Prob 1. (a) $5n = \Theta(100n)$

(b) $0.1n = \omega(100 \log n)$

(c) $4^n = \omega(10^6 n^3)$

(d) $2n^2 \log n = \omega(n^2)$

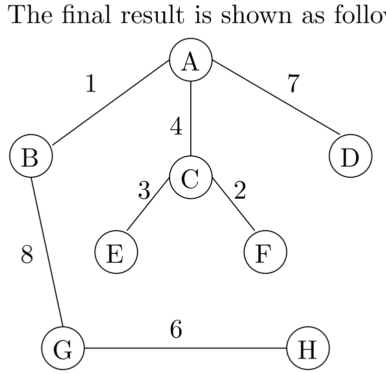
(e) $\log n = \text{small-}o(n \log n)$

Prob 2. One can solve this problem by slightly modifying the BFS algorithm.

Data: the graph G and two of its vertices s and t
Result: the number of shortest path between s and t

```
initialize: dist(s) = 0; dist(u) = ∞ for all other vertex u;
initialize: count(s) = 1; count(u) = 0 for all other vertex u;
Q ← queue with one element s ;
while Q is not empty do
    u ← dequeue(Q);
    for each neighbor v of u do
        if dist(v) > dist(u) + 1 then
            if dist(v) = ∞ then
                enqueue(Q, v);
            end
            dist(v) ← dist(u) + 1;
            count(v) ← count(u);
        else
            if dist(v) = dist(u) + 1 then
                count(v) ← count(v) + count(u);
            end
        end
    end
end
Output count(t);
```

Prob 3. (a) The final result is shown as follow:



(b) Using the result of problem 3 in homework 1. One can uniquely determine $|E| - |V| - 1$ edges which cannot be in the MST to remove. Thus the remained $|V| - 1$ edges form a unique MST.

Prob 4. (a) The optimal is achieved when the last two constraints are tight. Thus $x_1^* = -1$ and $x_2^* = -3$, and then $OPT = 3x_1^* + 4x_2^* = -15$.

(b) Newly introduce 3 slack variables r_1, r_2 and r_3. And break each x into its positive part and negative part.

$$
\begin{aligned}
\min \quad & 3x_1^+ - 3x_1^- + 4x_2^+ - 4x_2^- \\
\text{s.t.} \quad & x_1^+ - x_1^- + 2x_2^+ - 2x_2^- + r_1 = 14 \\
& 3x_1^+ - 3x_1^- - x_2^+ + x_2^- - r_2 = 0 \\
& x_1^+ - x_1^- - x_2^+ + x_2^- + r_3 = 2 \\
& x_1^+, x_1^-, x_2^+, x_2^-, r_1, r_2, r_3 \geq 0
\end{aligned}
$$

(c) The dual is

$$
\begin{aligned}
\max \quad & -14y_1 - 2y_3 \\
\text{s.t.} \quad & y_1 - 3y_2 + y_3 = -3 \\
& 2y_1 + y_2 - y_3 = -4 \\
& y_1, y_2, y_3 \geq 0
\end{aligned}
$$

Prob 5. Using dynamic programming. For any node u with children $v_1, v_2, \ldots, v_l$, let $f_1(u)$ be the size of minimum vertex cover of the subtree rooted at u where u is chosen, and $f_0(u)$ be the one where u is not chosen.

If u is not chosen, then every children of it should be chosen in order to cover those edges. Otherwise there is no constraint for its children. Then the recursion formula is

$$
\begin{aligned}
f_1(u) &= 1 + \sum_{i=1}^{t} \min(f_0(v_i), f_1(v_i)) \\
f_0(u) &= \sum_{i=1}^{t} f_1(v_i)
\end{aligned}
$$

One can solve this bottom-up. Suppose there are n nodes, in total there are $O(n)$ values to be calculated. When calculate all these values, each node is visited only once except the root. So the time complexity is $O(n)$.

The correctness can be proved by induction on the height of the tree. Clearly its correct for height-1 trees. Assume its correct for trees where the height is no larger then k. For height-$(k+1)$ tree with root u, if u is not chosen, then every of its children should be chosen, otherwise there is no constraint for the children. Then by the induction hypothesis, the recursion formula would be the optimal size of minimum vertex cover of this tree.

Prob 6. Consider the new graph with vertex set $V' = \{(u, v) : u, v \in V\}$ and edge set $E' = \{((u, v), (u', v')) : (u, u'), (v, v') \in E\}$. The two people random walk on the original graph is equivalent to the random walk on the new graph. Since starting at every vertex after one step it will reach each of its neighbors with equal probability.

So the problem becomes, for the new graph $G' = (V', E')$, starting from every vertex (u, v), on expected after $O(|E|^2 \cdot |V|)$ steps it will hit some vertex (t, t) where $t \in V$. Note that by the given Fact 2, it suffices to prove that for every vertex (u, v), with in a distance of $O(|V|)$ there exists some (t, t) where $t \in V$.

Here we are going to prove a stronger version:

Proposition 1. *In the new graph $G' = (V', E')$, the distance between any two vertices is $O(|V|)$.*

Proof. Let $d(\cdot, \cdot)$ be the shortest distance between two vertices in the original graph G. Clearly for any two vertex $s, t \in V$, $d(s, t) = O(|V|)$.

For any two vertex $(u_s, v_s), (u_t, v_t) \in V'$, because G' is symmetric, WLOG let's assume $d(u_s, u_t) \leq d(v_s, v_t)$. We are going to find a $O(|V|)$-length path between them.

The path is constructed as follow: Both people follow the shortest path in G to go to the destination. If the first one reaches u_t first, it chooses one of its neighbor t and alternatively moves between them.

Case 1: $d(u_s, u_t) + d(v_s, v_t)$ is even. Then by the time second person reaches v_t, first one stays at u_t thus complete the proof.

Case 2: $d(u_s, u_t) + d(v_s, v_t)$ is odd. The second person will change the strategy: find one odd cycle with shortest length in graph G (the existence of odd cycle is guaranteed by Fact 1) and one vertex t of it. If $d(u_s, u_t) + d(v_s, t) + d(t, v_t)$ is even then it's done, the second person just moves to t first and then to v_t. Otherwise after the second one moves to t, it goes through the odd cycle once back to t, and then moves to v_t. Note that the length of shortest odd cycle is also $O(|V|)$, this completes the proof. $\square$