Problem statement

A thief robbing a store finds n items; the ith item is worth v_i dollars and weights w_i pounds. He wants to take as valuable a load as possible, but he can carry at most W pounds in his knapsack. What items should be taken?

Formally, the problem can be stated as follws:

- Input: n items of values $v_1, v_2, \ldots, v_n$ and of the weight $w_1, w_2, \ldots, w_n$, and a total weight W, where v_i, w_i and W are positive integers.
- Output:a subsect $\mathcal{S} \subseteq \{1, 2, \ldots, n\}$ of the items such that

$$\sum_{i \in \mathcal{S}} w_i \leq W \quad \text{and} \quad \sum_{i \in \mathcal{S}} v_i \quad \text{is maximized.}$$

This is called the **0-1 knapsack problem** because each item must either be taken or left behind; the thief cannot take a fractional amount of an item or take an item more than once.

Example

$n = 9,$

$v = \langle 2, 3, 3, 4, 4, 5, 7, 8, 8 \rangle,$

$w = \langle 3, 5, 7, 4, 3, 9, 2, 11, 5 \rangle,$

$W = 15.$

Max benifit from problem at left 23

– why?

set of items to take
$\{9, 7, 5, 4\}$

```
ALGORITHM BruteForce (Weights [N], Values [N], N, Capacity )
//Finds the best possible combination of items for the KP
//Input: Array Weights contains the weights of all items
         Array Values contains the values of all items
         Array A initialized with 0s is used to generate the bit strings
//Output: Best possible combination of items in the knapsack bestChoice [N]

for i = 1 to 2^N do
         j ← N
         tempWeight ← 0
         tempValue ← 0
         while ( A[j] != 0 and j > 0)
                  A[j] ← 0
                  j ← j – 1
         A[j] ← 1
         for k ← 0 to N – 1 do
                  if (A[k] = 1) then
                           tempWeight ← tempWeight + Weights[k]
                           tempValue ← tempValue + Values[k]
         if  ((tempValue > bestValue) AND (tempWeight ≤ Capacity)) then
                  bestValue ← tempValue
                  bestWeight ← tempWeight
                  bestChoice ← A
return bestChoice
```

for $i = 0, 1, \ldots, n-1$

$$c[i, w] = \begin{cases} 0 & \text{if } \ i = 0 \text{ or } w \leq 0 \\ \max\left(v_i + c[i-1, w - w_i], c[i-1, w]\right) & \text{if } \ i > 0 \text{ and } w_i \leq w \\ c[i-1, w] & \text{if } \ i > 0 \text{ and } w_i > w \end{cases}$$

Example

$n = 9,$

$v = \langle 2, 3, 3, 4, 4, 5, 7, 8, 8 \rangle,$

$w = \langle 3, 5, 7, 4, 3, 9, 2, 11, 5 \rangle,$

$W = 15.$

$c[9, 15] = 23$

set of items to take
$\{9,\ 7,\ 5,\ 4\}$ – why?