Adam Frenkel

First one piece of notation: When referring to a budget that is smaller than the given budget, but must be solved in order to do tabulation, I'll use the notation $budget_k$.

<u>Key Insights</u>: The first insight was that that you must take into account using every possible type of each class. The second insight was that you can take it one step at a time adding in another class of seforim each time, and store the best possible total spent for every $budget_k$ (bottom up tabulation). The main insight was that when adding another class, figuring out the best type to add, for a $budget_k$ is trivial given the previous classes. This is because one can simply find the max of all the types of the given class when adding: (the previous class's value for the ($budget_k – T_i$)) + T_i. By taking the max of this you will find the best type to add for this $budget_k$. Here is a diagram to explain these insights:
Budget: 8
Class: Mishna. Types: Zeraim – 1, Moed – 4, Nashim – 7
Class: Chumash. Types: Bereshis – 2, Shemos – 4, Vayikra – 6

	Mishna		Mishna & Chumash
$Budget_1$	1		X
$Budget_2$	1		X
$Budget_3$	1		1+2=3 Max=3
$Budget_4$	4		1+2=3 Max=3
$Budget_5$	4		1+2=3, 1+4=5 Max=5
$Budget_6$	4		4+2=6, 1+4=5 Max=6
$Budget_7$	7		4+2=6, 1+4=5, 1+6=7 Max=7
$Budget_8$	7		4+2=6, 4+4=8, 1+6=7 Max=8

<u>Optimal Substructure</u>: At each step, you can calculate the optimal solution for each C_i, adding in one class at a time.
<u>Overlapping Subproblems</u>: At every step you can rely on the previous step's calculation of the optimal solution for the $budget_k$ – each T_i.

<u>Recursion</u>:
Notation: Let A[k,x] be the optimal value for $budget_k$ with x classes

$$A[k,x] = \begin{cases} \max\{T_i & \text{When } x = 1\text{: For every } T_i \text{ in } C_i\text{, when } T_i \le k \\ \text{Minimum Integer (No Solution)} & \text{When every } T_i \text{ (in } C_i) > k \\ \max \begin{cases} T_i + A[k - T_i, x - 1] \end{cases} & \text{When } x > 1\text{: For every } T_i \text{ in } C_i\text{, when } A[k - T_i, x - 1] \text{ has a solution (i.e. it's not the Minimum Integer)} \\ \text{Minimum Integer} & \text{When for every } T_i \text{ in } C_i\text{, } A[k - T_i, x - 1] \text{ has no solution} \end{cases}$$

<u>Runtime</u>:
For maxAmountThatCanBeSpent(), the runtime is:
$O(C_j \cdot T_j \cdot \text{budget})$
(Terminology: C_j is the number of classes, T_j is the largest number of Types in any of the given classes, and budget is the given budget for the problem.)

For solution(), the runtime is:
$O(C_i)$