Solving Knapsack problem with Dynamic Programming

Knapsack capacity = 5

Item No.	Weight	Value
1	1	4
2	2	1
3	3	3

Phase 1: Building solution grid: left to right, top to bottom

Working set		Sizes of auxiliary knapsacks (k)					
Qty of items (i)	Contents	0	1	2	3	4	5
0	{}	**0**	**0**	**0**	**0**	**0**	**0**
1	{1}	**0**	**4**	**4**	**4**	**4**	**4**
2	{1, 2}	**0**	**4**	**4**	**5**	**5**	**5**
3	{1, 2, 3}	**0**	**4**	**4**	**5**	**7**	**7**

s(1, 1) = 4	Into knapsack of size 1 we can fit item 1 and thus collect value 4. This is better than value zero, so 4 becomes the solution value and stays until the end of the row as we have no more items in the working set.
s(2, 1) = 4	We cannot fit item 2 (weight 2) into a sack of size 1. Therefore, the solution for this sack is the same as it was found for the working set without item 2. This value is recorded in the same column of the previous row.
s(2, 2) = 4	Now, having the sack of size 2, we can fit item 2 into the sack. It will take up all the room, so the solution value would be just the value of item 2 which is 1. However, if we ignore item 2, we go bavk to the solution value from the previous row, which is 4. The latter is better, so we do not take item 2 into the sack of size 2.
s(2, 3) = 5	We can fit item 2 into the sack of size 3. It will consume two units of weight and bring value=1. This leaves us with one vacant unit of weight. We already know the best way to fill it, as it is equivalent of the solution for sack of size 1 with the previous contents of the working set: s(1, 1)=4. Adding up we get the value of 5 which is better than ignoring item 2 (that would be 4).
s(3, 3) = 5	With item 3 in the working set, having reached sack of size 3, we can fit this item in the sack. It will bring the value =3 and no room will be left for more items. However, if we ignore item 3, we revert to the previous contents of the working set and consult the previous row for the solution value, which is s(2, 3)=5. The latter is better.
s(3, 4) = 7	Now, having sack of size 4, we can fit item 3 (value=3) and have a vacant room for one unit of weight. The best solution for the sack of size 1 with the previous contents of the working set is s(i=2, k=1) which is 4. Adding up, we get value of 7. If we skip item 3 we have to revert to s(2, 4)=5 which is less than 7, so we choose to take item 3 and get total value of 7. Note that, at this moment, we do not remember how we obtained that value of 4 for s(2, 1) but we are sure we had some valid solution there.
s(3, 5) = 7	Exactly the same reasoning applies here as at the previous step. This is the final iteration and we arrive at the solution value=7 for the knapsack of size 5 with working set consisting of items 1, 2, and 3.

Phase 2: Backtracking solution grid: bottom to top, right to left

Working set		Sizes of auxiliary knapsacks (k)					
Qty of items (i)	Contents	0	1	2	3	4	5
0	{}	**0**	**0**	**0**	**0**	**0**	**0**
1	{1}	**0**	**4**	**4**	**4**	**4**	**4**
2	{1, 2}	**0**	**4**	**4**	**5**	**5**	**5**
3	{1, 2, 3}	**0**	**4**	**4**	**5**	**7**	**7**

Starting with the final solution value 7, we check if we have the same solution value for the previous contents of the working set. This is the number in the same column in the previous row, and it is 5. This means, that the item 3 was taken into the knapsack, otherwise the solution values for working sets {1, 2, 3} and {1, 2} would not differ. Then we remove item 3 together with its part of sack's capacity and arrive at knapsack of size 5-3=2 and working set {1,2}.

Here again we compare the solution value with that of the previous row. They appear identical. This means that item 2 was not taken into the knapsack, otherwise it would change the accumulated value. So, we remove item 2 from consideration but do not shrink the sack as item 2 was not there. This way, we arrive at sack of the same size 2 with the working set {1}.

The value 4 differs from zero in the previous row meaning that item 1 was taken into the knapsack. Having removed item 1 we arrive at the sack of size 2-1=1 and empty working set {}. When we reach the empty working set, backtracking stops. **We found that items 1 and 3 were taken into he knapsack, item 2 was left out, and there remained one unit of unused capacity in the sack.**