0/1 Knapsack

Try to solve the 0/1 Knapsack problem.

We'll cover the following

- Statement
- Examples
- Understand the problem
- Figure it out!
- Try it yourself

Statement

You are given n items whose weights and values are known, as well as a knapsack to carry these items. The knapsack cannot carry more than a certain maximum weight, known as its **capacity**.

You need to maximize the total value of the items in your knapsack, while ensuring that the sum of the weights of the selected items does not exceed the capacity of the knapsack.

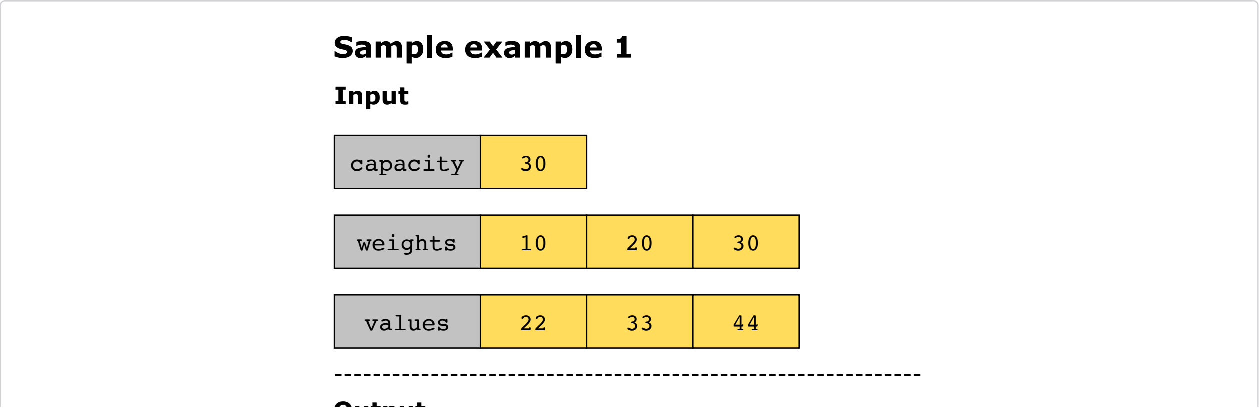
If there is no combination of weights whose sum is within the capacity constraint, return 0.

> **Notes:**
>
> 1. An item may not be broken up to fit into the knapsack, i.e., an item either goes into the knapsack in its entirety or not at all.
> 2. We may not add an item more than once to the knapsack.

Constraints:

- $1 \leq$ `capacity` $\leq 10^4$
- $1 \leq$ `values.length` $\leq 10^3$
- `weights.length` $==$ `values.length`
- $1 \leq$ `values[i]` $\leq 10^4$
- $1 \leq$ `weights[i]` $\leq$ `capacity`

Examples



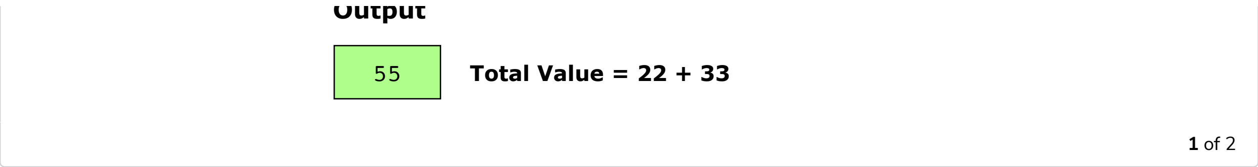
Sample example 1

Input

capacity	30

weights	10	20	30
values	22	33	44

Output



Understand the problem

Let’s take a moment to make sure you’ve correctly understood the problem. The quiz below helps you check if you’re solving the correct problem:

0/1 Knapsack

1 You have three items with weights 2, 3, and 4, respectively. You have a knapsack with a capacity of 7. How many different combinations of items can you include in the knapsack?

A) 3

B) 4

C) 5

D) 6

Submit Answer

Question 1 of 3
0 attempted

Reset Quiz

Figure it out!

We have a game for you to play. Rearrange the logical building blocks to develop a clearer understanding of how to solve this problem.

Drag and drop the cards to rearrange them in the correct sequence.

Create a 2D table to store the maximum profit for each item and capacity.

Initialize the table with 0s for the first row and column to handle base cases.

Iterate over the remaining rows and columns of the table, filling them in based on whether the weight of an item is less than or equal to the current capacity.

If the weight is less than or equal to the current capacity, use the maximum value that can be obtained by either including or excluding the item. Otherwise, exclude the item and use the previous best value at that capacity.

Return the value in the last row and column of the table, which represents the maximum value that can be obtained with the given capacity and items.

Reset Show Solution Submit

Try it yourself

Implement your solution in the following coding playground:

Java

usercode > FindMaxKnapsackProfit.java

```
import java.util.*;

```

```
        return 0;
    }
}

```

Powered by AI Submit

Test Cases Results

Case 1 Case 2 Case 3

Input #1

```
6
```

Input #2

```
[1,2,3,5]
```

Input #3

```
[1,5,4,8]
```

0/1 Knapsack

← Back

Dynamic Programmin...

Next →

Solution: 0/1 Knapsack

☑ Mark as Completed