We want to design an algorithm for a cryptocurrency startup. We're given two arrays of positive number $N = \{N_1, \ldots, N_T\}$ and $W = \{W_1, \ldots, W_T\}$ which represent the profits from NorthCoins and WestCoins respectively. Each type of coins requires its own software which takes 1 unit of time to load.

1. First we define several subproblems. Let ProfitN(t) be the subproblem where the supercomputer mines optimally for time intervals 1 to $t-1$, and mines NorthCoins at time t. Likewise define ProfitW(t) and ProfitLoad(t) so that the computer mines WestCoins or loads software at time t, respectively. We order subproblems by increasing t. Let MaxProfit(t) be the profit if the computer mines optimally for time intervals 1 to $t-1$.
2. For $t = 1$, then MaxProfit$(1) = \max(N_1, W_1)$.

   For $t = 2$, MaxProfit$(2) = \max(N_1 + N_2, W_1 + W_2)$.
3. The recurrence relation Let ProfitSwitch(t)

$$\text{MaxProfit}(t) = \max(\text{MaxProfit}(t-2) + N_t, \text{MaxProfit}(t-2) + W_t, \text{MaxProfit}(t-1)$$

4. We prove that the recurrence relation is correct
5. An algorithm for finding a solution to the subproblem
6. An algorithm for finding the optimal solution to the original problem
7. The running time.