0-1 Knapsack - 3rd most useful problem (out of 75).

Cave, n bars of gold 1..n, each with weight $w_i$, value $v_i$. You have knapsack with weight limit $W$.

Goal: Choose bars to fill knapsack while maximizing sum of $v_i$.

Applications:

Choosing:

1. Shares of stocks to invest in w/ max budget.
   * Stocks ~ items (some stocks more valuable).
   * Budget ~ weight limit.

2. Questions to answer on exam w/ time limit.
   * Question ~ items (some questions worth more points).
   * Time limit ~ weight limit.

3. Ads to place on website w/ max space.
   * Ads ~ items
   * Space ~ weight limit.

Formally:

$n$ items. For each item $i$, make a choice:

$x_i = 0$ if leaving item $i$.

$x_i = 1$ "taking" $i$.

Goal: Find $x_i$ for that maximizes $\sum_{i=1}^{N} x_i v_i$ such that $\sum_{i=1}^{N} w_i x_i \leq W$. 

\[
1 \leq i \leq N.
\]
Brute Force: $O(2^n)$: For each item, 2 choices: take or leave.

Greedy Ideas?
Sort by $T$ and choose next item from list:
1. decreasing value
2. increasing weight
3. decreasing value/weight

3-item examples where greedy fails?

(1) $W = 10$ lbs

<table>
<thead>
<tr>
<th>10 lbs</th>
<th>5 lbs</th>
<th>5 lbs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$100$</td>
<td>$90$</td>
<td>$80$</td>
</tr>
</tbody>
</table>

Greedy = $G = 100$

$OPT = 170$

(2) $W = 10$ lbs

<table>
<thead>
<tr>
<th>4</th>
<th>6</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1$</td>
<td>$1$</td>
<td>$100$</td>
</tr>
</tbody>
</table>

$OPT = 100$

(3) $W = 50$

<table>
<thead>
<tr>
<th>10</th>
<th>20</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>$60$</td>
<td>$100$</td>
<td>$120$</td>
</tr>
</tbody>
</table>

$G = 160$

$OPT = 220$

$\frac{V_i}{W_i} = 6, 5, 4$

None of the greedy ideas are optimal!

Optimal substructure property? Yes! $\Rightarrow$ Dynamic Programming
ex. \( W = 5, n = 4 \) items

<table>
<thead>
<tr>
<th>Item</th>
<th>Weight</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>40</td>
</tr>
</tbody>
</table>

Solu: items 2, 4 value = 90.

Start by finding max value (instead of items).

Express solution to a subproblem in terms of subsolutions.

What is a subproblem? Imagine bars of gold are lined up. Entire problem: make decisions for all \( n \) bars. Sub-problem: make decisions for first, say \( i \) bars.

\( A[i] \): max value for first \( i \) items (with weight limit \( W \)).


\( A[i-1] \): max value for first \( i-1 \) items (with weight limit \( W \)).

Problem? If we decide to take item \( i \), then the weight limit for previous \( i-1 \) items is not \( W \) (weight limit is \( W - W_i \)).

\( \text{ex.} \quad \square \quad \square \quad \square \quad \square \quad \quad W = 100, \text{if we take } i, \text{items } i = 5 \text{ lbs} \quad \text{1...i-1 can weigh at most 95} \)
Therefore, D.P. formulation depends on 2 factors:
(1) # items  (2) weight limit.

\[ A[i,j] = \text{max value from first i items with weight limit } j. \]

D.P. Formulation:

For each item \( i \), we have a choice: take \( i \), or leave \( i \).

- Leave \( i \): 
  \[ A[i,j] = A[i-1,j] \]
  
  If we leave item \( i \), we have weight limit \( j \) for previous \( i-1 \) items.

Where in the matrix can we find the best value we can get for the first \( i-1 \) items and weight limit \( j \)?
take \( i \)? \( A[i, j] = v_i + \max \{ A[i-1, j-w_i] \} \)

If weight limit is \( j \) and we take item \( i \), previous \( i-1 \) items can weigh at most \( j - w_i \).

Where in \( A \) can we find best value for first \( i-1 \) items and weight limit \( j - w_i \)?

ex Suppose deciding on 5th item \((i=5)\), weight limit is 15 \((j=15)\).

\[
\begin{array}{cccccc}
1 & 2 & 3 & 4 & 5 \\
\checkmark & \checkmark & \checkmark & \checkmark & \checkmark \\
\end{array}
\]

\((5 \text{ lbs})\)

already made a decision for each of these, got some total value $100$.

If we leave \( i \), value is still $100$.

If we take \( i \), total weight of all items must be 15.
So \( i-1 \) items can weigh at most 10.
One more case: What if weight of current item exceeds current weight limit?
(ex: 5th item weighs 25 lbs?)

⇒ must leave item i!

\[
A[i,j] = \begin{cases} 
0 & \text{if } i=0 \text{ or } j=0 \\
A[i-1,j] \text{ (leave)} & \text{if } i,j>0 \text{ and } w_i > j \quad \text{(A)} \\
\max \left\{ 
\begin{align*}
A[i-1,j] \text{ (leave)} \\
\text{(above 1)}
\end{align*}
\right. & \text{if } i,j>0, \\
v_i + A[i-1,j-w_i] \text{ (take)} & \text{if } w_i \leq j \quad \text{(B)}
\end{cases}
\]

1. Base Case?
2. How to fill A? \( A[0,0] = 0 \) \( A[0,n] = 0 \) 
Then, nested for loop: for \( i = 1 \ldots n \), for \( j = 1 \ldots W \)
3. Where is value (max, value)? in \( A[n,W] \)
4. Run Time? \( O(nW) \)
5. Actual solution (optimal items)? <Later>