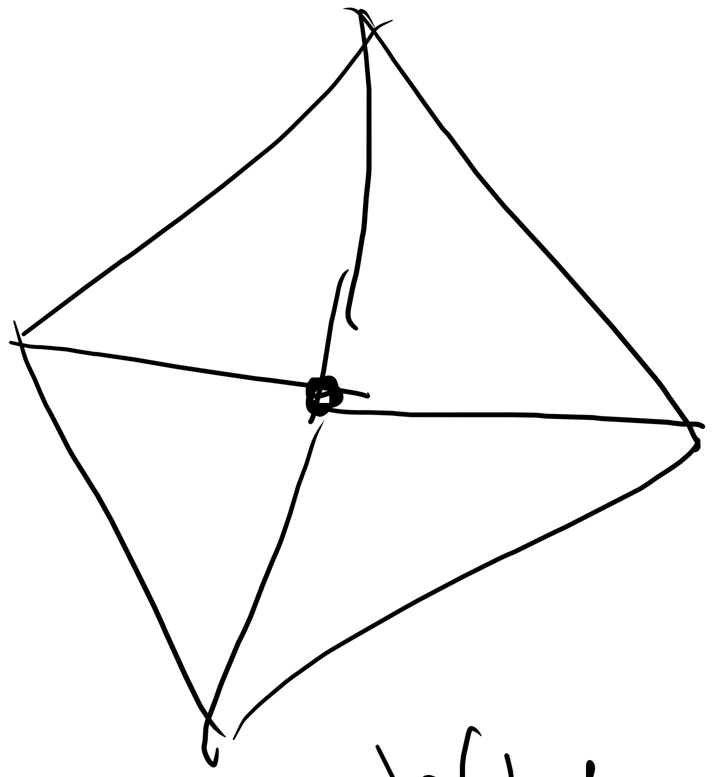
$(v_1, v_2)\ (v_2, v_3) \ldots\ (v_{n-1}, v_n), (v_n, v_1)$

(C_n)



C_5

Wheel graph



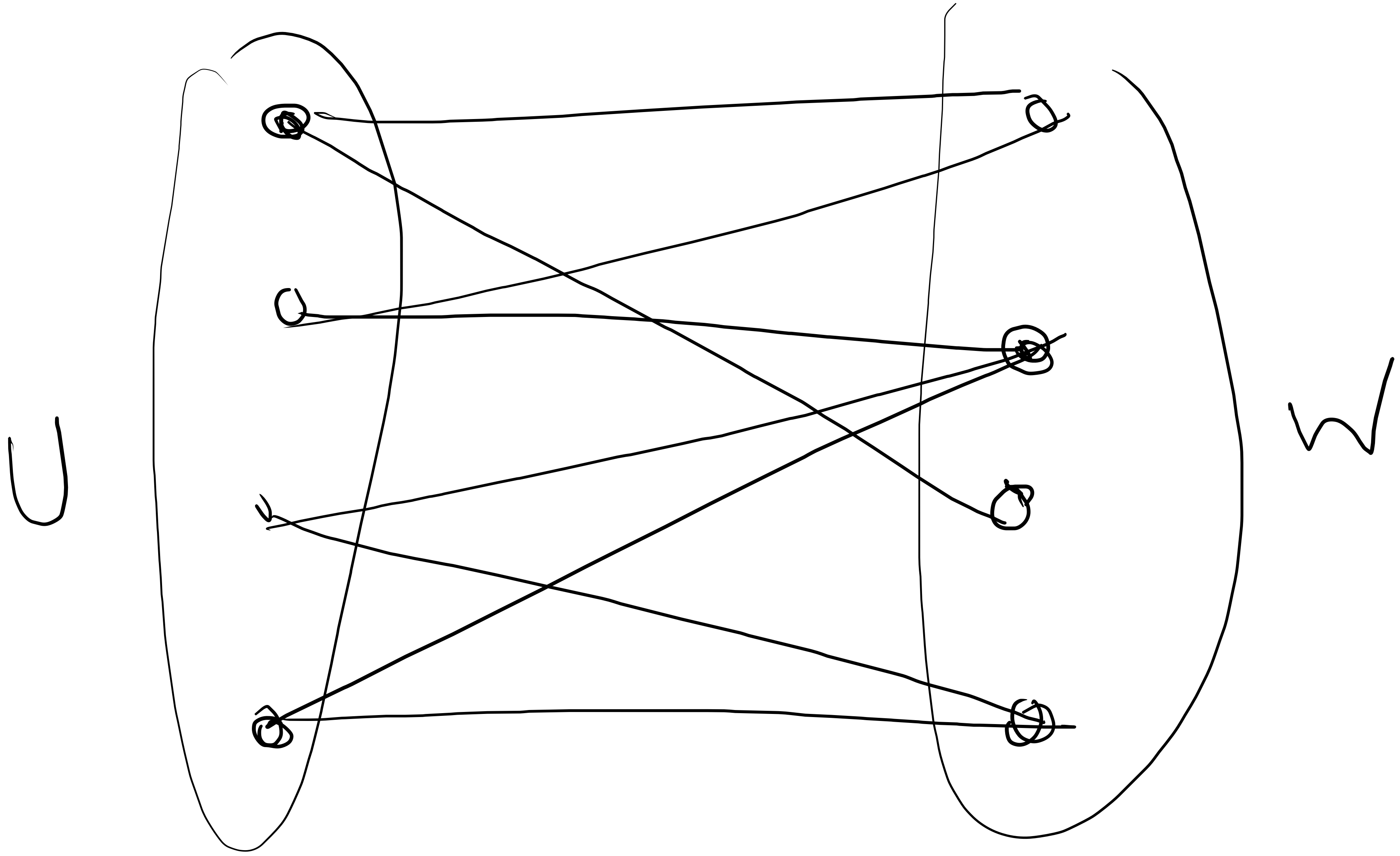
W4

cycle graph + one extra vertex connecting all other vertices by edges

(W_n)

Bipartite graph

- If the vertex set V is partitioned into two sets U and W and each edge is connecting one vertex in U and other vertex in W



→ complete bipartite graph

each pair (u, v) $v \in U$ and $u \in W$ are connected by an edge.

K_{mn}

Subgraph

A subgraph of a graph $G(V,E)$ is a graph $G'(V',E')$ where $V' \subseteq V$ and $E' \subseteq E$

Representation of graphs

1) Adjacency matrix
2) Adjacency list
3) Incidence matrix.

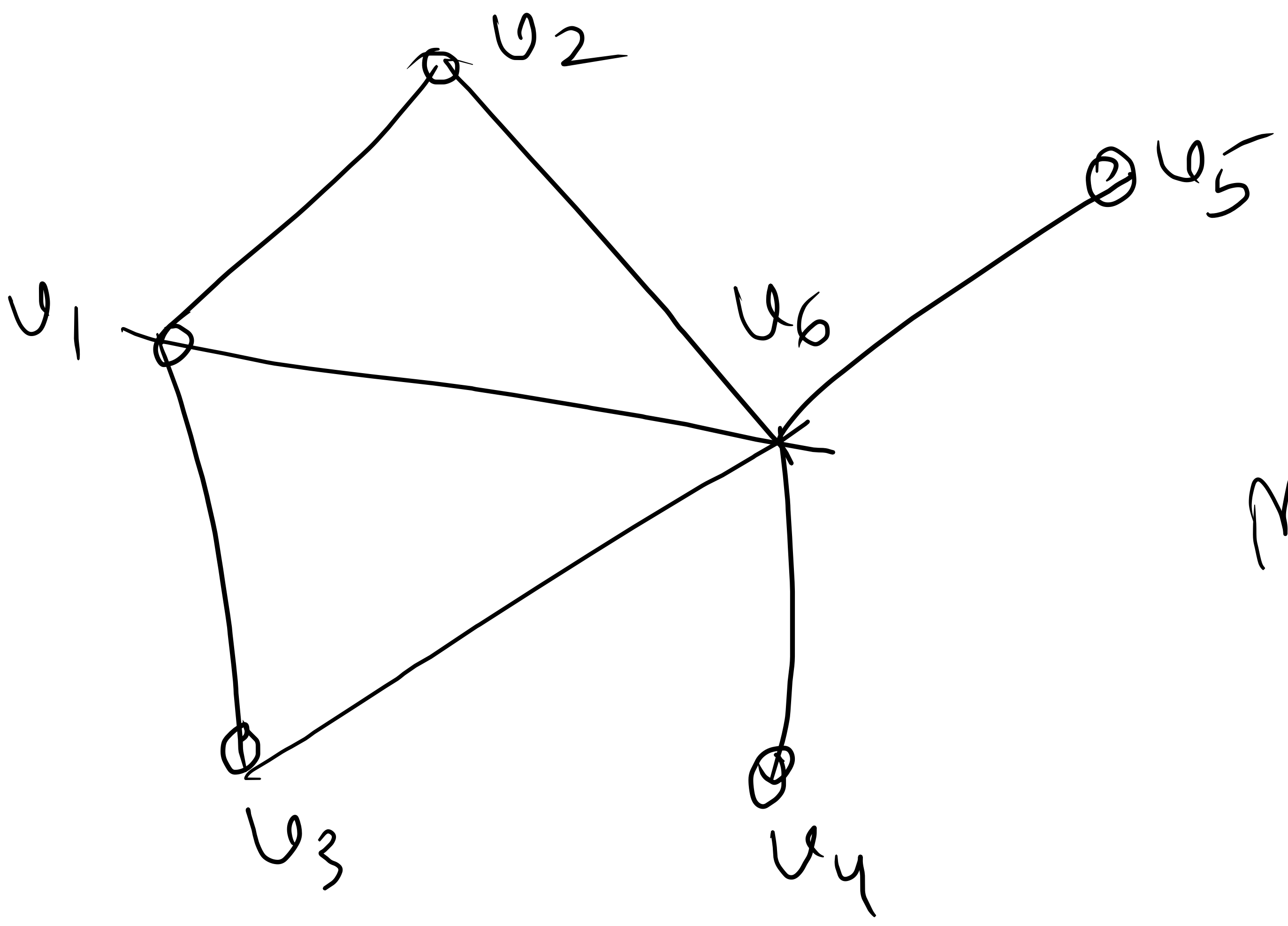
Adjacency matrix

$G(V,E)$ be a graph $|V| = n$

consider a matrix M of size $n \times n$.

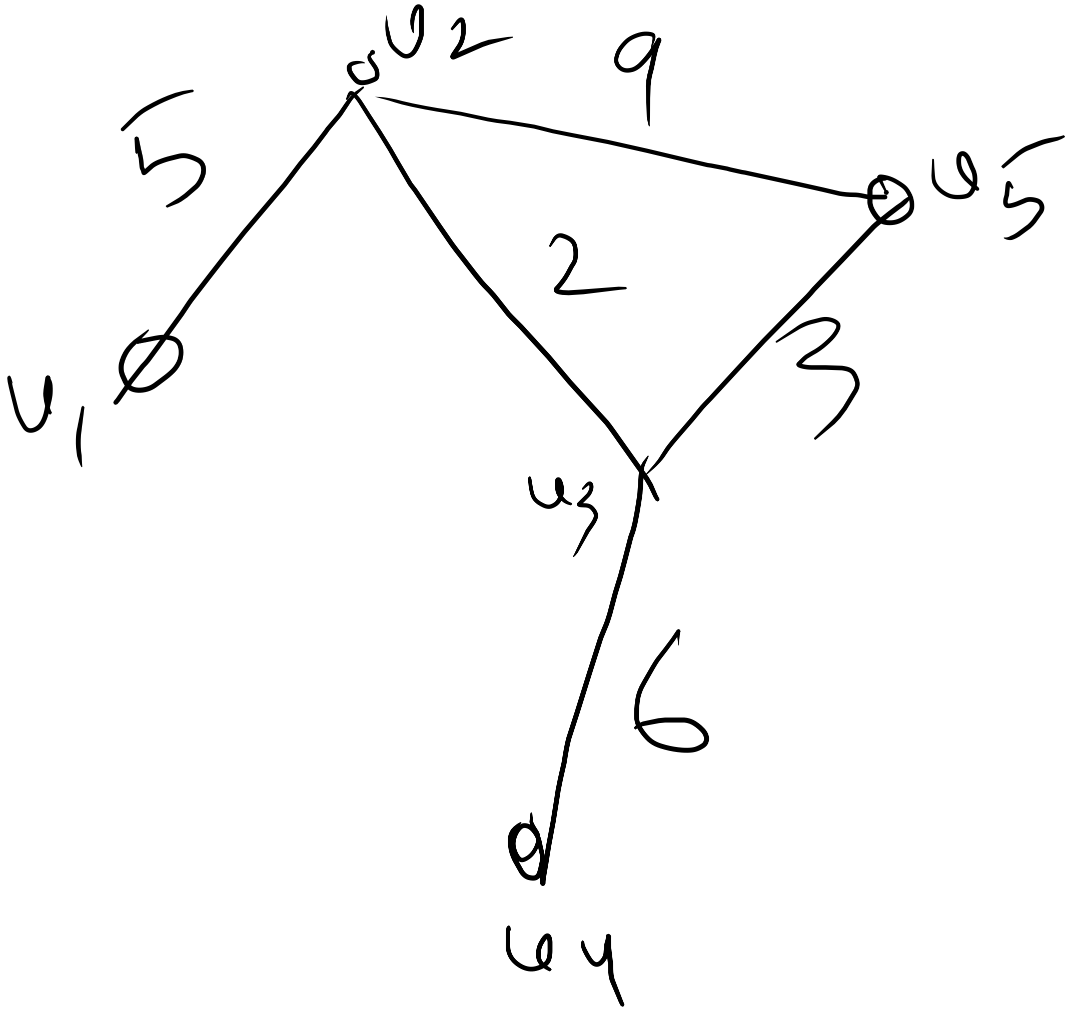
$$M = (m_{ij})_{n \times n}$$

$$m_{ij} = \begin{cases} 1 & \text{if } v_i \text{ and } v_j \text{ is an edge} \\ 0 & \text{Otherwise.} \end{cases}$$



$$M = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \left[\begin{array}{cccccc} 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{array}\right] \end{array}$$

unweighted
& undirected.



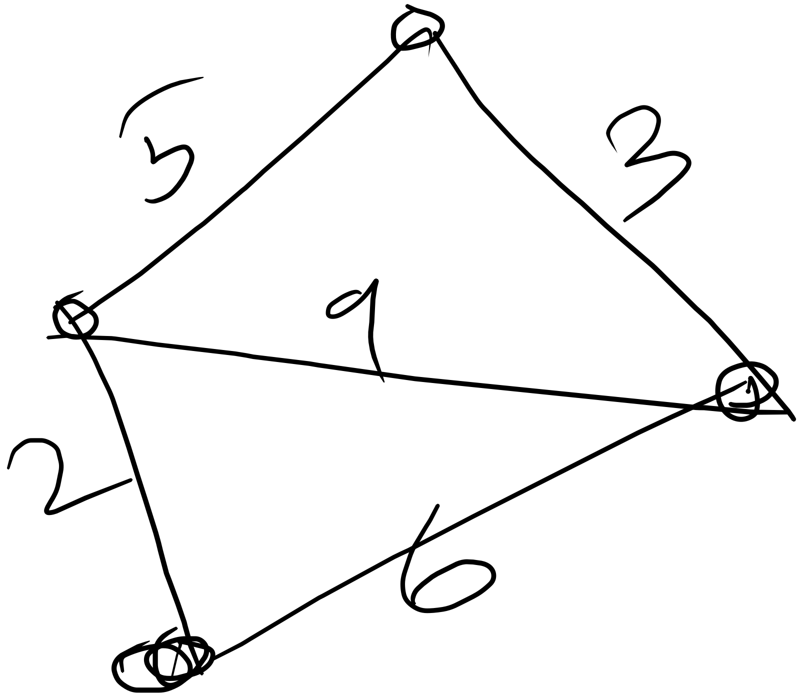
	1	2	3	4	5
1	0	5	0	0	0
2	5	0	2	0	9
3	0	2	0	6	3
4	0	0	6	0	0
5	0	9	3	0	0

unweighted and weighted graph.

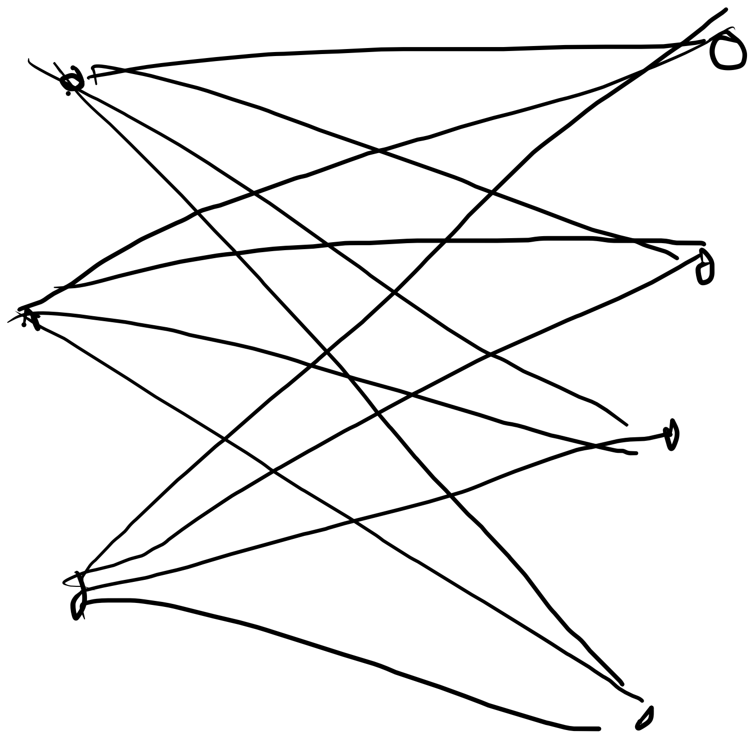
normally edge-weighted.

<u>weighted graph</u>;

each edge has a weight.



Exm



$K_{3,4}$