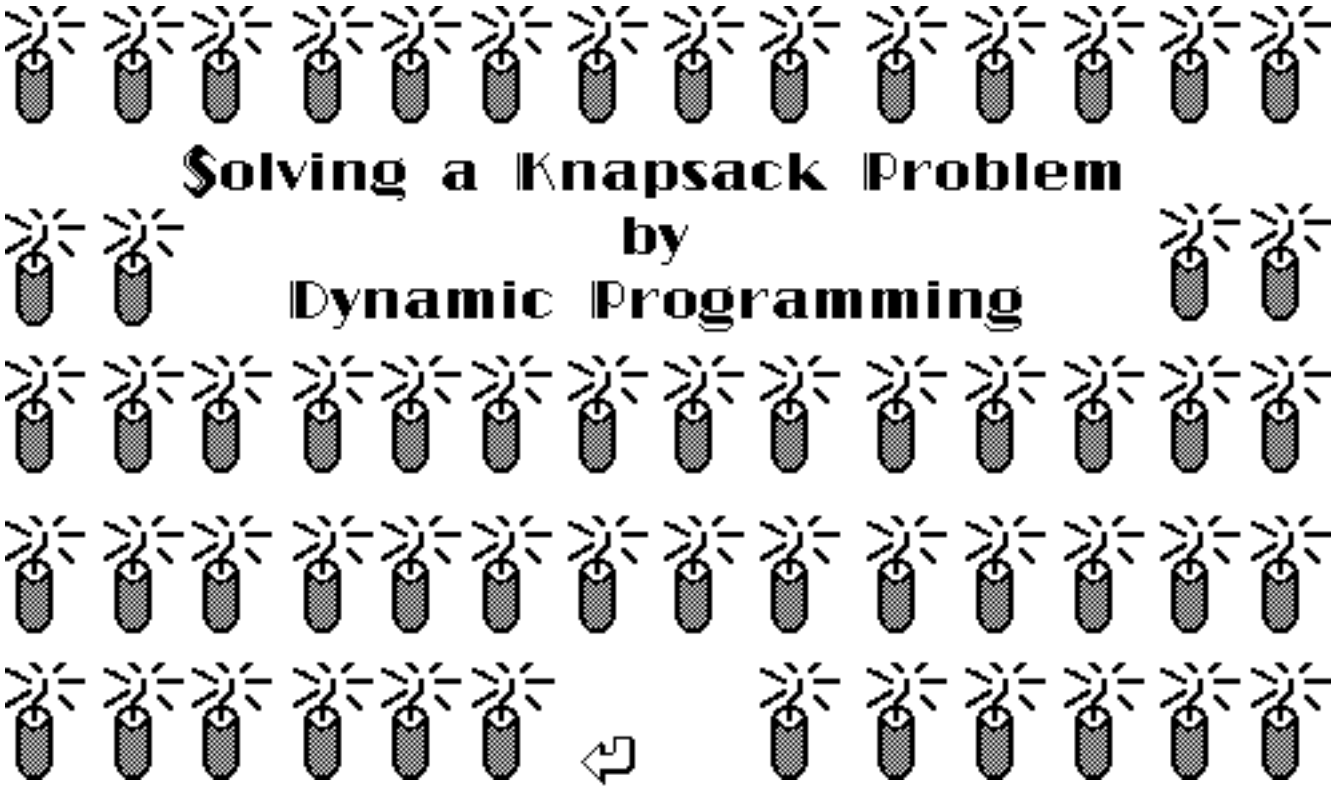
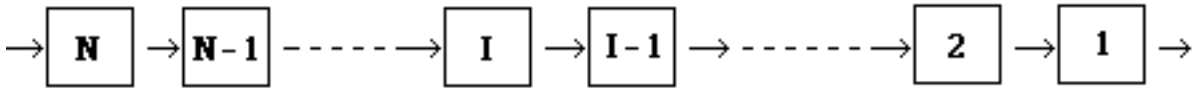


Solving a Knapsack Problem by Dynamic Programming

In DYNAMIC PROGRAMMING, we treat the knapsack problem as a sequential decision problem, in which at each stage in the sequence, we have a single decision variable, namely number of units of a single item to place into the knapsack.



Let's suppose that the knapsack is filled by considering the items in reverse order, i.e., first we decide how many units of item #N we will put into the knapsack. Then, taking into consideration the unused capacity that remains, we decide how many units of #N-1 to put into the knapsack, and so on, until we finally consider item #1.

At each stage, we need to know the amount of unused capacity which remains in order to be able to decide how many units of the item under consideration will be put into the knapsack.

This quantity, the amount of unused capacity, will be called the "state variable" of the system, while the number of units of the item will be called, of course, the "decision variable" of the system at that stage.

We will assume that the weights are integer-valued, and that the possible values of the state variable are: 0, 1, 2, ... CAP where CAP is the capacity of the knapsack.

At each stage of the dynamic programming computation, the optimal decision is selected for each of the possible values of the state variable, by **completely enumerating** all the combinations of state and decision variables.

Thus, if there are $m = CAP+1$ values for the state variable, and $n = X + 1$ values for the decision variable (where the decision variable may take values 0, 1, ... X), then we must enumerate $m \times n$ state-decision combinations.

Let's look at a simple example:

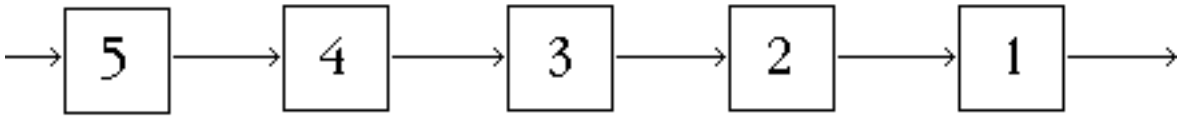
Example

Number of items: 5
Capacity of Knapsack: 12
Maximum units of any item to be included is 1

i	W	V
1	8	80
2	6	48
3	2	14
4	3	18
5	4	22

W = 'weight' of item
V = value of item

We will assume that the items are to be placed into the knapsack in reverse order:



When we finally arrive at item #1, the decision will be very easy: we simply add the item if there is enough capacity remaining, otherwise we have no choice but to omit it.

The number -99999999 actually represents negative infinity, representing an infeasible combination

state (unused capacity)

s \ x:	0	1
0	0.00	-99999999.00
1	0.00	-99999999.00
2	0.00	-99999999.00
3	0.00	-99999999.00
4	0.00	-99999999.00
5	0.00	-99999999.00
6	0.00	-99999999.00
7	0.00	-99999999.00
8	0.00	80.00
9	0.00	80.00
10	0.00	80.00
11	0.00	80.00
12	0.00	80.00

STAGE 1

The entry in row i, column j of the table is the value of the knapsack for that combination of state & decision.

At stage #1, we are deciding whether to choose item #1.

Since this is the last item to be considered, it is a simple matter to decide whether to put the item into the knapsack.

Since item #1 is worth $80, if there is sufficient capacity remaining (8 pounds or more), we choose the item. Otherwise, we cannot choose the item.

Item #1 requires 8 units of capacity, & has value $80

Once we have computed the value for every combination of state & decision, we find the maximum value which can be achieved for each state:

STAGE 1

s \ x:	0	1	STATE	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	-99999999.00	0	0.00	0	0
1	0.00	-99999999.00	1	0.00	0	1
2	0.00	-99999999.00	2	0.00	0	2
3	0.00	-99999999.00	3	0.00	0	3
4	0.00	-99999999.00	4	0.00	0	4
5	0.00	-99999999.00	5	0.00	0	5
6	0.00	-99999999.00	6	0.00	0	6
7	0.00	-99999999.00	7	0.00	0	7
8	0.00	80.00	8	80.00	1	0
9	0.00	80.00	9	80.00	1	1
10	0.00	80.00	10	80.00	1	2
11	0.00	80.00	11	80.00	1	3
12	0.00	80.00	12	80.00	1	4

This column, then is the best that we can do, using item #1, given the capacity that remains in the knapsack!

This is the capacity that remains after we make the best decision!

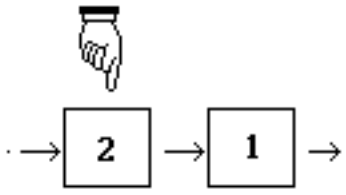
Only this part of the table needs to be kept for use in further computations.

—STAGE 1—

STATE	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	0.00	0	2
3	0.00	0	3
4	0.00	0	4
5	0.00	0	5
6	0.00	0	6
7	0.00	0	7
8	80.00	1	0
9	80.00	1	1
10	80.00	1	2
11	80.00	1	3
12	80.00	1	4

Remember: the optimal value here tells us the most value we can get of item #1, for each of the possible values of unused capacity when we arrive at stage #1.

Now comes the "tricky" part of DP:



We are at this stage, but we don't know what the previous decisions (at stages 5, 4, and 3) were, and our decision at stage 2 will determine the stage when we get to stage 1.

Suppose that we are now going to decide whether to add item 2 to the knapsack.

We don't know yet which of items 5, 4, and 3 were added, so we don't know what the remaining capacity is, and will have to find the best decision for every possible value of the state variable.

---STAGE 2---

units of item #2

unused capacity

s \ x:	0	1
0	0.00	-99999999.00
1	0.00	-99999999.00
2	0.00	-99999999.00
3	0.00	-99999999.00
4	0.00	-99999999.00
5	0.00	-99999999.00
6	0.00	48.00
7	0.00	48.00
8	80.00	48.00
9	80.00	48.00
10	80.00	48.00
11	80.00	48.00
12	80.00	48.00

Item #2 requires 6 units of capacity & has value $48

Consider, for example, that 9 units of capacity remain empty in the knapsack when we arrive at consideration of item 2.

If we add item 2 to the knapsack, then the value of the knapsack will be $ 48 (the value of item 2) plus the value that we can get from item 1, using the 2 units of capacity which would remain (8 now, minus 6 used by item 2), namely $0 (from the result of the previous calculation)

---STAGE 2---

units of item #2

unused capacity

s \ x:	0	1
0	0.00	-99999999.00
1	0.00	-99999999.00
2	0.00	-99999999.00
3	0.00	-99999999.00
4	0.00	-99999999.00
5	0.00	-99999999.00
6	0.00	48.00
7	0.00	48.00
8	80.00	48.00
9	80.00	48.00
10	80.00	48.00
11	80.00	48.00
12	80.00	48.00

from stage 1:

STATE	OPTIMAL VALUES
0	0.00
⋮	⋮
7	0.00
8	80.00
9	80.00

If, on the other hand, we do NOT add item 2 to the knapsack, then we neither add value nor use capacity at this stage, so that when we arrive at stage 1 (consideration of item 1), we will again have 8 units of unused capacity at our disposal. Consulting our table of calculations for stage 1, we find that we can achieve a value of $80 if we have 8 units of unused capacity. Therefore, we write 80 into our table here.

For each possible state, we find the maximum value that we can attain, and record that value and the decision yielding that value in the table:

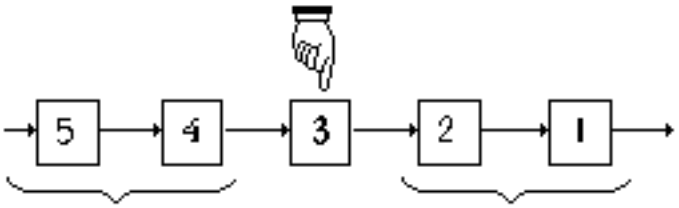
	—STAGE 2—				
			OPTIMAL	OPTIMAL	RESULTING
s \ x:	0	1	VALUES	DECISIONS	STATE
0	0.00	-99999999.00	0.00	0	0
1	0.00	-99999999.00	0.00	0	1
2	0.00	-99999999.00	0.00	0	2
3	0.00	-99999999.00	0.00	0	3
4	0.00	-99999999.00	0.00	0	4
5	0.00	-99999999.00	0.00	0	5
6	0.00	48.00	48.00	1	0
7	0.00	48.00	48.00	1	1
8	80.00	48.00	80.00	0	8
9	80.00	48.00	80.00	0	9
10	80.00	48.00	80.00	0	10
11	80.00	48.00	80.00	0	11
12	80.00	48.00	80.00	0	12

Again, only the optimal value and decisions need to be kept for use in further computations:

	—STAGE 2—		
	OPTIMAL	OPTIMAL	RESULTING
s	VALUES	DECISIONS	STATE
0	0.00	0	0
1	0.00	0	1
2	0.00	0	2
3	0.00	0	3
4	0.00	0	4
5	0.00	0	5
6	48.00	1	0
7	48.00	1	1
8	80.00	0	8
9	80.00	0	9
10	80.00	0	10
11	80.00	0	11
12	80.00	0	12

Remember: the optimal value here tells us the most value we can get of items #1 and 2, combined, for each possible value of the unused capacity when we get to stage #2!

Now we back up to stage 3. At this stage, we will have decided upon choice of items 4 & 5, and are to decide whether to include item 3 in the knapsack.



decisions will have been already made

decisions yet to be made, will depend upon choice of item 3.

—STAGE 3—

s	x: 0	1
0	0.00	-99999999.00
1	0.00	-99999999.00
2	0.00	14.00
3	0.00	14.00
4	0.00	14.00
5	0.00	14.00
6	48.00	14.00
7	48.00	14.00
8	80.00	62.00
9	80.00	62.00
10	80.00	94.00
11	80.00	94.00
12	80.00	94.00

—STAGE 3—

s	x: 0	1
0	0.00	-99999999.00
1	0.00	-99999999.00
2	0.00	14.00
3	0.00	14.00
4	0.00	14.00
5	0.00	14.00
6	48.00	14.00
7	48.00	14.00
8	80.00	62.00
9	80.00	62.00
10	80.00	94.00
11	80.00	94.00
12	80.00	94.00

As at stage 2, although we will have already made the decisions at stages 5 and 4, when we are doing these computations, we don't know what those decisions are!

Therefore we don't know what the state of the system (i.e., the amount of unused capacity) will be at stage 3, and must compute the best decision for each of the possible states (0, 1, 2, ... 12)

from stage 2:

s	OPTIMAL VALUES
0	0.00
1	0.00
2	0.00
3	0.00
4	0.00
5	0.00
6	48.00
7	48.00
8	80.00
9	80.00
10	80.00
11	80.00
12	80.00

Item #3 requires 2 units of capacity and has a value of $14

Take, for example, the case where there are 10 units of capacity remaining empty when we arrive at stage 3. Suppose that we choose item #3, which would use 2 units of the capacity, leaving 8 units. Consulting the table of optimal values from stage 2, we see that with 8 units of unused capacity we can get a value of $80 from the stages that remain. Therefore, the best that we can do, if we make this decision, is

—STAGE 3—

s \ x:	0	1
0	0.00	-99999999.00
1	0.00	-99999999.00
2	0.00	14.00
3	0.00	14.00
4	0.00	14.00
5	0.00	14.00
6	48.00	14.00
7	48.00	14.00
8	80.00	62.00
9	80.00	62.00
10	80.00	94.00
11	80.00	94.00
12	80.00	94.00

$14 + $80 = $94

from item 3 *from the remaining items*

Again, we record, for each possible value of the state variable, the maximum value that we can obtain. This represents the total value obtainable from stages 3, 2, and 1:

s \ x:	0	1	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	-99999999.00	0.00	0	0
1	0.00	-99999999.00	0.00	0	1
2	0.00	14.00	14.00	1	0
3	0.00	14.00	14.00	1	1
4	0.00	14.00	14.00	1	2
5	0.00	14.00	14.00	1	3
6	48.00	14.00	48.00	0	6
7	48.00	14.00	48.00	0	7
8	80.00	62.00	80.00	0	8
9	80.00	62.00	80.00	0	9
10	80.00	94.00	94.00	1	8
11	80.00	94.00	94.00	1	9
12	80.00	94.00	94.00	1	10

from stage 3:

s	OPTIMAL VALUES
0	0.00
1	0.00
2	14.00
3	14.00
4	14.00
5	14.00
6	48.00
7	48.00
8	80.00
9	80.00
10	94.00
11	94.00
12	94.00

Item #4 requires 3 units of capacity, and has a value of $18

Backing up to stage 4, we repeat the procedure. For example, if we have 12 units of unused capacity, and if we choose item 4, we will obtain a value of $18 + $80 = $98

---STAGE 4---

s	x: 0	1
0	0.00	-99999999.00
1	0.00	-99999999.00
2	14.00	-99999999.00
3	14.00	18.00
4	14.00	18.00
5	14.00	32.00
6	48.00	32.00
7	48.00	32.00
8	80.00	32.00
9	80.00	66.00
10	94.00	66.00
11	94.00	98.00
12	94.00	98.00

If we do not choose item 4, we obtain a value of $0 + $94

We again record the optimal values for each possible stage, by finding the maximum value obtainable in each row of the table on the left:

—STAGE 4—

s	x: 0	1	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	-99999999.00	0.00	0	0
1	0.00	-99999999.00	0.00	0	1
2	14.00	-99999999.00	14.00	0	2
3	14.00	18.00	18.00	1	0
4	14.00	18.00	18.00	1	1
5	14.00	32.00	32.00	1	2
6	48.00	32.00	48.00	0	6
7	48.00	32.00	48.00	0	7
8	80.00	32.00	80.00	0	8
9	80.00	66.00	80.00	0	9
10	94.00	66.00	94.00	0	10
11	94.00	98.00	98.00	1	8
12	94.00	98.00	98.00	1	9

These optimal values represent the best total value of items 4, 3, 2, and 1 which can be obtained for a given amount of unused capacity!

from stage 4:

s	OPTIMAL VALUES
0	0.00
1	0.00
2	14.00
3	18.00
4	18.00
5	32.00
6	48.00
7	48.00
8	80.00
9	80.00
10	94.00
11	98.00
12	98.00

Item #5 requires 4 units of capacity, and has a value of $22

Finally, we have arrived at the computations for stage #5. Since this is the first decision to be made, we know what the state will be, namely 12 units of unused capacity.

For the sake of consistency, however, we compute the entire table (not just the row for s=12)

—STAGE 5—

s \ x:	0	1
0	0.00	-99999999.00
1	0.00	-99999999.00
2	14.00	-99999999.00
3	18.00	-99999999.00
4	18.00	22.00
5	32.00	22.00
6	48.00	36.00
7	48.00	40.00
8	80.00	40.00
9	80.00	54.00
10	94.00	70.00
11	98.00	70.00
12	98.00	102.00

The maximum value in each row of the table on the left is recorded, along with the decision which yields that optimal value:

—STAGE 5—

s \ x:	0	1	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	-99999999.00	0.00	0	0
1	0.00	-99999999.00	0.00	0	1
2	14.00	-99999999.00	14.00	0	2
3	18.00	-99999999.00	18.00	0	3
4	18.00	22.00	22.00	1	0
5	32.00	22.00	32.00	0	5
6	48.00	36.00	48.00	0	6
7	48.00	40.00	48.00	0	7
8	80.00	40.00	80.00	0	8
9	80.00	54.00	80.00	0	9
10	94.00	70.00	94.00	0	10
11	98.00	70.00	98.00	0	11
12	98.00	102.00	102.00	1	8

—STAGE 5—

s	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	14.00	0	2
3	18.00	0	3
4	22.00	1	0
5	32.00	0	5
6	48.00	0	6
7	48.00	0	7
8	80.00	0	8
9	80.00	0	9
10	94.00	0	10
11	98.00	0	11
12	102.00	1	8

This table now tells us that if we begin to fill the knapsack, having 12 units of capacity available to us, we can obtain a maximum value of $102.

It also tells us that we should choose item #5 in this situation, which will leave only 8 units of unused capacity when we arrive at stage #4.

—STAGE 4—

s	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	14.00	0	2
3	18.00	1	0
4	18.00	1	1
5	32.00	1	2
6	48.00	0	6
7	48.00	0	7
8	80.00	0	8
9	80.00	0	9
10	94.00	0	10
11	98.00	1	8
12	98.00	1	9

Now that we know what the state of the system will be in stage 4, we can consult the table which we computed earlier to find the optimal decision, namely:
if we enter stage 4 with 8 units of unused capacity, DO NOT choose item #4.
Therefore, 8 units of unused capacity remain when we reach stage 3.

—STAGE 3—

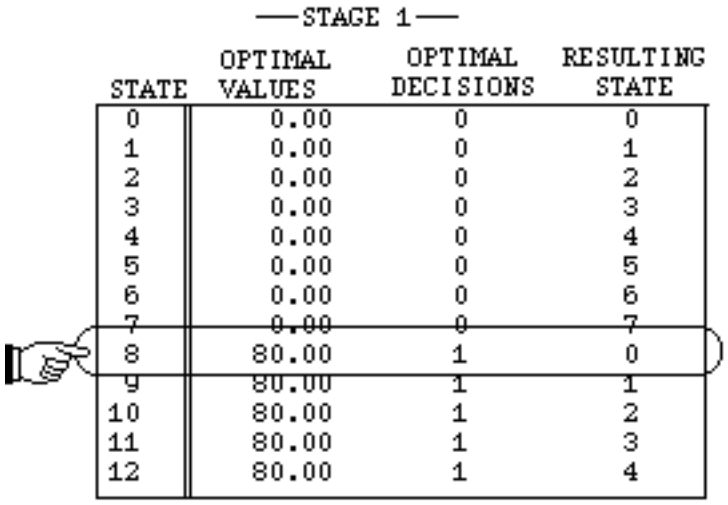
s	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	14.00	1	0
3	14.00	1	1
4	14.00	1	2
5	14.00	1	3
6	48.00	0	6
7	48.00	0	7
8	80.00	0	8
9	80.00	0	9
10	94.00	1	8
11	94.00	1	9
12	94.00	1	10

Consulting the table of optimal values and decisions for stage 3, we see that if we enter stage 3 with 8 units of unused capacity, the optimal decision is to NOT choose item #3, leaving again 8 units of unused capacity when we reach stage 2.

—STAGE 2—

s	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	0.00	0	2
3	0.00	0	3
4	0.00	0	4
5	0.00	0	5
6	48.00	1	0
7	48.00	1	1
8	80.00	0	8
9	80.00	0	9
10	80.00	0	10
11	80.00	0	11
12	80.00	0	12

Consulting the table we computed for stage 2, we see also that we should NOT choose item #2 if we have 8 units of unused capacity. Therefore, when we arrive at stage 1, we will still have the 8 units of unused capacity.



—STAGE 1—

STATE	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	0.00	0	2
3	0.00	0	3
4	0.00	0	4
5	0.00	0	5
6	0.00	0	6
7	0.00	0	7
8	80.00	1	0
9	80.00	1	1
10	80.00	1	2
11	80.00	1	3
12	80.00	1	4

Not surprisingly, the table which we computed for stage 1 tells us that we SHOULD choose item #1 for the knapsack, completely filling the knapsack (0 units of unused capacity will remain!)

The APL workspace displays the optimal solution, and allows you to view the tables saved at each stage:

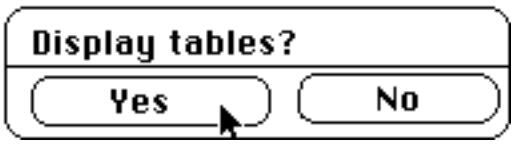
Example

*** Optimal value is 102 ***

i	V	W	X
1	80	8	1
5	22	4	1

Total Weight: 12

You may now display the tables of the optimal value function and optimal decision for each state at every stage



Notice that we have really solved, with little additional effort, several other knapsack problems (one for each of the capacities 0 through 11).

What is the optimal value that we can obtain if we have only 11 units of capacity, rather than 12?

Which items are to be placed into the knapsack to obtain this value?

OPTIMAL
RETURNS AND DECISIONS

STAGE 5:

STATE	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	14.00	0	2
3	18.00	0	3
4	22.00	1	0
5	32.00	0	5
6	48.00	0	6
7	48.00	0	7
8	80.00	0	8
9	80.00	0	9
10	94.00	0	10
11	98.00	0	11
12	102.00	1	8

Go through the tables again, this time entering state #5 with 11 units of unused capacity, and construct the optimal solution.

STAGE 4:

STATE	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	14.00	0	2
3	18.00	1	0
4	18.00	1	1
5	32.00	1	2
6	48.00	0	6
7	48.00	0	7
8	80.00	0	8
9	80.00	0	9
10	94.00	0	10
11	98.00	1	8
12	98.00	1	9

STAGE 3:

STATE	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	14.00	1	0
3	14.00	1	1
4	14.00	1	2
5	14.00	1	3
6	48.00	0	6
7	48.00	0	7
8	80.00	0	8
9	80.00	0	9
10	94.00	1	8
11	94.00	1	9
12	94.00	1	10

STAGE 2:

STATE	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	0.00	0	2
3	0.00	0	3
4	0.00	0	4
5	0.00	0	5
6	48.00	1	0
7	48.00	1	1
8	80.00	0	8
9	80.00	0	9
10	80.00	0	10
11	80.00	0	11
12	80.00	0	12

STAGE 1:

STATE	OPTIMAL VALUES	OPTIMAL DECISIONS	RESULTING STATE
0	0.00	0	0
1	0.00	0	1
2	0.00	0	2
3	0.00	0	3
4	0.00	0	4
5	0.00	0	5
6	0.00	0	6
7	0.00	0	7
8	80.00	1	0
9	80.00	1	1
10	80.00	1	2
11	80.00	1	3
12	80.00	1	4

While Dynamic Programming may seem like a very tedious, complicated procedure, it can be easily coded in the APL language.

```
     ∇VALUE←F N;t
[1]   ⍝
[2]   ⍝           Optimal Value Function of DP model
[3]   ⍝               of the Knapsack problem
[4]   ⍝
[5]   →LAST IF N=0
[6]   VALUE←MAX (((⍴s)⍴0) ∘.+ V[N]×x ) + (F N-1)[TRANSITION s ∘.- W[N]×x ]
[7]   →0
[8]   LAST:VALUE←((⍴s)⍴0), -BIG
     ∇
```