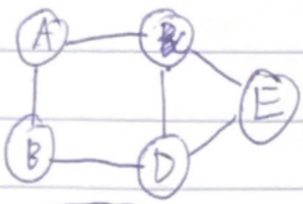
2022-11-04

Graph Theory

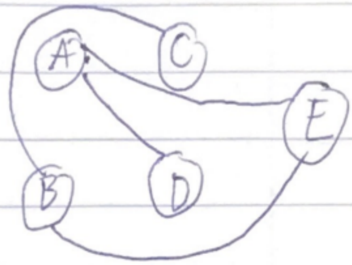
Exercise

Ex:

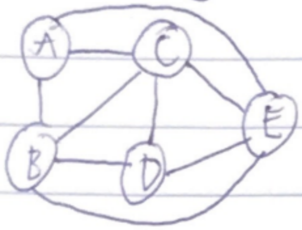
G_1:



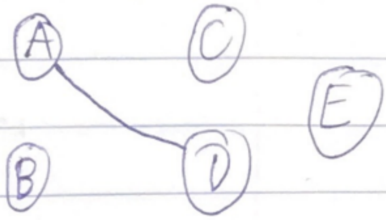
$\rightarrow \overline{G_1}$:



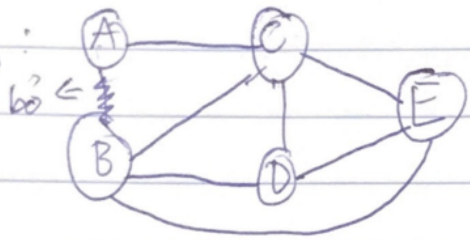
G_2:



$\rightarrow \overline{G_2}$:

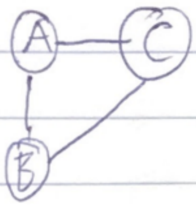


G_3:

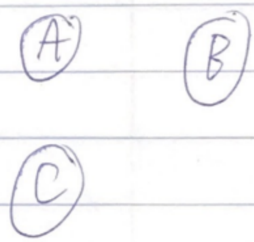


$\not\rightarrow \cancel{\overline{G_3}}$

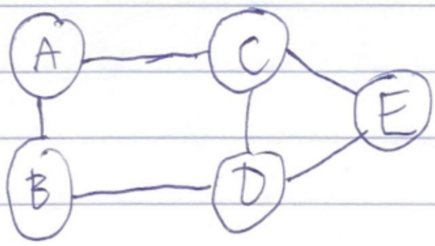
G_4:



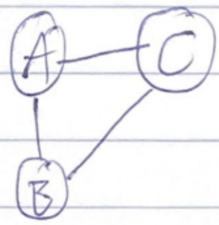
$\rightarrow \overline{G_4}$:



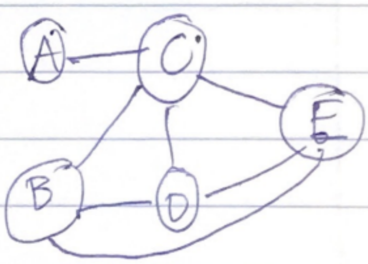
$G_1 \cap G_2 =$



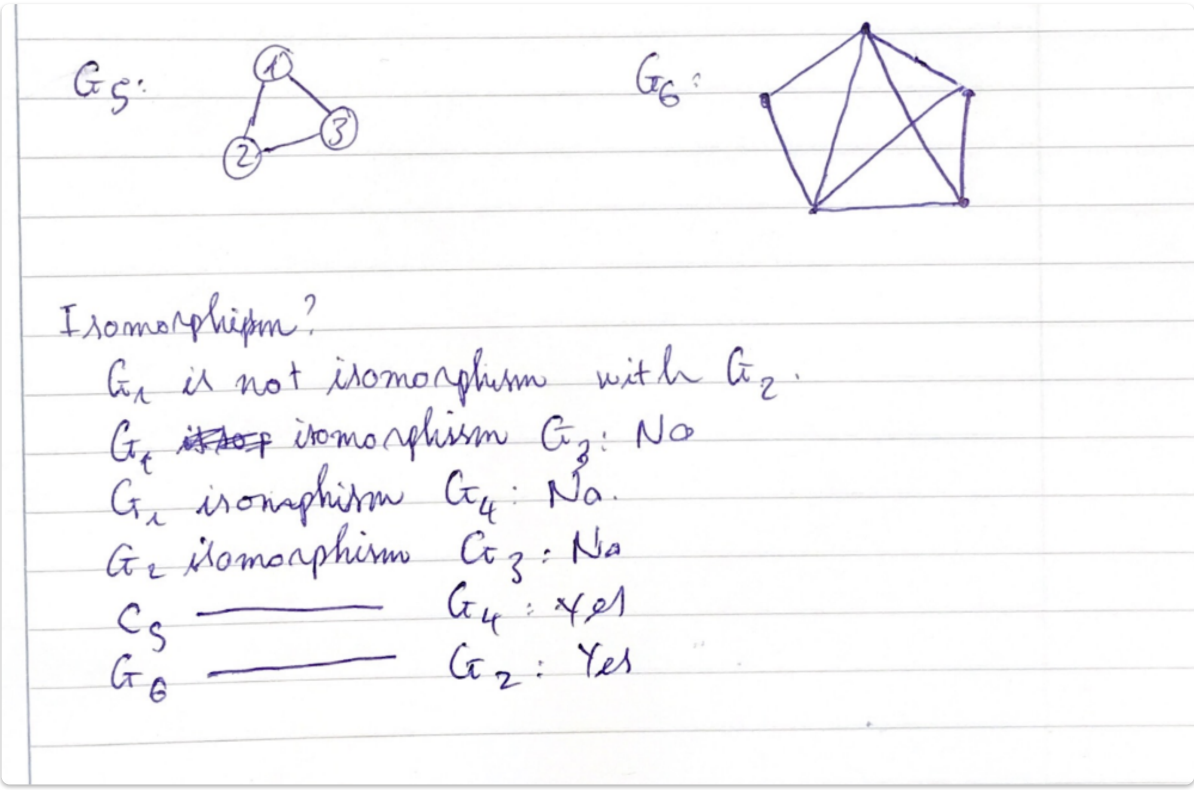
$(G_2 \cup G_1) \cap G_4 =$

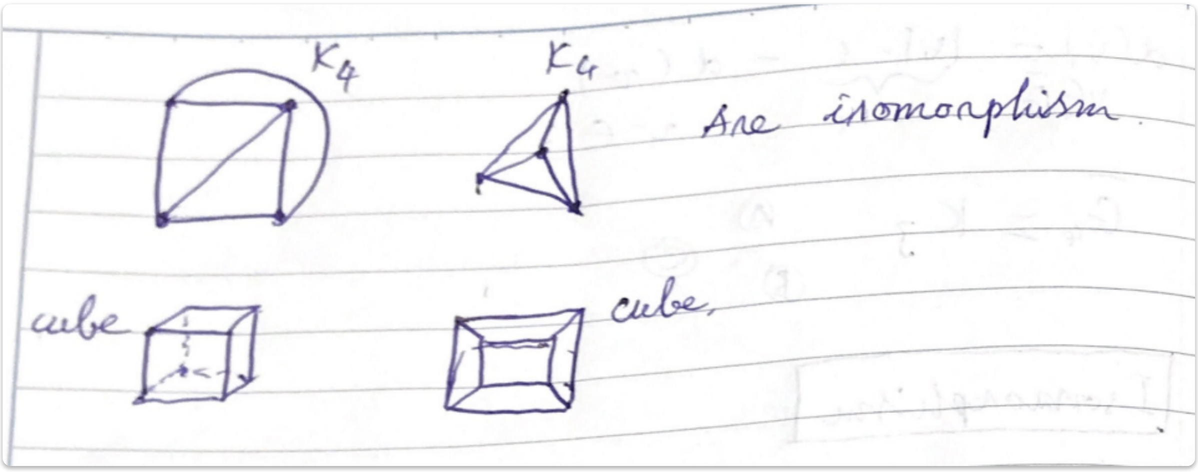


$(G_2 \cap G_3) \cup \overline{G_4} =$

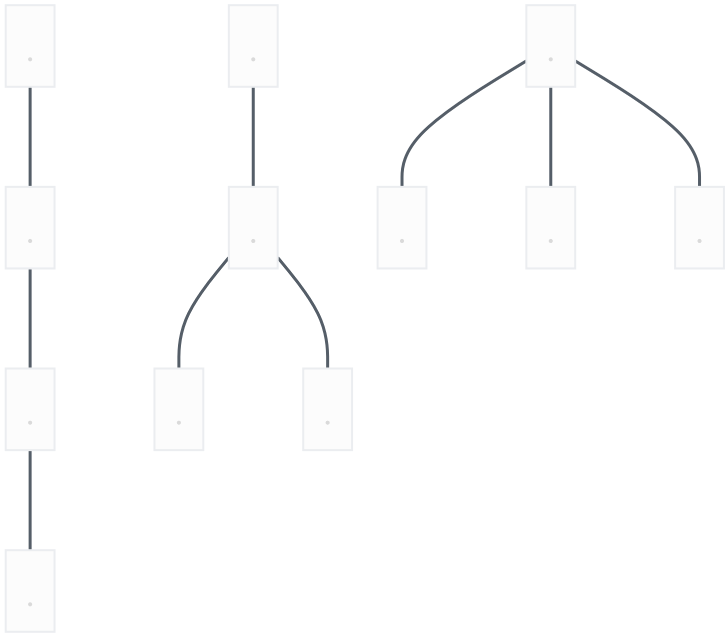


Isomorphism

- Two graphs G_1 and G_2 are *isomorphic* if there is a one-one correspondence between the vertices of G_1 and those of G_2 such that the number of edges joining two vertices of G_1 is equal to the number of edges joining the corresponding vertices of G_2
- Example:
  - 

  - 

- Ex: Draw all no isomorphic *trees* with 5 nodes (connected graph without cycles)

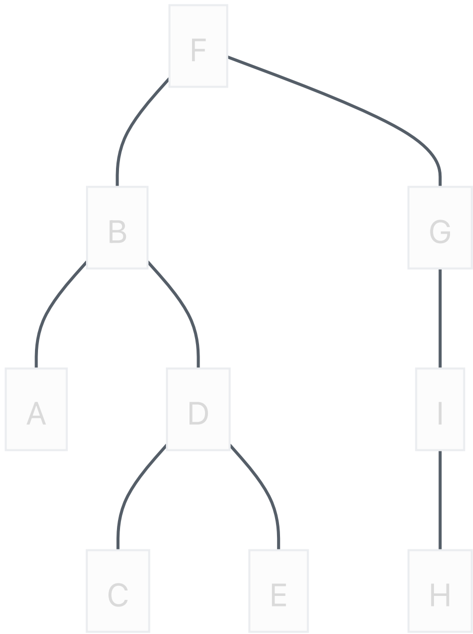


Algorithm

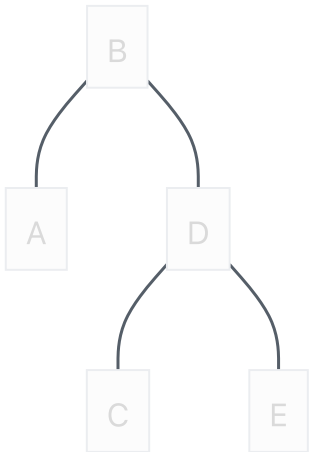
- Explain:
  - For *visiting* each node in a tree
    - Pre-order
    - Post-order
    - In-order (binary tree)

Pre-order

- Guide:
  - Left to right
  - Print root
  - Go down
- Orders of tree ($preorder(T)$ algorithm)
  1. Print root r
  2. Let $x_0, \ldots, x_n$ be all the children of the root r, in *left-to-right* order
  3. While ($i \le n$) do
     1. Set T_i be the subtree whose root is x_i
     2. Run $preorder(T)$
     3. Increment i
- Example 1:



1. root $r = F$
2. $\{B; G\}$ are the children of the root r
3. While you don't visit all the children $\{B, G\}$
    1. T_1 with root B:

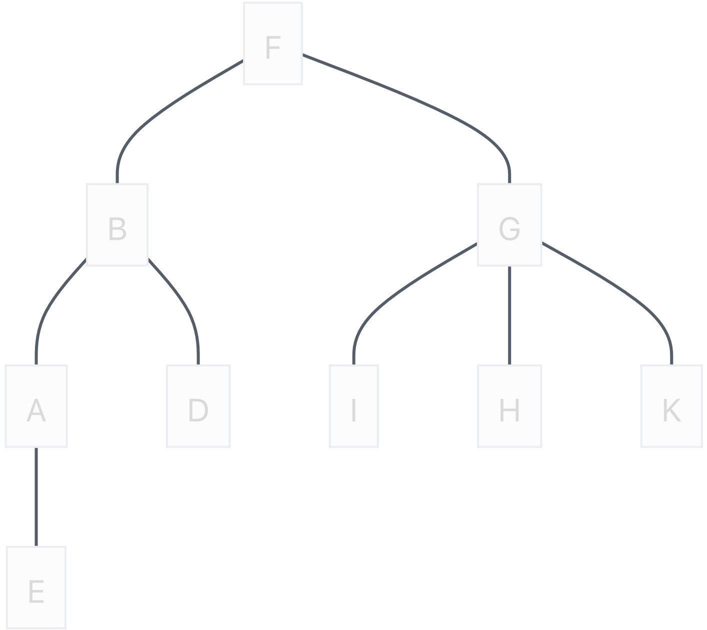
       

       - Run the $preorder(T)$ algorithm, we have:
         - Root $T_1 = \{F, B, A, D, C, E\}$
    2. T_2 with root G:

       - Run the $preorder(T)$ algorithm, we have:
         - Root $T_2 = \{G, I, H\}$

3. Totally, we have: $T = \{F, B, A, D, C, E, G, I, H\}$

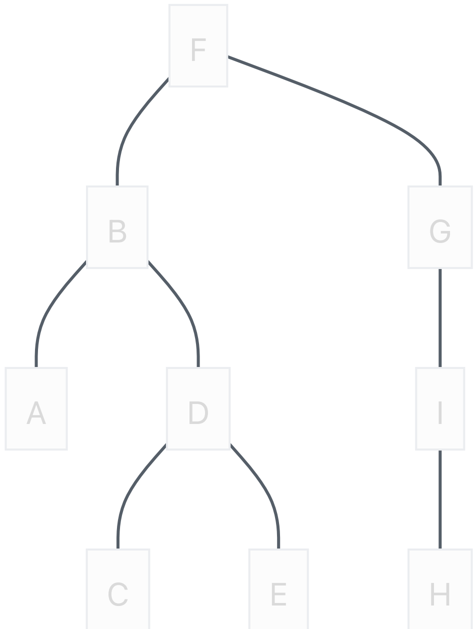
- Example 2:



  - $T = \{F, B, A, D, E, G, I, H, K\}$

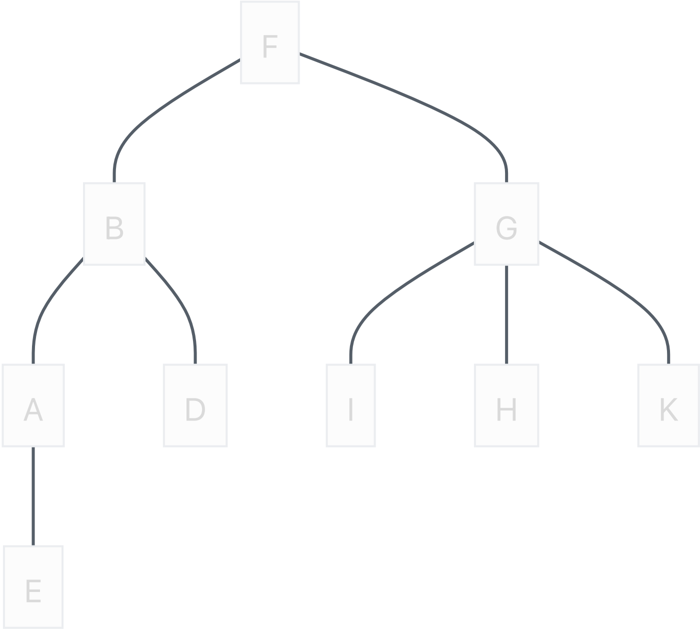
Post-order

- Explain:
  - Orders of tree ($postorder(T)$ algorithm)
    1. Let $x_0, \ldots, x_n$ be all the children of the root r, in *left-to-right* order.
    2. While $(i \leq n)$ do
       1. Set T_i be the subtree whose root is x_i
       2. Run $postorder(T_i)$;
       3. Increment i
    3. Print root r
- Example 1:



  - $T = \{A, C, E, D, B, H, I, G, F\}$

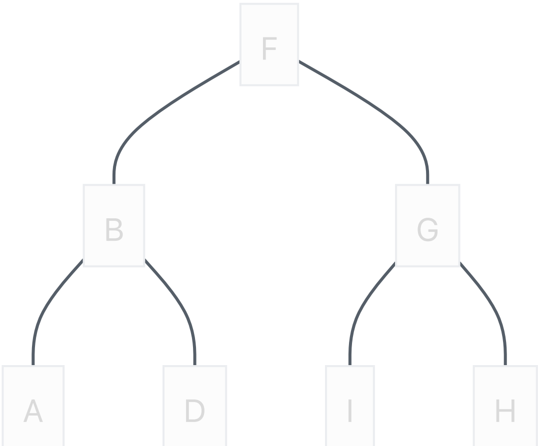
- Example 2:



  - $T = \{E, A, D, B, I, H, K, G, F\}$

In-order (for binary trees)

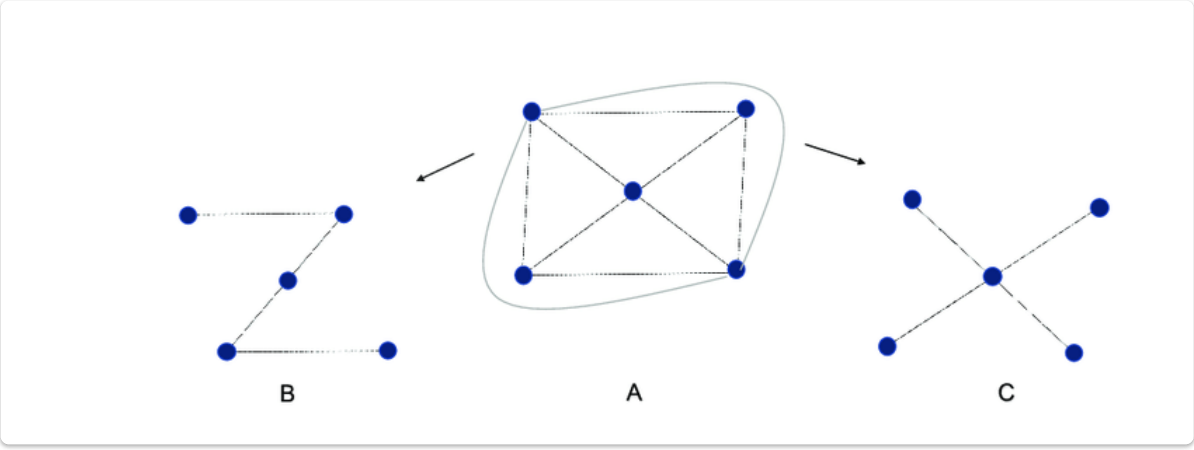
- Example



  - $T = \{A, B, D, F, I, G, H\}$

Spanning Tree

- Given $G = \{V, E\}$
  - T is a spanning tree of G if $T = \{V_1, E_1\}$ with
    - $V_1 = V$
    - $E_1 \subseteq E$
    - T is a tree (connected)
- How many non-isophormic spanning tree has K_5

- 

- $\exists$ Every connected G has a spanning tree (Always $\exists$)
- Unique? No

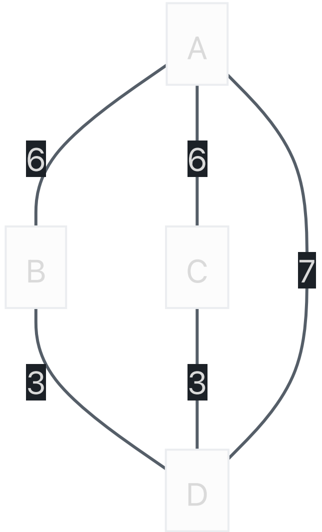
Weighted graphs

- A weighted graph $G = \{V, E\}$ is graph with associated weights in the edges, we denote the weight of an edge $w(e)$
  - weight is *the value of edge*
- A weighted tree $T = \{G_1, E_1\}$ is tree with asssociated weights in the edges.
- We say that a tree T is a *minimum spanning tree* for a weighted graph G if the following conditions are held:
  - T is spanning tree of G_1
  - It is satisfies that $\sum_{e \in E} w(e)$ is the minimal one.

A minimum $spanningTree(G)$ algorithm

$Prim(G, v)$

- Guide
  - Initialize $V_1 = \{v\}, E_1 = \{\}$ and set $G_1 = \{V_1, E_1\}$
  - While (exists an edge in V_1 that connects a vertex in V_1 to a vertex not in V_1)
    - Find the *minimal* $w(e)$ with $e = \{u, w\}$ such that $u \in V_1, w \notin V_1$)
    - Set $V_1 = V_1 \bigcup \{w\}$,
    - Set $E_1 = E_1 \bigcup \{e\}$
    - $G_1 = (V_1, E_1)$
  - Output G_1
- Example 1:

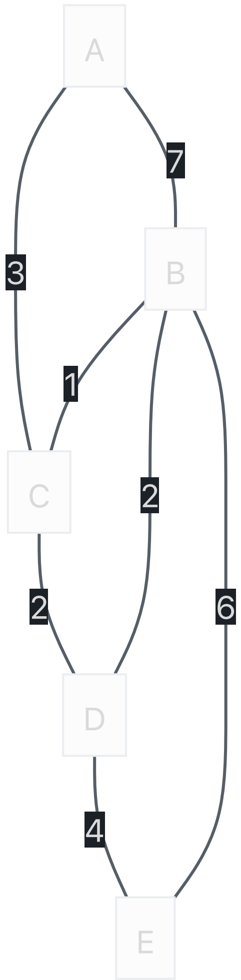


- $Prim$: $T = \{V_1, E_1\}$ minimal spanning tree of G
- $V_1 = \{C\}, V \setminus V_1 = \{A, B, D\}$

1. Select edge $\{\mu_1, \omega_1\}$ with $\mu \in V_1$, $\omega \notin V_1$
    - And $w(\{\mu_1, \omega_1\})$ is minimal
    - Options:
        - $\{C; A\} - 6$ -> No
        - $\{C; B\} - 1$ -> Yes - Select minimal
    - Update:
        - $V_1 = \{C, B\}$
        - $E_1 = \{\{C, B\}\}$
2. Select edge $\{\mu_2, \omega_2\}$ with $\mu \in V_1$, $\omega \notin V_1$
    - And $w(\{\mu_1, \omega_1\})$ is minimal
    - Options:
        - $\{C; A\} - 6$ -> No
        - $\{B, A\} - 5$ -> No
        - $\{B, D\} - 3$ -> Yes - Select minimal
    - Update:
        - $V_1 = \{C, B, D\}$
        - $E_1 = \{\{C, B\}, \{B, D\}\}$
3. Select edge $\{\mu_3, \omega_3\}$ with $\mu \in V_1$, $\omega \notin V_1$
    - And $w(\{\mu_3, \omega_3\})$ is minimal
    - Options:
        - $\{C; A\} - 6$ -> No
        - $\{B, A\} - 5$ -> No
        - $\{A, D\} - 7$ -> Yes - Select minimal
    - Update:
        - $V_1 = \{A, B, C, D\}$
        - $E_1 = \{\{B, A\}, \{C, B\}, \{B, D\}\}$

- $\sum w(e) = 9$

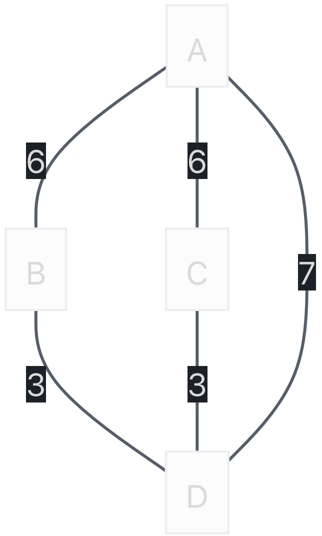
- Example 2:



- $Prim$: $T = \{V_1, E_1\}$ minimal spanning tree of G
- $V_1 = \{A\}, E_1 = \{\}, V \setminus V_1 = \{B, C, D, E\}$
- $\{A, C\}$ -> $V_1 = \{A, C\}, E_1 = \{\{A, C\}\}$
- $\{C, B\}$ -> $V_1 = \{A, B, C\}, E_1 = \{\{A, C\}, \{C, B\}\}$
- If you have two $=$ options select random
- $\{B, D\}$ -> $V_1 = \{A, B, C, D\}, E_1 = \{\{B, D\}, \{C, B\}, \{A, C\}\}$
- $\{D, E\}$ -> $V_1 = V, E_1 = \{\{B, D\}, \{C, B\}, \{A, C\}, \{D, E\}\}$
- $\sum w(e) = 2 + 1 + 3 + 4 = 10$

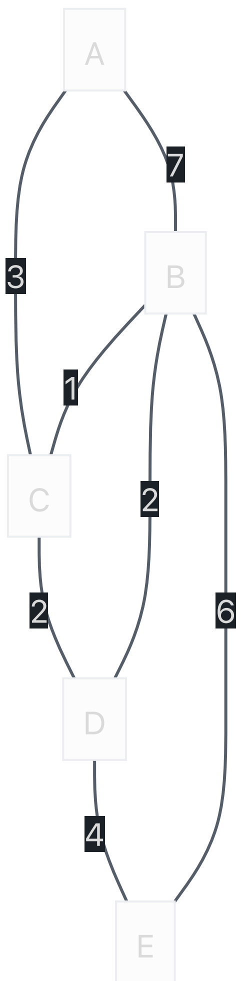
$Kruskal(G, v)$

- Guide
    -
- Example 1:



- $E_1 = \{\}$
- $\{B, C\}$ --update--> $E_1 = \{\{B, C\}\}$
- $\{B, D\}$ --update--> $E_1 = \{\{B, D\}, \{B, C\}\}$
- $\{A, B\}$ --update--> $E_1 = \{\{A, B\}, \{B, D\}, \{B, C\}\}$
- $\sum w(e) = 9$

- Example 2:



- $E_1 = \{\}$
- $\{B, C\}$ --update--> $E_1 = \{\{B, C\}\}$
- $\{C, D\}$ --update--> $E_1 = \{\{B, D\}, \{C, D\}\}$
- $\{A, C\}$ --update--> $E_1 = \{\{B, D\}, \{C, D\}, \{A, C\}\}$

- $\{D, E\}$ --update--> $E_1 = \{\{D, E\}, \{B, D\}, \{C, D\}, \{A, C\}\}$
- $\sum w(e) = 10$

$Dijksta(G, s)$

- Guide

  1. Initialize $d(s, s) = 0, d(s, v) = \infty, \forall v \neq s$. Also, set $F = \{s\}$ and $R = V \backslash F$
  2. While ($F \neq V$) do:
     1. Select $v \in R$ with the smallest path-distance $d(s, v)$
     2. Extend $F : F = F \bigcup \{v\}$
     3. Relaxation: we update the distance of edge (v, u) outgoing from v as follows:
        - if $d(s, u) > d(s, v) + d(v, u)$ then set $d(s, u) = d(s, v) + d(v, u)$
     4. Remove v from R

- Note:

  - Shortest path: sum of weights in the path is minimal
    - From s (node) to ν (other node), for all ν

- Example:
  `mermaid graph TD; id1[A]; id2[B]; id3[C]; id4[D]; id5[E]; id1--- |3|id3; id1--- |7|id2; id2--- |1|id3; id3--- |2|id4; id2--- |2|id4; id2--- |6|id5; id4--- |4|id5;`
  - It ∃ way of following the pseudo-code using a table
		iteration					
Node	Distance to A	Dist A	Dist B	Dist C	Dist D	Dist E	
A (Start)	0	0	0	0			
B	∞	4C	4C	4C			
C	∞	3A	3A	3A			
D	∞	5C	5C	5C			
E	∞	∞	10B	9D			

-
  - $d(A, A) = 0, d(s, \nu) = \infty, s \neq \nu$
    - $F = \{A\}, R = \{B, C, D, E\}$
    - Select $C, F = \{A, C\}, R = \{B, D, E\}$
    - Select a node in $\{B, D, E\}$ with small distance $d(A, node)$
  - Relaxation
    - (Cv, ν):
      - $S = A$
      - $d(A, A) = 0$
      - $d(A, B) = d(A, C) + d(C, B) = 3 + 1 = 4$
      - $d(A, C) = 3$
      - $d(A, D) = d(A, C) + d(C, D) = 3 + 2 = 5$

        - $d(A, E) = d(A, C) + d(C, E) = 3 + \infty = \infty$
    - Select a node in B, D, E with small distance $d(A, node)$
    - Select B:
        - $F = \{A, C, B\}, R = \{D, E\}$
        - $d(A, E) = d(A, B) + d(B, E) = 4 + 6 = 10$
        - $d(A, D) = d(A, B) + d(B, D) = 4 + 2 = 6$
    - Select a node in $R = \{D, E\}$
        - $$\left.\begin{array}{r} d(A, D) = 5 \\ d(A, E) = 10 \end{array}\right\} \Rightarrow \text{Select D}$$
        - extend F : $F = \{A, B, C, D\}, R = \{E\}$

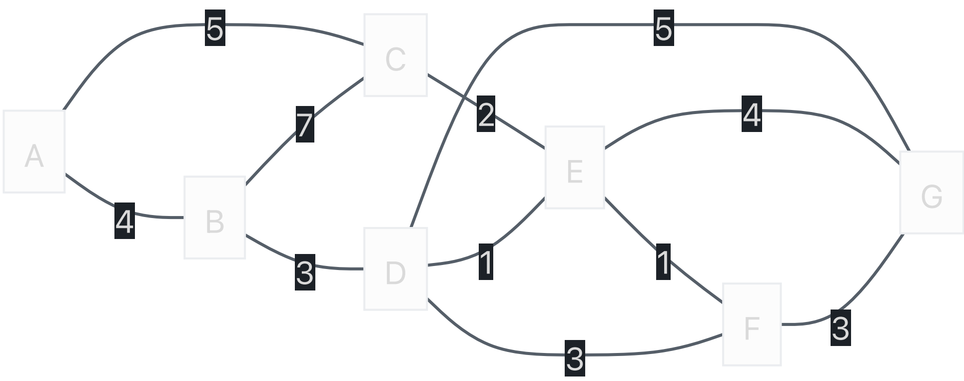
- Example 2: FInd the shortest $dist(A, \nu), \nu \in V$
  ```
  mermaid graph TD; id1[A]; id2[B]; id3[C]; id4[D]; id5[E]; id6[F]; id1--- |2|id2; id1--- |1|id3; id2--- |3|id4; id3--- |2|id5; id1--- |7|id5; id4--- |2|id6; id5--- |1|id6; id4--- |1|id5;
  ```
  - Let $G = \{V, E\}, V = \{A, B, C, D, E\}$
  - $F = \{A\}$
  - $R = \{B, C, D, E, F\}$

	dist	check adjunction with A			
A	0	0	0	0	0
B	∞	2_A	2_A	2_A	2_A
C	∞	1_A	1_A	1_A	1_A
D	∞	∞	∞	5_B	5_B
E	∞	7_A	3_C	3_C	3_C
F	∞	∞	∞	∞	4_C

-
    - iteration 1: Select from $\{B, C, D, E, F\}$ smallest distance, select C
        - $F = F \bigcup \{C\}$
        - Relaxation to neighbours of C (in R):
            - $d(C, E) = 1 + 2 = 3, R = \{B, F, D, E\}$
    - Iteration 2: Select B
        - $F = \{A, C, B\}, d(A, D) = 5$
    - Iteration 3: Select E
        - $F = \{A, B, C, E\}$
        - Relaxation to neighbours of E (in R):
            - $d(A, F) = d(A, E) + d(E, F) = 3 + 1 = 4$
            - $d(A, D) = d(A, E) + d(E, D) = 3 + 1 = 4$
            - $R = \{F, D\}$
    - Iteration 4: Select between D and F. Select F
        - $F = \{A, B, C, E, F\}$
        - Relaxation to neighbours of F (in R):
            - $d(A, D) = d(A, F) + d(F, D) = 4 + 2 = 6$

- Iteration 5: $F = V$. Select D
    - Relaxation to neighbours of F (in R):
        - $dist(A, F) = 4$
- $A \to B \to C \to D$

Exercise



Prim

- Let $T_1 = \{V_1, E_1\}$ be the minimal spanning tree of G
- Let $V_1 = \{A\}, V \backslash V_1 = \{B, C, D, E, F, G\}$
- Iterator 1:
    - Select edge $\{A, B\} - 4 \to \text{Yes - Select minimal}$
    - Select edge $\{A, C\} - 5 \to \text{No}$
    - Update:
        - $V_1 = \{C, D, E, F, G\}$
        - $E_1 = \{\{A, B\}\}$
- Iterator 2:
    - Select edge $\{B, C\} - 7 \to \text{No}$
    - Select edge $\{B, D\} - 3 \to \text{Yes - Select minimal}$
    - Update:
        - $V_1 = \{C, E, F, G\}$
        - $E_1 = \{\{A, B\}, \{B, D\}\}$
- Iterator 3:
    - Select edge $\{D, E\} - 1 \to \text{Yes - Select minimal}$
    - Select edge $\{D, F\} - 3 \to \text{No}$
    - Select edge $\{D, G\} - 5 \to \text{No}$
    - Update:
        - $V_1 = \{C, F, G\}$
        - $E_1 = \{\{A, B\}, \{B, D\}, \{D, E\}\}$
- Iterator 3:
    - Select edge $\{E, C\} - 2 \to \text{No}$
    - Select edge $\{E, F\} - 1 \to \text{Yes - Select minimal}$
    - Select edge $\{E, G\} - 4 \to \text{No}$

    - Update:
        - $V_1 = \{C, G\}$
        - $E_1 = \{\{A, B\}, \{B, D\}, \{D, E\}, \{E, F\}\}$
- Iterator 4:
    - Select edge $\{F, G\} - 3 \rightarrow \text{No}$
    - Select edge $\{E, C\} - 2 \rightarrow \text{Yes - Select minimal}$
    - Update:
        - $V_1 = \{G\}$
        - $E_1 = \{\{A, B\}, \{B, D\}, \{D, E\}, \{E, F\}, \{E, C\}\}$
- Iterator 5:
    - Select edge $\{F, G\} - 3 \rightarrow \text{Select minimal}$
    - Update:
        - $V_1 = \{V\}$
        - $E_1 = \{\{A, B\}, \{B, D\}, \{D, E\}, \{E, C\}, \{E, F\}, \{F, G\}\}$
- $\text{Total cost} = 4 + 3 + 1 + 2 + 1 + 3 = 14$

$Kruskal$

- Iterator 1:
    -