Prob 1.

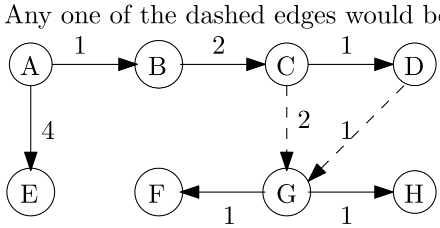
1. $n = \Theta(10n)$
2. $0.1n = \omega(10 \log n)$
3. $2^n = \omega(n^3)$
4. $2n \log n = o(n^2)$

Prob 2.

1.

Iterations	0	1	2	3	4	5	6	7	8
Vertex	A	B	C	D	E	G	F	H	
$dist(A)$	0	0	0	0	0	0	0	0	0
$dist(B)$	∞	1	1	1	1	1	1	1	1
$dist(C)$	∞	∞	3	3	3	3	3	3	3
$dist(D)$	∞	∞	∞	4	4	4	4	4	4
$dist(E)$	∞	4	4	4	4	4	4	4	4
$dist(F)$	∞	8	7	7	7	7	6	6	6
$dist(G)$	∞	∞	7	5	5	5	5	5	5
$dist(H)$	∞	∞	∞	∞	8	8	6	6	6

2. Any one of the dashed edges would be okay.



Prob 3.

1. This property is correct.

   For any cycle $C = (v_1, v_2, \ldots, v_n)$, WLOG assume that edge (v_1, v_n) has the largest weight. Then I claim that for any spanning tree T contains $e = (v_1, v_n)$, we can find another edge $e' = (v_i, v_{i+1})$ from the cycle, so that by replacing e with e' one can obtain another spanning tree with smaller total weight. And this claim directly implies the result.

   After removing edge (v_1, v_n) from T, it will break into two connected components. Because v_1 and v_n belong to different connected components, there must exists $1 \le i < n$ such that v_i and v_{i+1} belong to different connected components. By adding this edge (v_i, v_{i+1}) back clearly the graph is connected again and has $|V| - 1$ edges. Thus it is again a spanning tree. Since $w(v_1, v_n)$ is larger than $(w(v_i, v_{i+1}))$, the new spanning tree has smaller total weight.

2. Correct.

   By previous question, for any cycle, the largest weighted edge cannot belong to any minimum spanning tree. So removing them does not affect the answer. And an undirected graph is a tree if and only if it contains no cycles. Thus this algorithm works correctly.

Prob 4.

1. Because G is 3-colorable. Take any of it's 3-coloring $c_3 : V \to \{1, 2, 3\}$. Construct a 2-coloring $c_2 : v \to \{1, 2\}$ in the following

way: for any $v \in V$, if $c_3(v) \in \{1, 2\}$, then $c_2(v) = c_3(v)$, otherwise arbitrarily assign a color for $c_2(v)$. Easy to check this 2-coloring does not generate monochromatic triangle.

2. Yes.

   Fix any 3-coloring c_3 of G. Denote the algorithm's current 2-coloring by c_2. Let err be the number of vertices where $c_3(v) = 1$ but $c_2(v) = 2$ or $c_3(v) = 2$ but $c_2(v) = 1$. We'll get a satisfying result once err becomes 0.

   Each time if we find a monochromatic triangle (v_1, v_2, v_3), WLOG assume that $c_3(v_1) = 1, c_3(v_2) = 2, c_3(v_3) = 3$ and $c_2(v_1) = c_2(v_2) = c_2(v_3) = 1$. With probability $1/3$ we flip the color of v_1 and increase err by 1. With probability $1/3$ we flip the color of v_2 and decrease err by 1. With probability $1/3$ we flip the color of v_3 and keep err unchanged.

   In expectation it's the same as after 3 moves we'll change the value of err. And increase/decrease err by 1 with probability one half. Be reminded that the hitting time in line graph is $O(n^2)$. Thus implies the result.