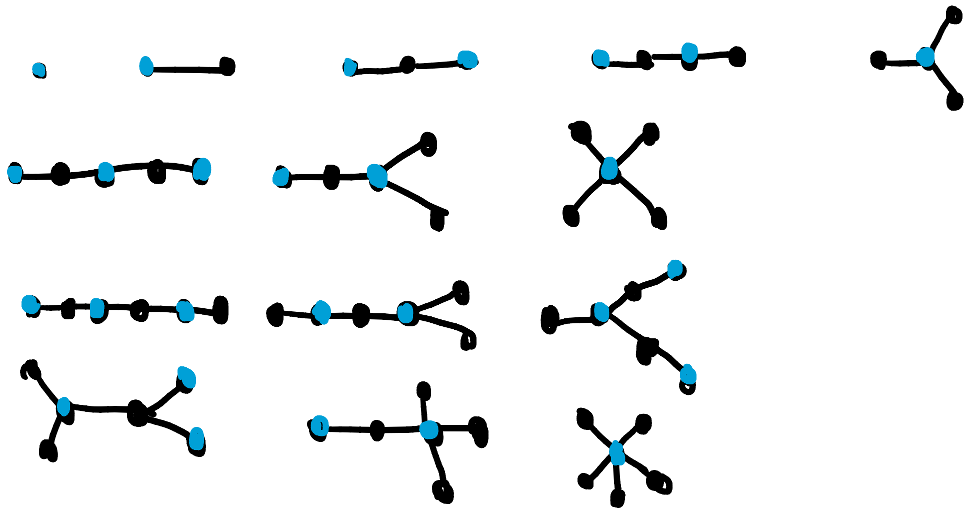
MATH 239 Exercises 5

5.1.1



5.1.2

We know that a graph is bipartite if and only if it has no odd cycles. Since a tree has no cycles at all, it must be bipartite.

Alternatively, we prove this using strong induction on the number of vertices. A tree with 1 vertex is trivially bipartite. A tree with 2 vertices is bipartite.

Assume for all k > 2, all trees on less than k vertices are bipartite. Observe a tree with k vertices. We can remove all the leaves of this tree (vertices of degree 1), which leaves us with a smaller tree. This tree must be bipartite - let the bipartition be X, Y. We add back each leaf. For each leaf, if the node it is connected to is in bipartition Y, we can place it in partition X, and vice versa. Hence the tree on k vertices is bipartite.

5.1.3

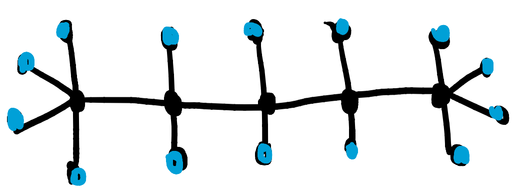
A tree with n vertices has n-1 edges. We have that

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

$$2n-2 = 3\cdot 4 + 2\cdot 5 + 1(n-5)$$

$$n = 17$$

Hence the minimum number of vertices of degree 1 is 14.



5.1.4

A tree with r vertices has r-1 edges. Let n_k be the number of vertices of degree k.

$$r = \sum_k n_k$$

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

$$2r-2 = \sum_k k n_k$$

$$-2 + 2\sum_k n_k = \sum_k k n_k$$

$$0 = 2 + \sum_k (k-2)n_k = 2 - n_1 + 0n_2 + n_3 + 2n_4 + 3n_5 + 4n_6 + \sum_{k\geq 7}(k-2)n_k$$

$$= 2 - n_1 + 0 + 2 + 2 + 3n_5 + 8 + \sum_{k\geq 7}(k-2)n_k$$

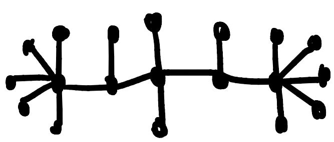
$$= 14 - n_1 + 3n_5 + \sum_{k\geq 7}(k-2)n_k$$

Hence we have

$$n_1 = 14 + 3n_5 + \sum_{k\geq 7}(k-2)n_k \geq 14$$

$$r \geq n_1 + n_3 + n_4 + n_6 \geq 14 + 2 + 1 + 2 = 19$$

There must be at least 19 vertices.



5.1.5

Let p be the number of vertices in the tree. Since the tree is cubic, there are (k + 3(p-k))/2 edges. Since it is a tree, it must have p-1 edges. Hence we have

$$p-1 = \frac{k+3(p-k)}{2}$$

$$2p-2 = k+3p-3k$$

$$p = 2k-2 = 2(k-1)$$

5.1.6

Let G be a forest. Since G has no cycles, none of its components can have cycles. Hence every component is a tree. Let c be the number of components in G. For each component T_i where 1 <= i <= c, we have that

$$q = \sum_{i=1}^{c} |E(T_i)| = \sum_{i=1}^{c} (|V(T_i)|-1) = -c + \sum_{i=1}^{c} |V(T_i)| = p-c$$

Hence we have that there are p-q components.

5.1.7 We have that a tree with n vertices has n-1 edges. Hence if (d1,…dp) is a degree sequence of a tree on p vertices, then we have

$$\sum_{i=1}^{p} d_i = \sum_{v \in V} \deg(v) = 2|E(T)| = 2p-2$$

We prove that if the sum of (d1…dp)=2p-2, then this sequence is the degree sequence of a tree on p vertices using induction.

Our base case is p=2. The degree sequence (1, 1) is that of the 2 vertex tree. We have 1+1 = 2(2)-2.

Assume that for all p > 2, the statement holds for all k < p. Since the sum of (d1,…dp) = 2p-2, and are all positive integers, we must have that at least one of them, di, is equal to 1. Let this element be d1 without loss of generality. We also have that at least one element is greater than or equal to 2. Let this element be d2, without loss of generality. Removing d1, and subtracting one from d2, we obtain

$$\sum_{i=2}^{p} d_i = 2p-2-1-1 = 2(p-1)-2$$

Hence this is the degree sequence of a tree on (p-1) vertices. We can add d1 to this sequence, and add this vertex as a leaf to vertex d2 to obtain a tree.

5.3.1 **a)** Suppose that there exists an edge uv of T where d(u) = d(v). Let P1 be the path from u to T. P1 cannot contain edge uv, otherwise d(u) = d(v)+1 != d(v), and vice versa for P2, the path from v to T. Then we have that P1uP2 is a cycle, starting and ending at u. But this contradicts our statement that T is a tree, and so for each edge uv, we have d(u) != d(v).

b) If x is adjacent to r, we have that d(r) = 0 < 1 = d(x). If x is not adjacent to r, we must have a path P from x to r, which contains 3 or more vertices. There must only be one element on this path that is adjacent to x, otherwise there would be a cycle. Let this element by called y. Hence this element has a path with has a length one less than P to r, and we have d(y) < d(x).

5.3.2 By definition of d(v), G must be connected.

Suppose there exists an odd cycle in G. Let the x be the vertex such that d(x) < d(y) for every vertex y in the cycle. Since the cycle is odd, there must exist two vertices u and v that both have the same longest distance from x in the cycle. But then we have d(u) = d(v), which contradicts our statements.

Suppose there exists an even cycle in G. Let the x be the vertex such that d(x) < d(y) for every vertex y in the cycle. Since the cycle is even, there must exist one vertex y such that it has the longest distance from x in the cycle. But then y is adjacent to two vertices u and v where d(u) = d(v) < d(y), which contradicts our statements.

Hence G cannot contain a cycle, and so must be a tree.

5.4.1

Let r be the initial vertex. Since the algorithm has terminated, no edge joins a vertex u in D to a vertex v not in D. Hence D contains all vertices in G that are connected to r. Since D contains fewer than p vertices, it must be a component of G.

5.4.2

Nope sorry :P

5.4.3

1, 2, 13, 14, 4, 5, 6, 10, 11, 12, 3, 7, 9, 8

5.4.4

A graph where all vertices are adjacent to the root.

5.4.5

Vertices do not enter the tree in order of level - vertex b enters after vertex c.

5.4.6

Let w = pr(v). We have that level(w)+1 = level(v) = level(u)+1. Hence w is at the same level as u. Since level(v) > level(u), u must become active before v. Since u and v are joined by a non-tree edge, v must be in the tree when u first is active (otherwise uv would be a tree edge). Hence it must have been added by w, and hence w must be active before u.

5.5.1

We can construct a breadth first search tree on G, terminating when we detect an edge that would create an odd cycle. At each stage, we check if the current vertex contains an edge joining it to a vertex already in the tree, at the same level - if there is, that would be an odd cycle.

5.5.2

Since G has diameter 3, there are no vertices of greater than distance 3 from v. There must be a path from v to each of these vertices of length 3. We can construct a spanning tree containing each of these vertices and the edges of each of these paths. We terminate the search as soon as we reach each of these 20 vertices, so each of these vertices has degree 1.

Note: 20 vertices at distance 3 from v = distance of 6 between each of these vertices = diameter of 6??

5.5.3

A) We can use breadth first search to compute the longest distance for every node in the graph.

B) Breadth first search will find the length of the shortest path from every node v to u - we can take the maximum of the longest of these paths to find the diameter of the tree.

5.5.4 There are m vertices with a shortest path of n from v. We can create a spanning tree containing each of these paths. Since the path terminates at each of these m vertices and there is no shorter path from them to v, they will have a degree of 1 in the spanning tree.

5.5.5 Every cycle in the tree must contain at least one non-tree edge. Since every non-tree edge joins two vertices u and v at equal levels, then the paths from u and v to the root first meet at a vertex, which is m levels less than u and v, for some m >= 1. Then the path in the tree between u and v has length 2m, and so the cycle must contain an even number of tree edges.