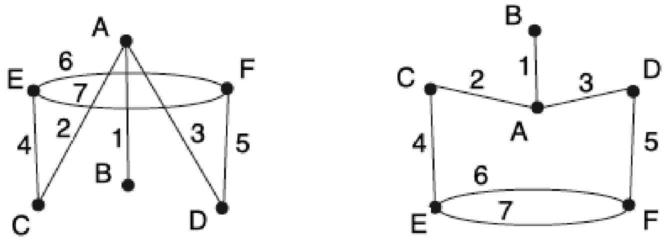
1. There are many correct answers here. For instance, $H_1 = (\{v_1, v_2\}, \{e_1, e_2, e_5\})$ is connected and $H_2 = (\{v_1, v_2, v_3\}, \{e_1, e_3, e_4\})$ is disconnected.

2. Again there are many correct answers, but your graph must contain a loop and/or parallel edges in order to be non-simple. One example is below.

3. A few different possibilities here; below is one.



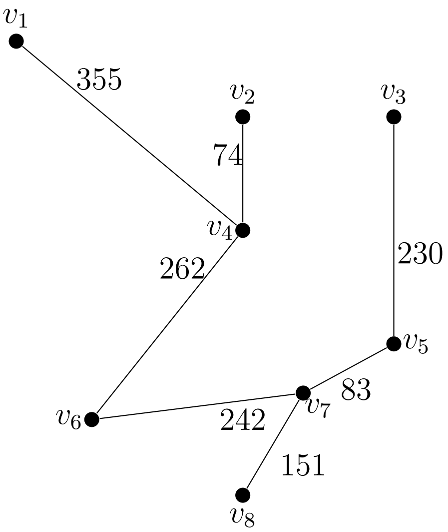
4. (a) First, how many edges does K_n have in total? Vertex v_1 is incident to all other vertices, so that's $n-1$ edges. Then v_2 is incident to all other vertices but the edge to v_1 has already been counted, so $n-2$ more edges. Then v_3, $n-3$ more edges. And so on until v_{n-1}, one more edge to v_n. The total is $1+2+3+\cdots+(n-1)$ edges, which is $\frac{n(n-1)}{2}$. The question asks us to keep all the vertices and pick an arbitrary subset of all the edges, how many such subsets exist? It is the cardinality of the power set of the set of edges, so $2^{\frac{n(n-1)}{2}}$.

   (b) There are 4: (1) 3 vertices and no edges, (2) 3 vertices and 1 edge, (3) 3 vertices and 2 edges, (4) 3 vertices and all 3 edges. Every subgraph of K_3 is isomorphic to one of these 4. (There are $2^3 = 8$ subgraphs with all 3 vertices in total.)

5. Kruskal's Algorithm gives the following table.

Edge	Weight	Will adding edge make a circuit?	Action taken	Cumulative Weight of subgraph
(v_2, v_4)	74	no	added	74
(v_5, v_7)	83	no	added	157
(v_7, v_8)	151	no	added	308
(v_3, v_5)	230	no	added	538
(v_6, v_7)	242	no	added	780
(v_4, v_6)	262	no	added	1042
(v_4, v_7)	269	yes	not added	1042
(v_3, v_7)	306	yes	not added	1042
(v_2, v_7)	348	yes	not added	1042
(v_1, v_4)	355	no	added	1397

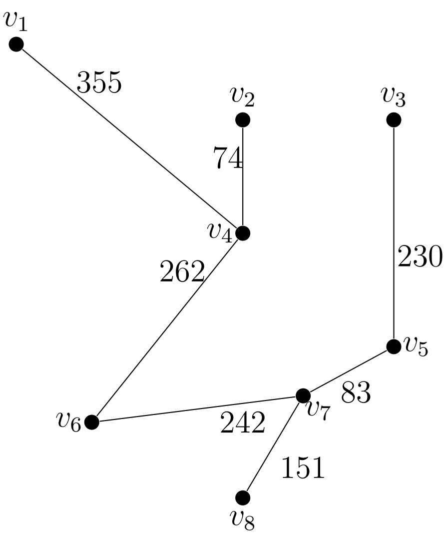
So a minimum spanning tree is the following.



Prim's Algorithm starting at v_1 gives the following table.

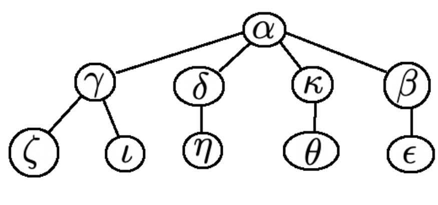
Vertex added	Edge added	Weight	Cumulative weight
v_1			
v_4	(v_1, v_4)	355	355
v_2	(v_2, v_4)	74	429
v_6	(v_4, v_6)	262	691
v_7	(v_6, v_7)	242	933
v_5	(v_5, v_7)	83	1016
v_8	(v_7, v_8)	151	1167
v_3	(v_3, v_5)	230	1397

So a minimum spanning tree is as follows.

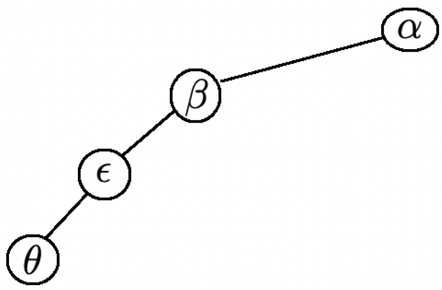


These two trees are the same, which will always happen if there are no two edges with the same weight, or if for every instance of edges with the same weight the same edge happens to be chosen by both algorithms.

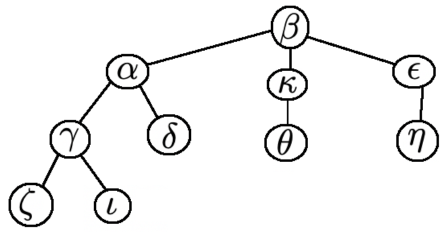
6. Below is breadth-first starting with α and inspecting adjacent nodes in left-to-right order, which is only one of the options. The inspection order is $\alpha \to \gamma \to \delta \to \kappa \to \beta \to \zeta \to \iota \to \eta \to \theta$; the only one not inspected is ϵ.



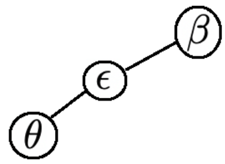
Below is depth-first starting with α, again pushing left-to-right nodes onto the stack.



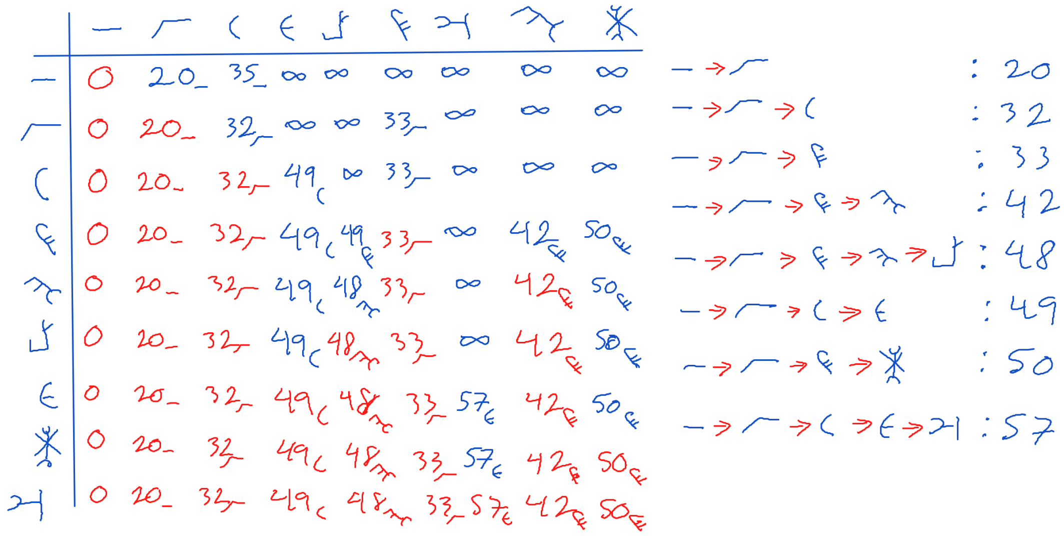
Below is breadth-first starting with β.



Below is depth-first starting with β.



7. Klingon!



8. This labelling of the nodes shows that Graphs 1, 2 and 4 are bipartite and Graph 3 is not.