Optional Homework Due April 30

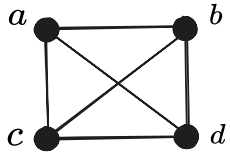
Show your work. Answer without work receives no credit.

Graphs

Problem 1

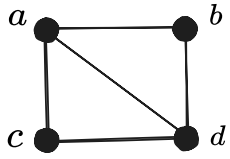
Draw a (finite, simple) graph G with the following properties, or explain why this task is impossible.
(A) G has four vertices, each having degree 3.
(B) G has 13 vertices, of which 5 vertices are of degree 3 and 8 vertices are of degree 4.
(C) G has vertex set $\{a, b, c, d\}$ with $\deg(a) = \deg(d) = 3$ and $\deg(b) = \deg(c) = 2$:
(D) G has vertex set $\{a, b, c, d\}$ with $\deg(a) = 1$, $\deg(b) = 2$, $\deg(c) = 3$, $\deg(d) = 4$.

Answer:
(A)

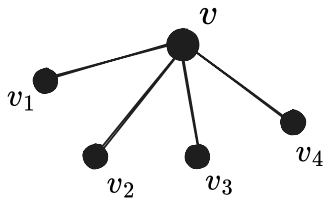


(B) This is impossible by the handshaking lemma since the sum of all degrees is $5 \cdot 3 + 8 \cdot 4 = 15 + 32 = 47$ would mean there are $47/2 = 23.5$ edges, which cannot be.

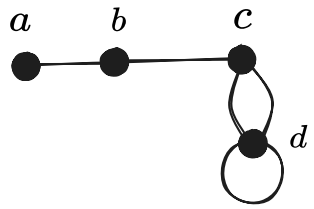
(C)



(D)
If a vertex v in a simple graph G has degree 4, then that vertex v must have four adjacent vertices v_1, v_2, v_3, v_4 so G has at least five vertices v, v_1, v_2, v_3, v_4, contrary to our assumption that G has exactly four vertices. Thus such a simple graph cannot exist.



(Note: if G were instead a **multigraph**, where we allow multiple edges between any two given vertices, as well as self-loops on a vertex itself, then a multigraph with the listed degrees does exist, as shown below.

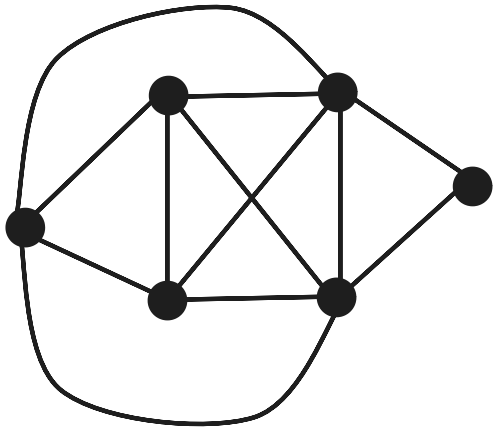


Here, the loop on d contributes $+2$ to $\deg(d)$. However, these are not graphs we study in this class.)

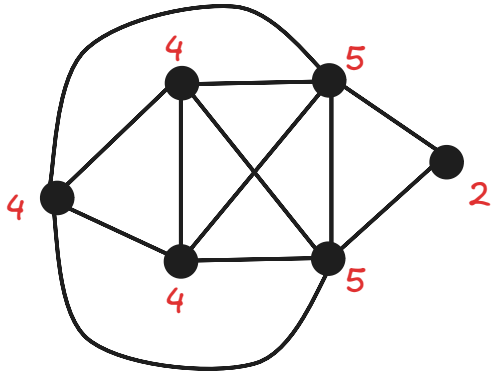
Euler walks

Problem 2

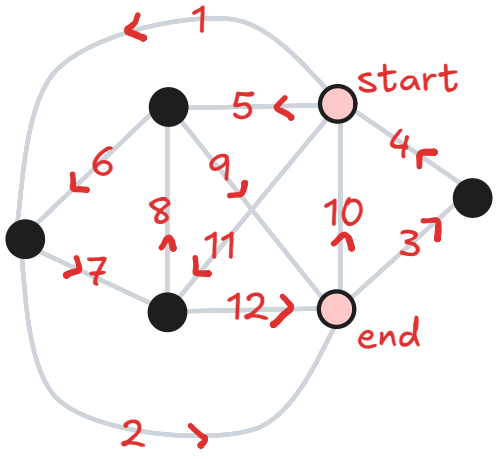
State whether an Euler walk exists for the graph below. If it exists, label the edges of the Euler walk with increasing numbers from starting vertex to the ending vertex



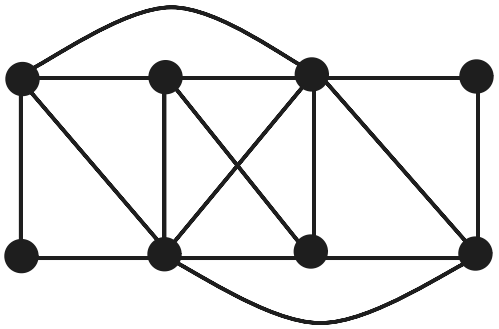
Answer.
This graph admits an Euler walk but not an Euler circuit since it has exactly two odd degree vertices:



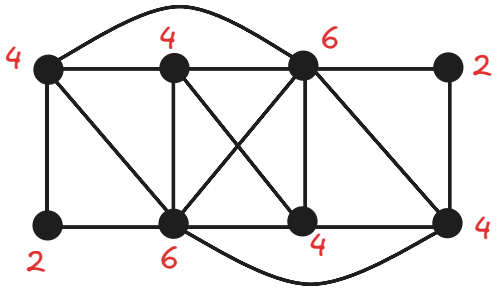
We choose one of the odd degree vertices to be the start and the other one to be the end vertex, and proceed to walk through each edge exactly once:



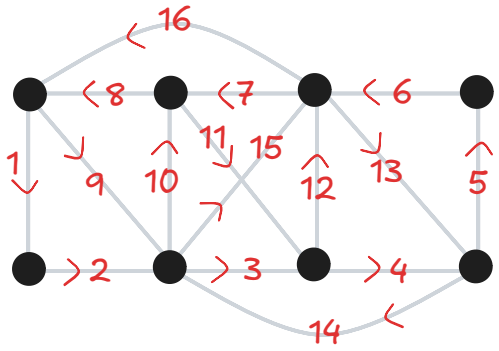
Problem 3



Answer.
We first compute the degrees of the graph:



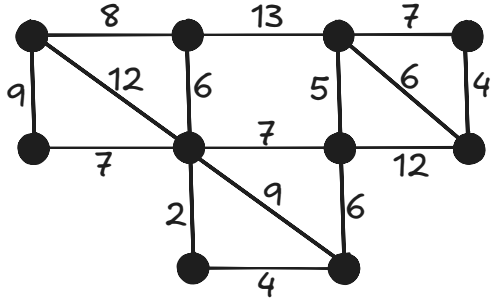
This graph has only even degrees so it admits an Euler circuit. We can pick any point to be both the start and end point, say, the upper left vertex:



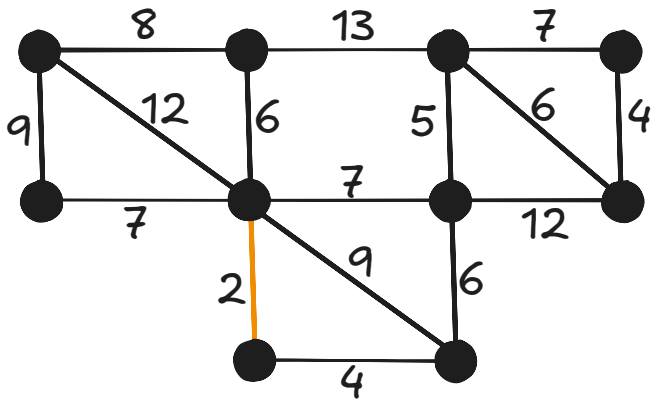
Minimum spanning trees

Problem 4

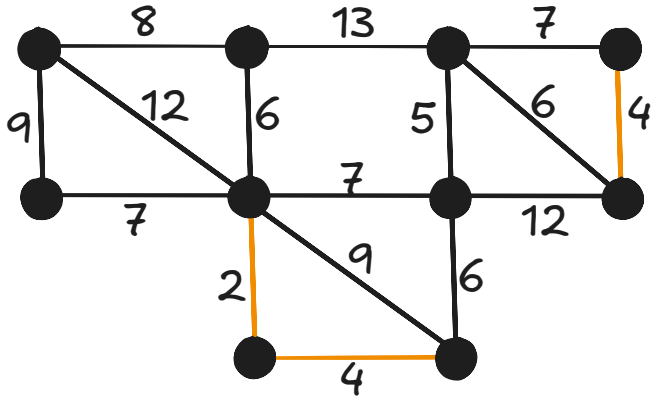
Houses are to be connected by utility lines. Due to the rocky soil, the costs of extending utility lines varies from house to house. The graph below depicts the houses as vertices, proposed excavation routes as edges, and the cost of extending utility lines through a given edge as an edge weight. Determine the best way to connect the houses onto one utility network.



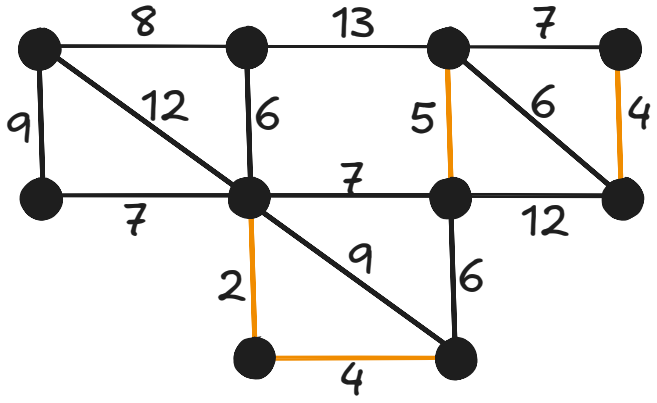
Solution:
We successively select the lowest weight edges without introducing circuits:



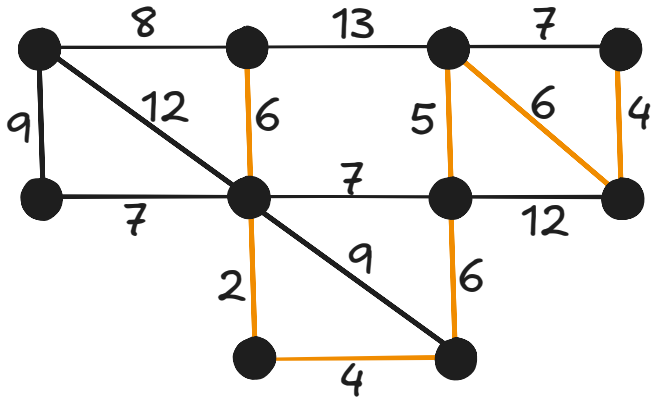
Next:



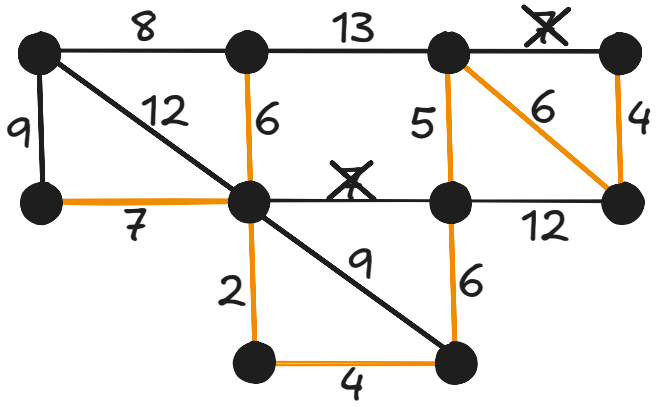
Next:



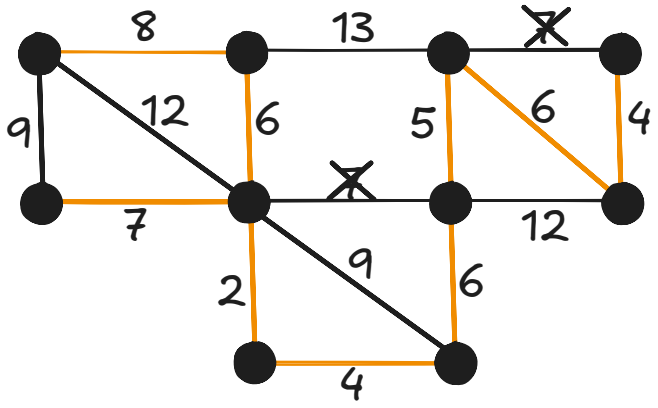
Next:



We cannot use two of the three 7's since those would introduce circuits, but we can use the left-most 7:



Next:



We have connected every vertex into one network, and so Kruskal's algorithm guarantees that the orange subgraph is a MST of the original graph. The total cost of the orange network is obtained by summing the edge weights of this subgraph:

$$2 + 4 + 4 + 5 + \ + \ + \ + 7 + 8 = 48$$

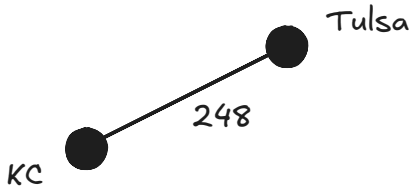
Problem 5

The table below shows the distances between Atlanta, Columbus, Kansas City, Minneapolis, Pierre, and Tulsa. Use Kruskal's algorithm to find the MST connecting the six cities.

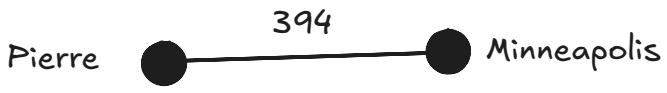
	Atlanta	Columbus	KC	Minneapolis	Pierre	Tulsa
Atlanta		533	798	1068	1361	772
Columbus	533		656	713	1071	802
KC	798	656		447	592	248
Minneapolis	1068	713	447		394	695
Pierre	1361	1071	592	394		760
Tulsa	772	802	248	695	760	

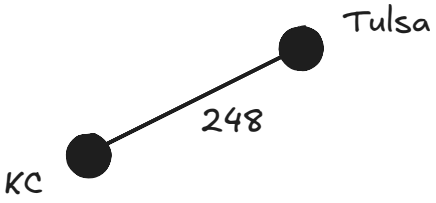
Solution:
Instead of drawing the graph out, we directly apply Kruskal's algorithm to these table entries, which are the edge weights of the graph where the cities are vertices and the edges are the distances between the cities.

First, Kruskal's algorithm finds the lowest cost edge (248 between KC and Tulsa) and adds that to our graph:

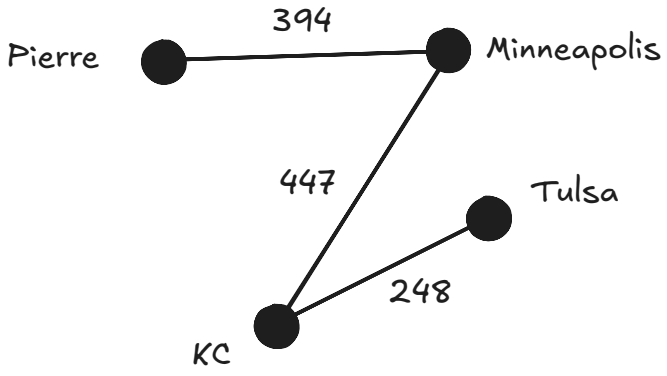


Then we add the next lowest cost edge (394 between Pierre and Minneapolis):

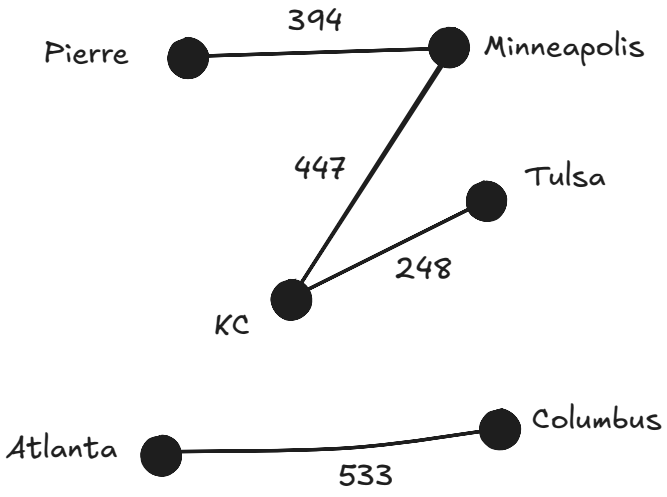




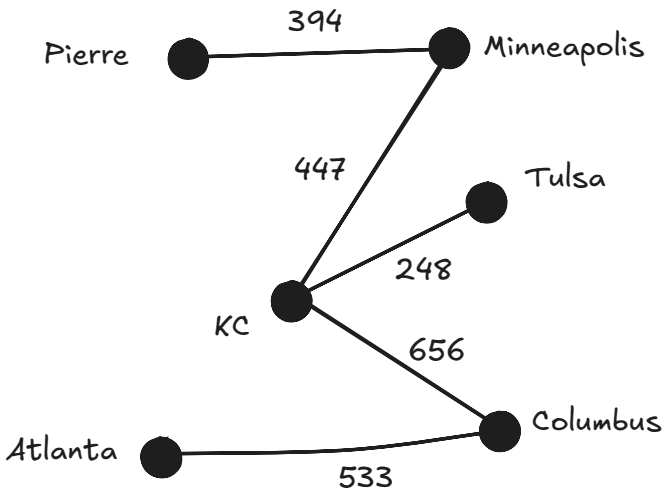
Next we add 447 between KC and Minneapolis:



Next we add 533 between Atlanta and Columbus



followed by 656 between Columbus and KC:



Since all cities are connected, Kruskal's algorithm terminates, and the resulting graph is a minimum spanning tree of the original graph.