Graph Theory

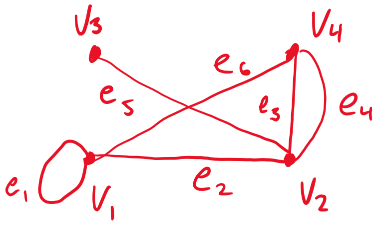
- Many real-world problems concern objects and relations, e.g. people with friendships, cities connected by highways, web pages linked to others, etc.
- The mathematical abstraction of these situations is the study of graph theory.
- A **graph** is a collection of points and curves.

Definition:

- A **graph** G consists of a pair of finite sets:
  - A nonempty set of V of **vertices** and a set of E of **edges**, where each edge is associated to a subset of V of either 1 or 2 vertices, called the **endpoints** of the edge.
- An edge with just one endpoint is called a **loop**.
- 2 edges with the same endpoints are called **parallel edges**.
- An edge is said to **connect** its endpoints and be **incident** on each endpoint.
- A vertex on which no edges are incident is called **isolated.**
- 2 vertices connect by an edge are called **adjacent.**

Exercise:

Write down V and E for the following graph. List any loops and parallel edges.

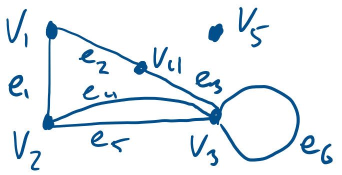


$V = \{V_1, V_2, V_3, V_4\}$

$E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$

Exercise:

Draw a graph that has 5 vertices include 1 isolated, 1 loop, and 1 pair of parallel edges.



$V = \{V_1, V_2, V_3, V_4, V_5\}$

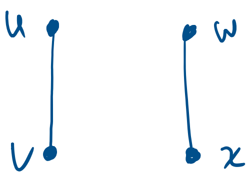
$E = \{(V_1, V_2), (V_1, V_4), (V_4, V_3),$
$(V_3, V_2), (V_3, V_2), (V_3, V_3)\}$

Definition (Simple Graph):

- A **simple graph** is one that does not have loops nor parallel edges.

Exercise:

Draw a simple graph with $V = \{u, v, w, x\}$ and 2 edges, one of which has endpoints u and v.



Definition (Complete Graph):

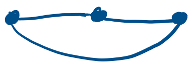
- A **complete graph** on n vertices, denoted by K_n, is a simple graph with n vertices whose edge set contains exactly one edge for every pair of distinct vertices.

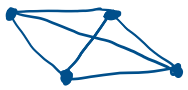
Exercise:

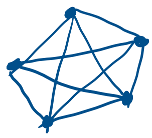
Draw K_1, K_2, K_3, K_4, K_5.

K_1:


K_2:


K_3:


K_4:


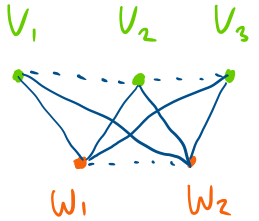
K_5:


Definition (Complete Bipartite Graph):

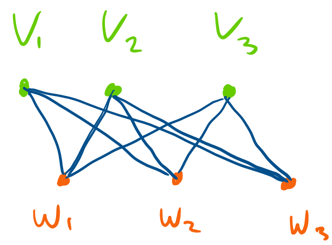
- A **complete bipartite graph** on (m, n) vertices, denoted by K_{m_1,n_1} is a simple graph with $V = \{V_1, \dots, V_m, W_i, \dots, W_n\}$ such that for all $1 \le i, k \le m$ and all $1 \le j, l \le n$, we have
  1) An edge from each V_i to each W_j;
  2) No edge from any V_i to any other V_k;
  3) No edge from any W_j to any other W_l

Exercise:

Draw $K_{3,2}, K_{3,3}$.

$K_{3,2}$:


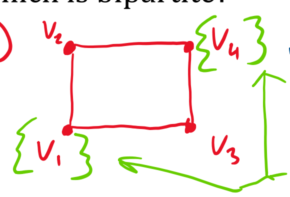
By (2), ··· not allowed

$K_{3,3}$


- A simple graph is **bipartite** if there exists $\mathcal{U} \subseteq V$ and $W \subseteq V$ such that:
  1) $\mathcal{U} \cup W = V$ and $\mathcal{U} \cap W = \emptyset$;
  2) Every edge connects to a vertex of $\mathcal{U}$ with a vertex of W.

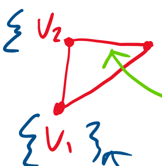
Exercise:

Which is bipartite?

a)


✓

take these subsets.

b)


Subset U

X

fails bipartite rule $V_2 \not\to V_3$

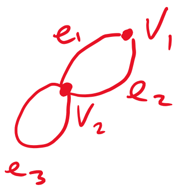
Subset W

Definition (subgraph):

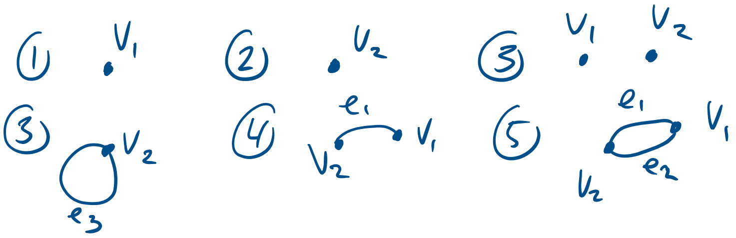
- A graph H is a **subgraph** of a graph G if
  - Every vertex in H is in G;
  - Every edge in H is in G; and
  - Every edge in H has the same endpoints in G.

Exercise:

Draw all the subgraphs of

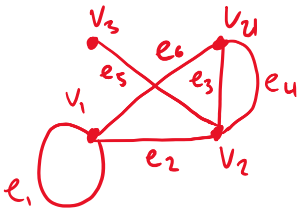


Subgraphs:

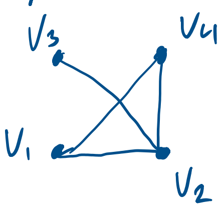


Exercise:

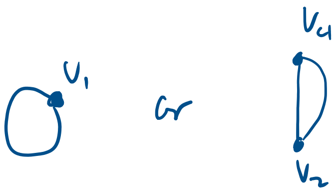
Draw 2 subgraphs containing 2 vertices each, one simple and one not.



① Simple H_1



② NOT simple



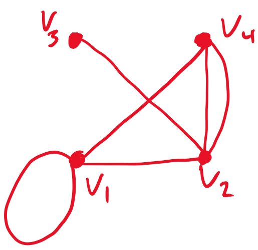
Definition (Degree):

- Let G be a graph, $v \in V$.
- The **degree** of v, denoted by $\delta(v)$, is the number of edges incident on v(with loops counted twice).
- The degree of G is the sum of degrees of all $v \in V$.

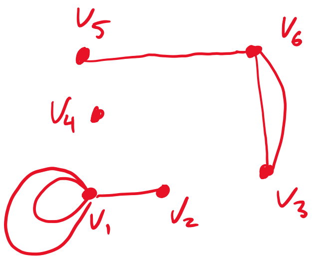
Exercise:

Find the degree of G_1 and G_2.

G_1 :



G_2 :



$\delta(v_1) = 4$

$\delta(v_2) = 4$

$\delta(v_3) = 1$

$\delta(v_4) = 3$

$\delta(G_1) = 4 + 4 + 1 + 3 = 12$

$\delta(G_2) =$ number of edges $\times 2$

$= 6 \cdot 2$

$= 12$

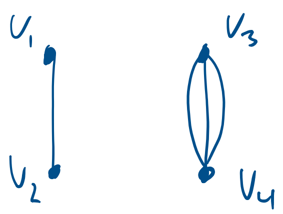
Exercise:

Draw graphs with $|V| = 4$ and vertices of degree:

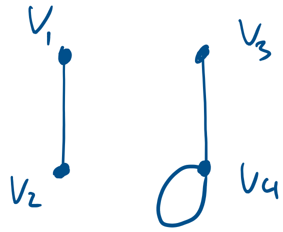
a) $1, 1, 3, 3$;

b) $1, 1, 2, 3$

a)



b)



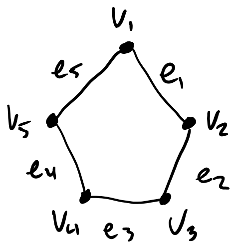
Not possible

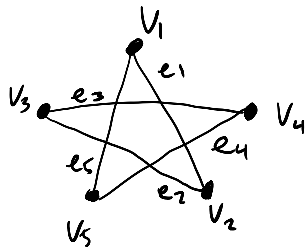
The Hand Shake Theorem:

- The degree of a graph is twice the number of its edges.
- This holds because each edge always has 2 endpoints.
- So, the degree of a graph is always even, and a graph with 4 vertices of degree 1, 1, 2, 3 is impossible.

Isomorphic Graphs

Is there any difference between these graphs?





No

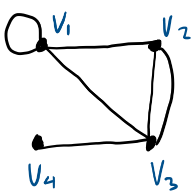
Definition:

- Let G, G' be graphs $G = (V, E), G'(V', E')$.
- We say G is **isomorphic** to G' if there exist bijective functions $f: V \to V', h: E \to E'$ that preserve adjacency, i.e. V is an endpoint of $e \Leftrightarrow f(v)$ is an endpoint of $h(e)$.

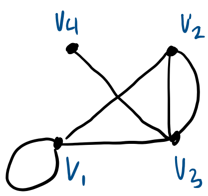
Exercise:

Show that G and G' are isomorphic.

G:



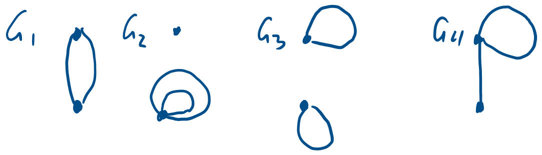
G':



$f: \{v_1, v_2, v_3, v_4\} = \{w_1, w_2, w_3, w_4\}$

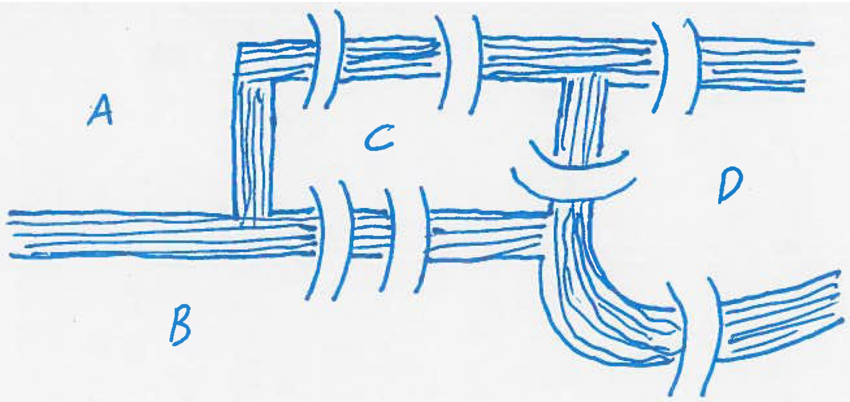
Exercise:

Draw all possible graphs (up to isomorphism) with $|V| = |E| = 2$.



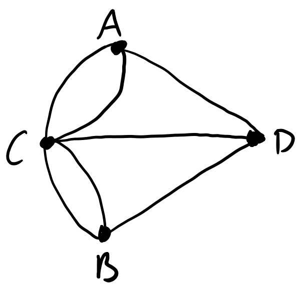
The Königsberg Bridge Problem

- In 1736, Leonhard Euler introduced graph theory by solving the following problem.



Is it possible for someone to walk in Königsberg, starting and ending at the same point, and crossing each of 7 bridges exactly once?

- This can be translated to a graph: bridges are edges and regions A, B, C, D are vertices.



Is it possible to find a route through the graph that starts and ends at a vertex and traverses each edge exactly once?

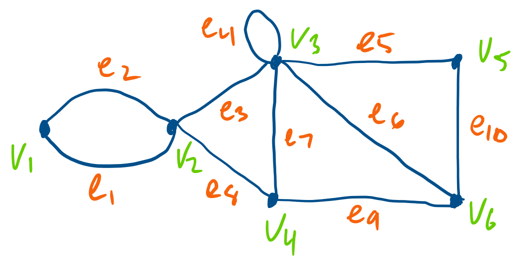
Walks, Paths, and Circuits

- **Walk:** from vertex V to vertex W in G is a finite alternating sequence of adjacent vertices and edges of G that starts at V and ends at W:

$$v_0 e_1 v_1 e_2 \dots v_{n-1} e_n v_n, \text{where } v_o = v \text{ and } v_n = w$$

  - **Length:** of a walk is the number of edges in the sequence.
  - If it is not ambiguous, a walk can be denoted by a sequence of only vertices or only edges.
- **Trail:** a walk that does not contain a repeated *edge*.
- **Path:** a trail that does not contain a repeated *vertex*.
- **Circuit:** a walk whose first and last vertices are the same.
- **Simple circuit:** a trail whose first and last vertices are the same.

Exercise:



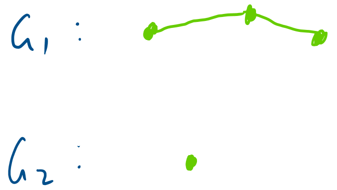
Are the following walks, trails, paths, circuits, or simple circuits?

1) $v_1 e_1 v_2 e_3 v_3 e_4 v_3 e_5 v_5$
2) $e_1 e_3 e_4 e_4 e_6$
3) $v_2 v_3 v_4 v_6$
4) $v_2 v_3 v_4 v_2$
5) $v_1 e_1 v_2 e_1 v_1$
6) v_1

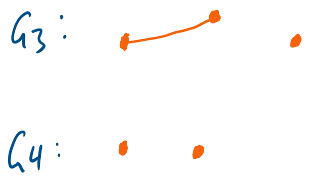
Definition (Connected):

- 2 vertices v, w in G are **connected** if there exists a walk from v to w.
- G is connected if there is a walk between every pair of vertices.
- Otherwise, G is **disconnected.**

Example: CONNECTED GRAPHS



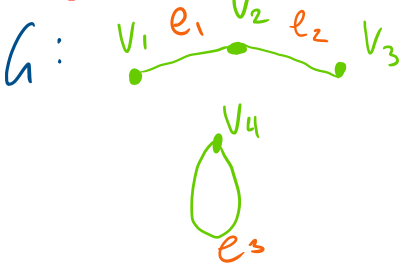
DISCONNECTED GRAPHS



Definition (Connect Component):

- A graph H is a **connected component** of G if:
  1) H is a subgraph of G
  2) H is connected
  3) No connected subgraph of G has H as a subgraph and contains vertices or edges that are outside of H.

Example:



$(\{v_1, v_2, v_3\}, \{e_1, e_2\})$ is a CONNECTED COMPONENT of G

$(\{v_1, v_2\}, \{e_1\})$ is NOT.

Definition (Eulerian Circuit):

- An **Eulerian circuit** of G is a simple circuit that contains every vertex and every edge of G.
- If an Eulerian circuit exists, G is an **Eulerian Graph**.
- An **Eulerian path** from v to w is a path from v to w that passes through every vertex in G at least once, and every edge in G exactly once.

Theorem:

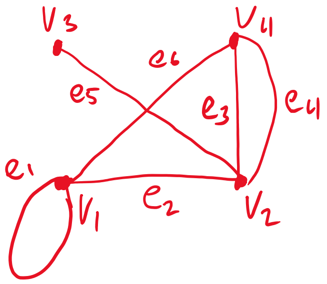
- If G is an Eulerian graph, then every vertex of G has an even degree.
- Equivalently, if some vertex of G has an odd degree, then G is not Eulerian.

Proof:

G has an Eulerian Circuit, which uses each edge exactly once. Beginning at vertex v, follow the circuit. As the circuit passes through a vertex, it uses 2 edges; one arriving to the vertex and one leaving it. Each edge is used once, so each vertex uses an event number of incident edge endpoints. The starting point v is of even degree as well, since the circuit begins by leaving v, then using v an even number of times, then arriving at v. ■

Exercise:

Does the following have an Eulerian circuit? An Eulerian path?



Not an Eulerian circuit, because V_3 and V_4 have an odd degree.

Yes, an Eulerian path if you start or end at V_3 or V_4.

Euler's Theorem:

- In a connected graph, the degree of every vertex is even and positive IFF the graph is Eulerian.

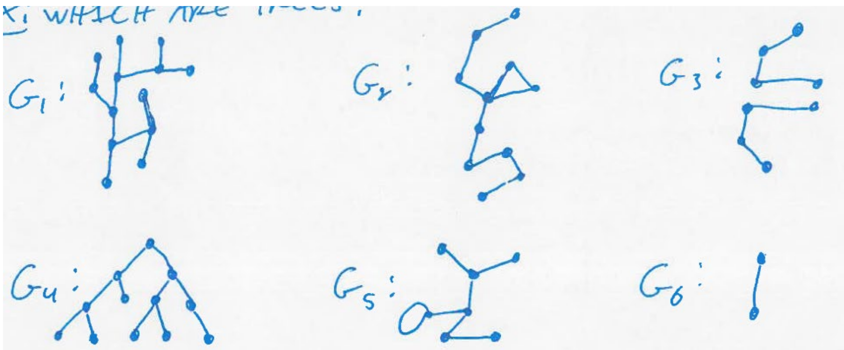
Tree

Definition:

- A graph is a **tree** if it is connected and has no circuits.

Example:

Which are trees?



A: G_1, G_4, G_6

Theorem:

- For any $n \in \mathbb{N}$, a tree with n vertices has $n-1$ edges.

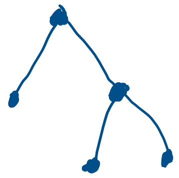
Theorem:

- For any $n \in \mathbb{N}$, if G is connected with $|V| = n$ and $|E| = n-1$, then G is a tree.

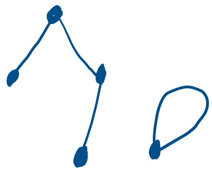
Exercise:

Draw a tree with 5 vertices and 4 edges. Draw a graph with 5 vertices and 4 edges that is not a tree.

a)



b)

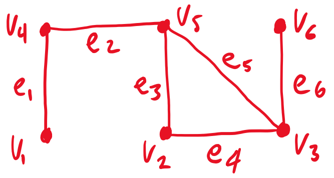


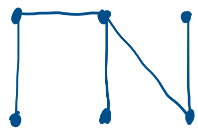
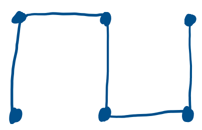
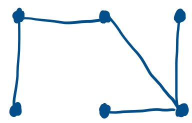
Definition (Spanning Tree):

- A **spanning tree** of G is a subgraph that contains every vertex of G and is a tree.
- Every connected graph has a spanning tree.
- Any 2 spanning trees for a graph have the same number of edges.

Exercise:

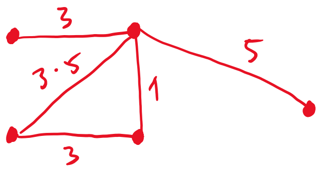
Find all spanning trees.



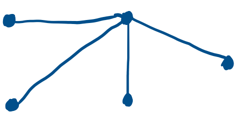
A: ①  ②  ③ 

Exercise:

Let the edges represent phone lines, the number represent the cost (in thousands) of installing the lines.



Find the spanning tree and determine the minimum cost.



$3 + 1 + 3 + 5 = 12$

Definition (Weighted Graph):

- A **weighted graph** is a graph for which each edge has an associated positive weight.
- The sum of edge weights is the weight of the graph.

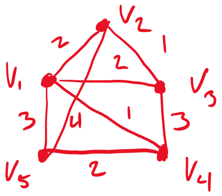
- A **minimum spanning tree** for a connected, weighted graph is a spanning tree that has the least possible weight.
- NOTE: minimum spanning trees are not necessarily unique.
- We use $w(e)$ and $w(G)$ for the weights of edge e and graph G.

Kruskal's Algorithm

- To find a minimum spanning tree, the edges are examined in order of increasing weight.
- At each step, we add an edge to what will be the minimum spanning tree, one does not create a circuit.

Example:

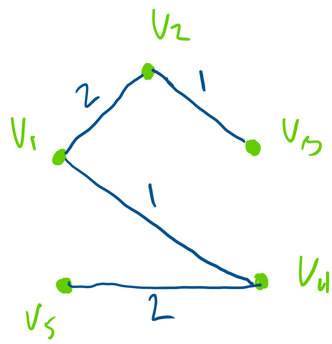
Find a minimum spanning tree.



A: First, put the edges in order by weight.

Edge	Weight	Will adding edge make a circuit?	Action	Cumulative Weight
V_2V_3	1	No	Add	1
V_1V_4	1	No	Add	2
V_1V_3	2	No	Add	4
V_2V_3	2	Yes	Skip	4
V_4V_5	2	No	Add	6
V_3V_4	3	Yes	Skip	6
V_1V_5	3	Yes	Skip	6
V_2V_5	4	Yes	Skip	6

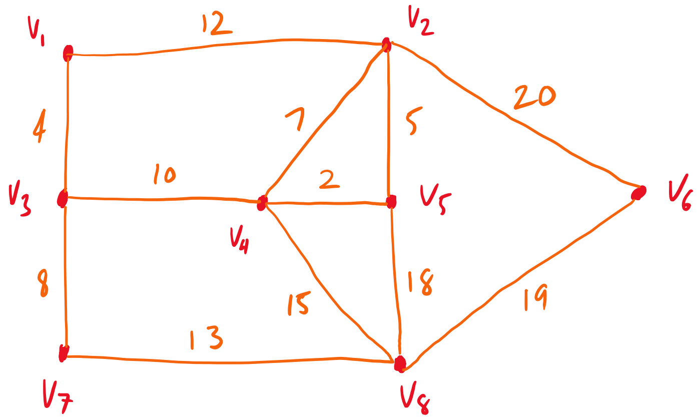
MST:

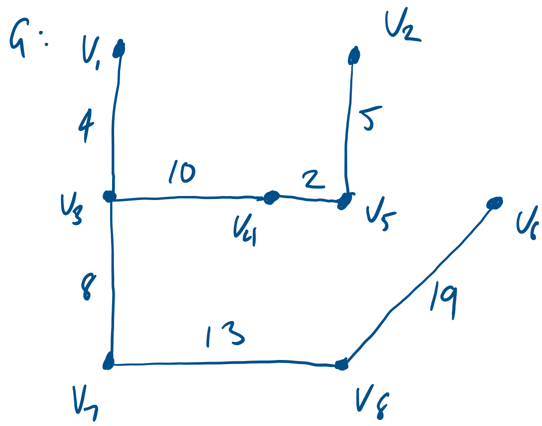


$\therefore w(G) = 6$

Exercise:

Find a minimum spanning tree.





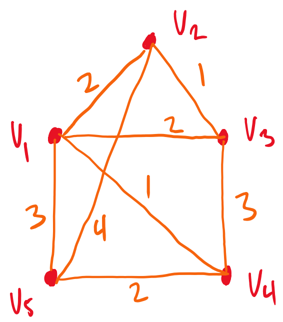
Edge	Weight	Circuit?	Action	Cumulative Weight
$V_4 V_5$	2	No	Add	2
$V_1 V_3$	4	No	Add	6
$V_2 V_5$	5	No	Add	11
$V_2 V_4$	7	Yes	Skip	11
$V_3 V_7$	8	No	Add	19
$V_3 V_4$	10	No	Add	29
$V_1 V_2$	12	Yes	Skip	29
$V_7 V_8$	13	No	Add	42
$V_4 V_8$	15	Yes	Skip	42
$V_5 V_8$	18	Yes	Skip	42
$V_6 V_8$	19	No	Add	61
$V_2 V_6$	20	Yes	Skip	61

$\therefore\ w(G) = 61$

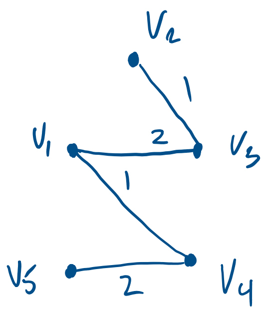
Prim's Algorithm

- Build a minimum spanning tree by choosing a vertex and expanding outwards adding one edge and one vertex at each step.

Example:



M :

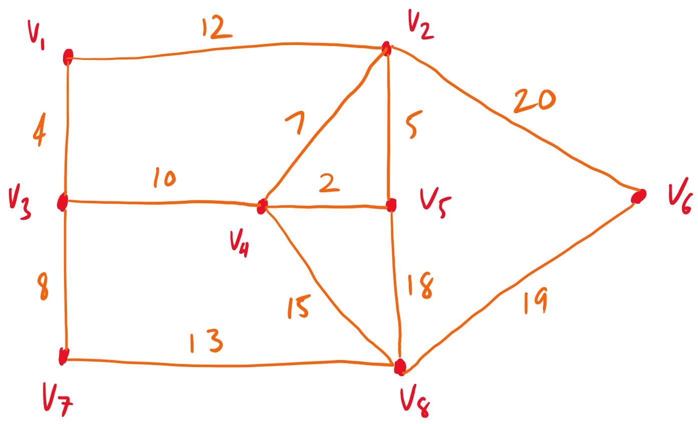


A: Start with (arbitrarily) V_1.

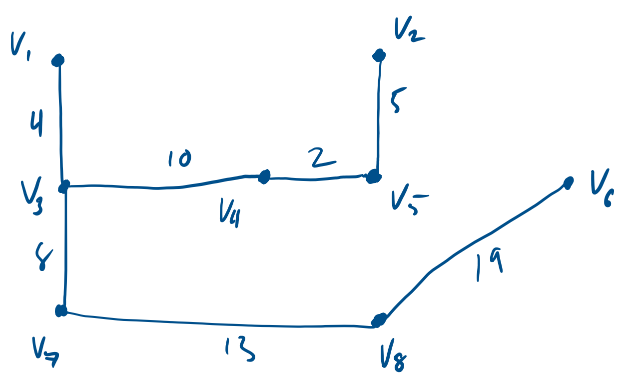
Vertex Added	Edge Added	Weight	Cumulative Weight
V_4	V_1V_4	1	1
V_3	V_1V_3	2	3
V_2	V_2V_3	1	4
V_5	V_4V_5	2	6

Exercise:

Find the MST with Prim's algorithm.



A: Start with V_4 :



$V_5 \to V_4V_5 : 2$

$V_2 \to V_2V_5 : 5$

$V_3 \to V_3V_4 : 10$

$V_1 \to V_1V_3 : 4$

$V_7 \to V_3V_7 : 8$

$V_8 \to V_7V_8 : 13$

$V_6 \to V_6V_8 : 19$

$\therefore w(G) = 61$