Homework 8 Solutions

1. A graph is 2-colorable iff it is bipartite iff it does not have odd-length cycles.

   ```
   is-2-colorable(V, E)
   {
       return is_bipartite(V, E);
   }
   ```

   `is_bipartite()` is implemented using BFS/DFS and runs in time $\Theta(n+m)$.

2. We show 3-COLOR $\leqslant$ 4-COLOR. First we give a polynomial-time reduction function:

   ```
   // add a new vertex and an edge from this new vertex to each old vertex
   f(V, E)
   {
       V' = V + {new};
       E' = E;
       for each v in V
          E += (new, v);
       return (V', E');
   }
   ```

   The reduction runs in time $\Theta(n+m)$.

   Suppose (V, E) is 3-colorable. Then (V', E') is 4-colorable: simply use the fourth color for `new`. Conversely, suppose (V', E') is 4-colorable. Then `new`'s color different from any other vertex `w`'s color, since `(new, w)` is an edge. Hence 3 colors are enough to color (V, E).

3.

   ```
   is-4-colorable(V, E)
   {
        n = V.size();
        int C[0..n-1] = {0};
        while (true)
        {
                for (i = 0; i < n && C[i] == 3; ++i)
                         C[i] = 0;

                if (i == n)   // tried all assignments and found none
                    return false;

                C[i]++;
                valid = true;
                for each edge (a, b) in E
                    if (C[a] == C[b])
                    {
                        valid = false;
                        break;
                    }
                if (valid)
                    return true;
        }
   }
   ```

   The `while` loop iterates 4^n times, and the inner `for` loop iterates m times, so the algorithm runs in time $\Theta(m4^n)$.