1 Problem 1

We got two different approaches:

$$1)\ \mathcal{L}_{coupled}(A, \Theta) = \|A - PQ^T\|_F^2 + \|X - PG^T\|_F^2 + \|Y - QW^T\|_F^2$$

$$2)\ \mathcal{L}_{coupled}(A, \Theta) = \|A - PQ^T\|_F^2 + \|X - PG^T\|_F^2 + \|YW - Q\|_F^2$$

Let's compare them in terms of memory usage for cold item. For first option, we obtain embedding for item in this way:

$$q^* = argmin_q \|y - Wq\|_2^2 \rightarrow \nabla_q(\|y - Wq\|_2^2) = 0 \rightarrow 2W^T(-y + Wq) = 0$$

$$W^T y = W^T W q$$

We can solve it in a straightforward manner and count inverse of $W^T W$. However, W has size of $n_y * d$, and to find inverse matrix we need $O(d^3)$ time and additional $O(d^2)$ memory. Even if we want solve this problem with gradient methods, we need store $O(d)$ memory each step, and calculate additionally $O(d)$ memory as a gradient, which cost $O(n_y * d)$ in time for each step. So whole time consumption will be $O(k * n_y * d)$, where k - number of GD steps.

Let's look on second approach:

$$q^* = argmin_q \|yW - q\|_2^2 \rightarrow \nabla_q(\|yW - q\|_2^2) = 2(-yW + q) = 0$$

$$q = Wy$$

Here we need only count new q, which cost $O(d * n_y)$ in time and $O(n_y)$ in memory, because we don't need store q in RAM, for getting it just find Wy.

Thus, second representation of coupled factorization form will be more economic in terms of memory.