Q-1) $T(n) = T(n-1) + \underset{\text{for loop}}{2^{n-1}} + \underset{\text{union op}}{2^{n-1}}$

$T(n) = T(n-1) + 2^n \qquad T(0) = 1$

$T(1) = T(0) + 2$

$T(2) = T(1) + 2^2 \qquad T(2) = T(0) + 2^1 + 2^2$

$\vdots \qquad\qquad T(3) = T(0) + 2^1 + 2^2 + 2^3$

$T(n) = T(n-1) + 2^n \qquad T(n) = T(0) + 2^1 + 2^2 + \dots + 2^n$

$T(n) = 2^{n+1}$

$T(n) \in O(2^n)$

Q-5) $T(n) = 2T(n/2) + \Theta(1)$

$a = 2$

$b = 2 \qquad 2 > 2^0$

$d = 0 \qquad a > b^d$

$T(n) \in O(n^{\log_b a})$

$T(n) \in O(n)$

Q-4) $T(n,m) = (T(n, m-a)) * n + \Theta(1) * n$

$a \le m$

Complexity depends on the magnitude of elements in the array. In the worst case every amount can be found by using only 1. So $T_w = \Theta(\text{amount})$

Best case is one of coins will be equal to amount so $T_{best} = \Theta(1)$

$\Theta(1) < \text{Average} < \Theta(n)$

Q-2)

This algorithm is creating all combinations possible. For each case it will traverse all three distrete sets. So in this exhaustive traversing complexity will be $(n!)^2$. Because if there is n user, processes and processers, user1 can choose one of n process and one of n processors, user2 can choose one of $(n-1)$ " and one of $(n-1)$ processors. So it will be $n^2 \cdot (n-1)^2 \cdot (n-2)^2 \cdots 1^2$ which is $T(n) = \Theta(n!)^2$

Q-3)

It creates all possible permutations. So we can order them $n!$ different possibilities.