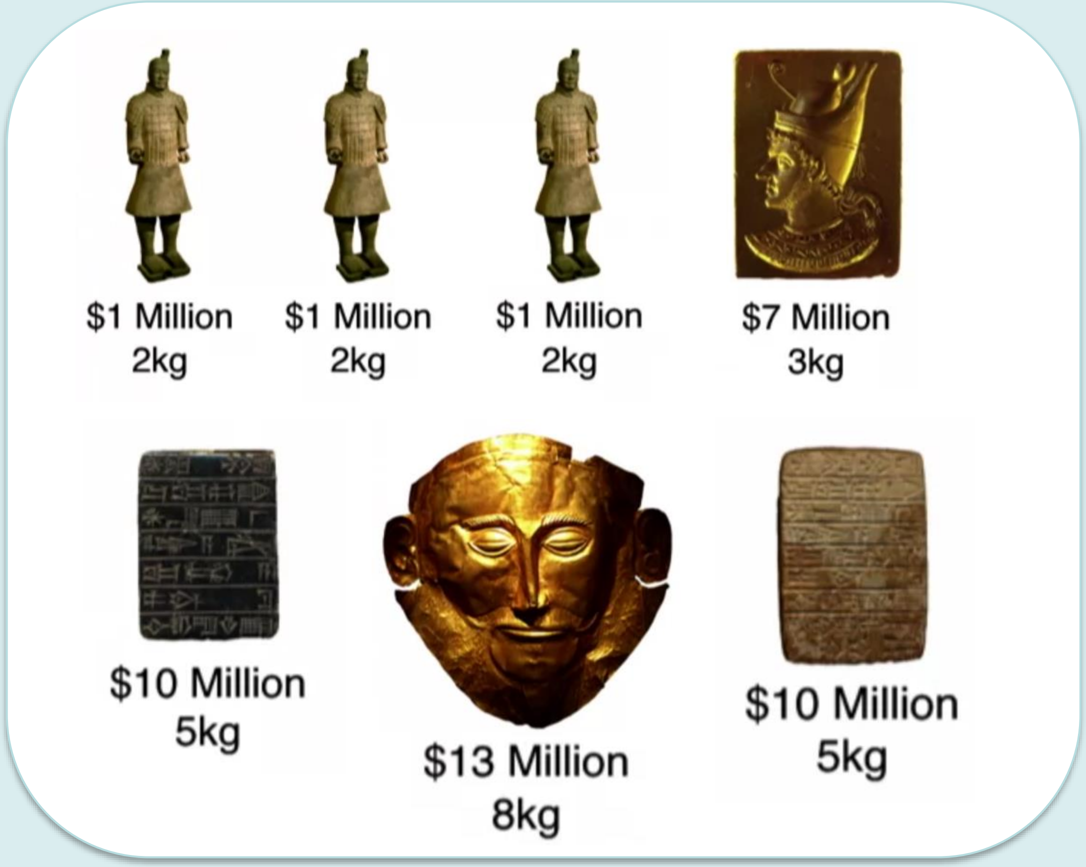
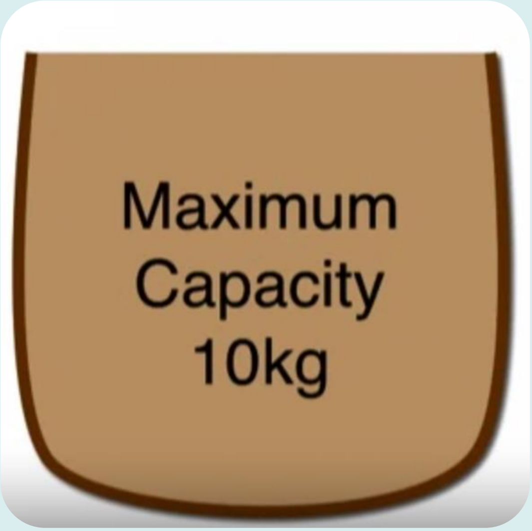
Das Rucksack Problem

Ein Rucksack hat ein Fassungsvermögen von genau 10 kg. Welche Items sollte man einpacken, damit der Gesamtwert **maximal** wird?

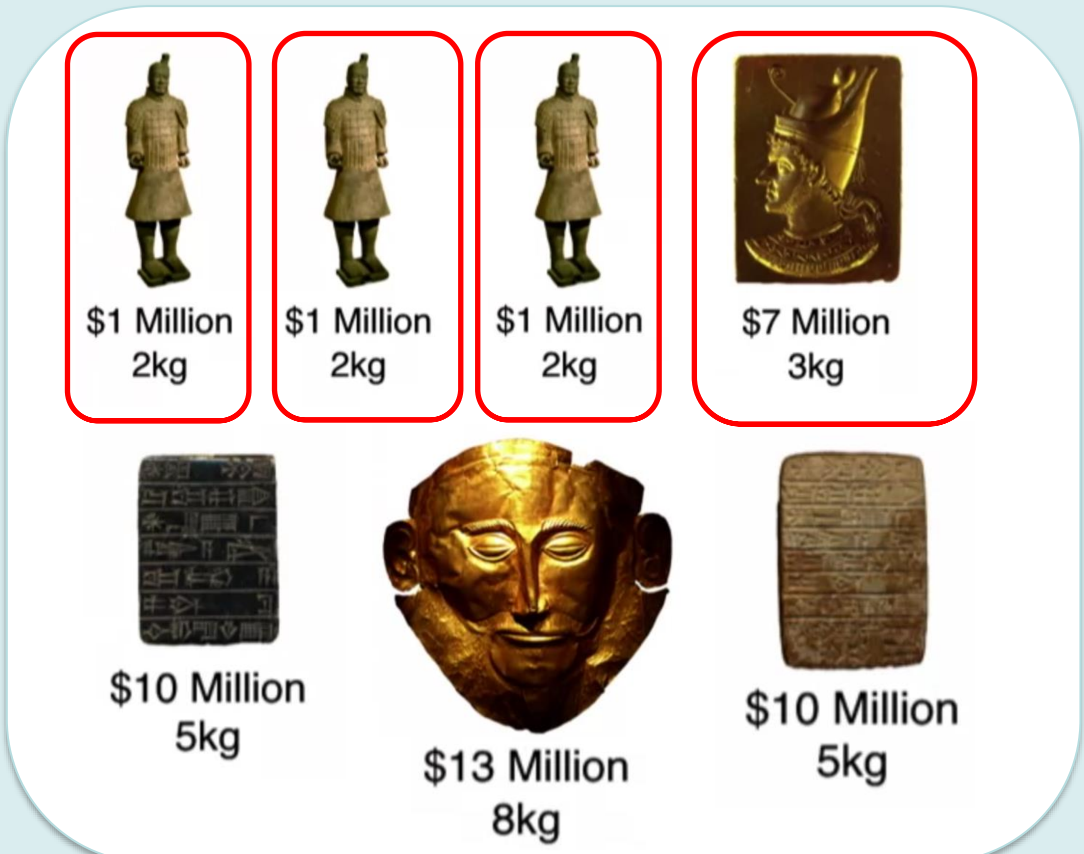


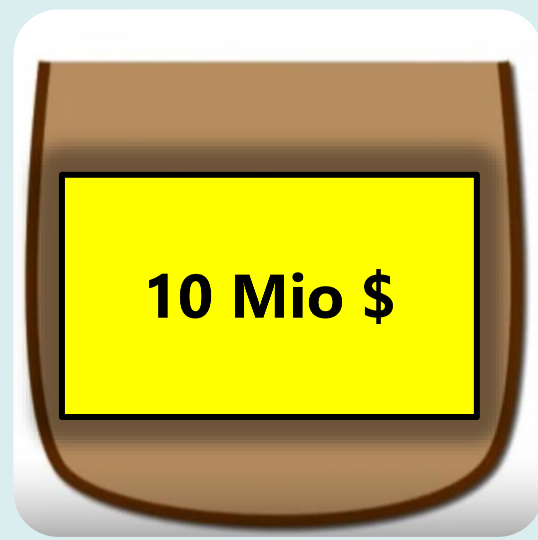


Rucksack Problem – **Scheiternde** Greedy Algorithmen

Idee 1: **Je mehr Items im Rucksack desto besser**

⇒ Beginne mit dem **leichtesten** Item und packe so viele ein wie möglich

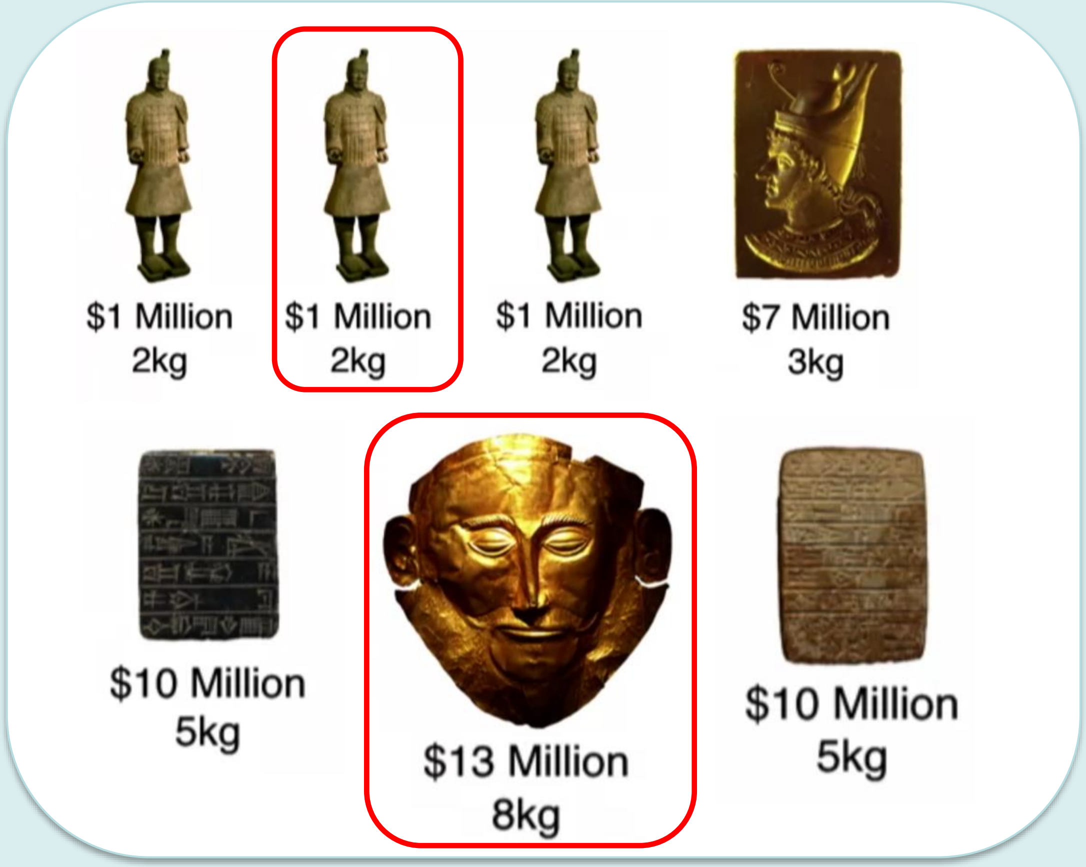




Rucksack Problem – **Scheiternde** Greedy Algorithmen

Idee 2: **Je wertvoller die Items im Rucksack desto besser**

⇒ Nimm (falls möglich) zuerst das **wertvollste** Item, dann das zweitwertvollste etc.

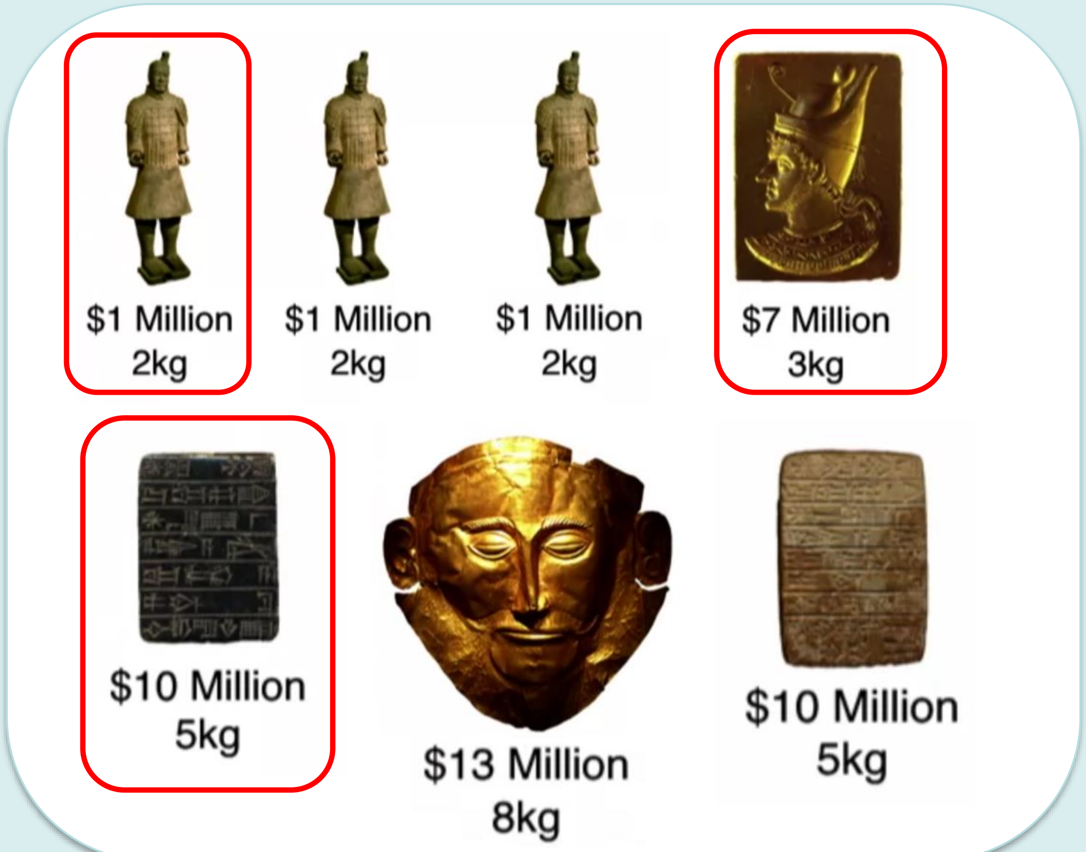


Rucksack Problem – **Scheiternde** Greedy Algorithmen

Idee 3:

Wähle die Items in der Reihenfolge ihre **Wertedichte** aus, d.h. nach ihrem „Wert pro Kilogramm“

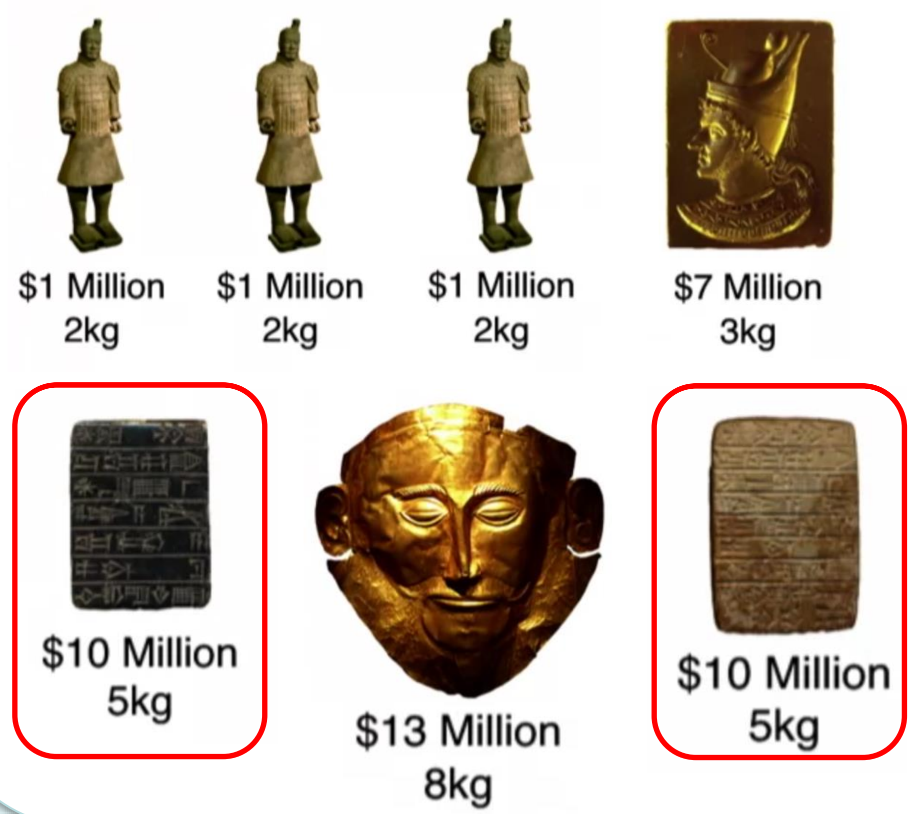
Bester Greedy Algorithmus
bei großer Rucksack-Kapazität

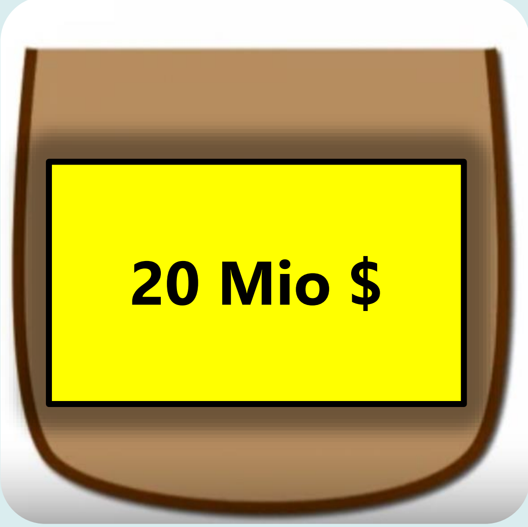


Rucksack Problem – **Scheiternde** Greedy Algorithmen

Beste Lösung

Wie findet man diese Lösung?





Mathematische Formulierung des Problems

Gegeben sei eine Menge von Items $\boldsymbol{I}$

Das i-te Item habe den Wert $\boldsymbol{v_i}$ und das Gewicht $\boldsymbol{w_i}$

Wähle eine Teilmenge aus $\boldsymbol{I}$ so aus, dass die Summe

$$\sum_{i \in I} v_i \, x_i \qquad \text{mit} \qquad x_i \in \{0, 1\} \quad (i \in I)$$

x_i zeigt an, ob das Item i ausgewählt wurde oder nicht

1: ausgewählt
0: nicht ausgewählt

maximal wird unter der **Bedingung**

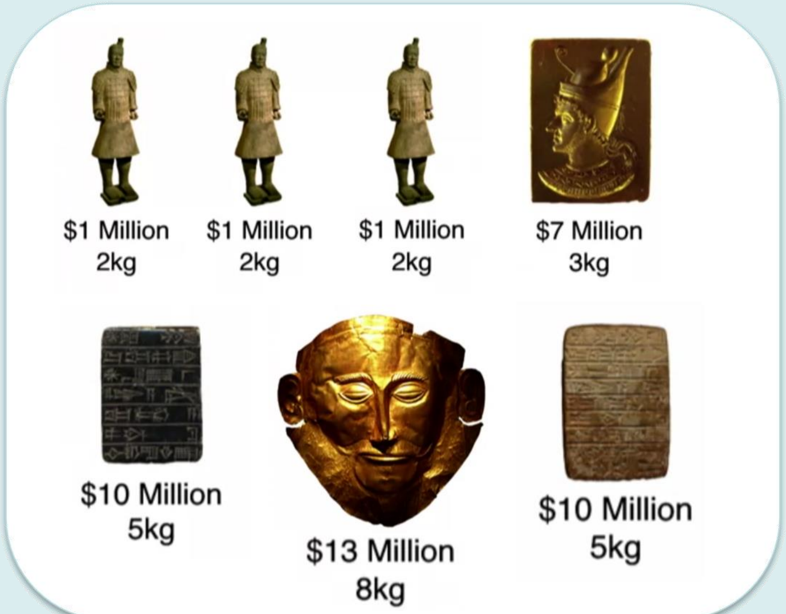
$$\sum_{i \in I} w_i x_i \leq K$$

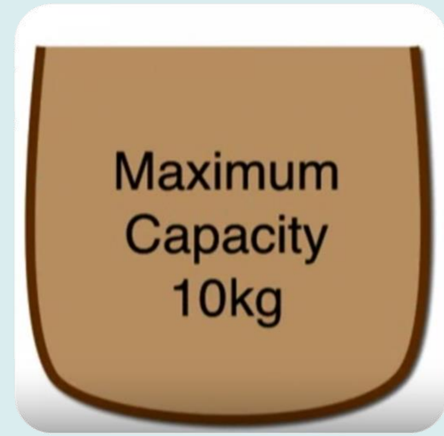
(K = Kapazität des Rucksacks)

Die Lösung des Optimierungsproblems ist ein Vektor $\vec{\mathrm{x}}$

Mathematische Formulierung des Problems

Unser Beispiel:





Math. Formulierung:

Maximiere $1x_1 + 1x_2 + 1x_3 + 7x_4 + 10x_5 + 13x_6 + 10x_7$ (Gesamtwert)

unter der Bedingung $2x_1 + 2x_2 + 2x_3 + 3x_4 + 5x_5 + 8x_6 + 5x_7 \leq 10$ (Gesamtgewicht)

mit $x_i \in \{0, 1\} \quad i \in \{1, 2, \ldots 7\}$

Lösung: $\vec{x} = (0, 0, 0, 0, 1, 0, 1)$ ($\Rightarrow$ Gesamtwert : 20)

Wie viele Lösungen muss ein Brute Force Algo prüfen? 2^7

Dynamic Programming

Idee *(Richard Bellman):*

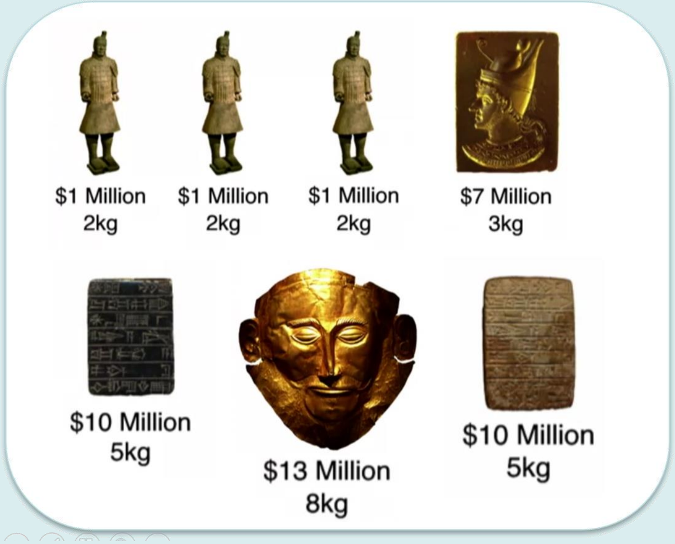
Löse das Optimierungsproblem durch **Aufteilung in Teilprobleme** und systematische **Speicherung von Zwischenresultaten**.

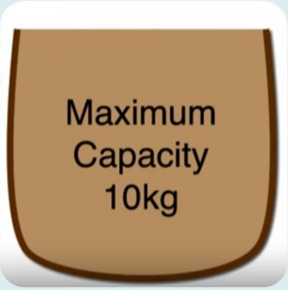
Funktioniert immer, wenn das Optimierungsproblem aus vielen gleichartigen Teilproblemen besteht und eine optimale Lösung sich aus optimalen Lösungen der Teilprobleme zusammensetzt.

Dynamic programming

Idee *(Richard Bellman):*

Löse das Optimierungsproblem durch **Aufteilung in Teilprobleme** und systematische **Speicherung von Zwischenresultaten**.





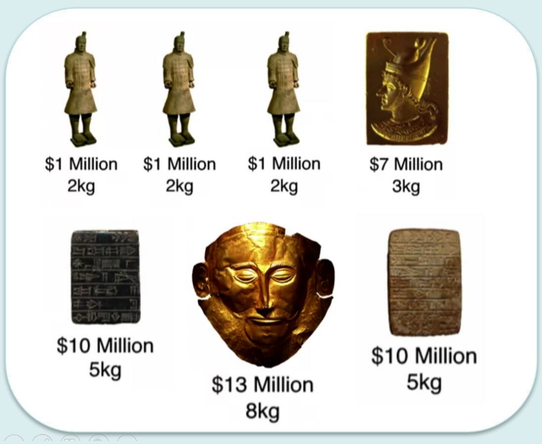
Teilproblem hier:

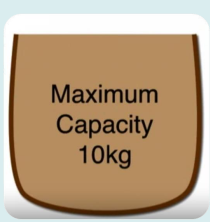
Weniger Items und geringere Kapazität

Dynamic programming

Teilproblem hier:

Weniger Items und geringere Kapazität





Algorithmus

Löse das Problem für **null** Items und Kapazität **0**
Löse das Problem für **null** Items und Kapazität **1**
Löse das Problem für **null** Items und Kapazität **2**
…
Löse das Problem für **null** Items und Kapazität **10**
Löse das Problem für **ein** Item und Kapazität **0**
Löse das Problem für **ein** Item und Kapazität **1**
…
Löse das Problem für **ein** Item und Kapazität **10**
Löse das Problem für **zwei** Items und Kapazität **0**
Löse das Problem für **zwei** Items und Kapazität **1**
…
Löse das Problem für **zwei** Items und Kapazität **10**
…
Löse das Problem für **sieben** Items und Kapazität **9**
Löse das Problem für **sieben** Items und Kapazität **10**

Lege hierfür eine Tabelle an (siehe Arbeitsblatt)

Ein einfaches Beispiel

5 €	6 €	3 €
4 kg	5 kg	2 kg
Item 1	*Item 2*	*Item 3*

Rucksack fasst **9 kg**

Maximiere $5x_1 + 6x_2 + 3x_3$

Bedingung $4x_1 + 5x_2 + 2x_3 \leq 9$ mit $x_i \in \{0, 1\}$ $i \in \{1, 2, 3\}$

Lösung aller Teilprobleme und des Gesamtproblems mit Hilfe einer **Tabelle**!

(siehe prakt. Übung Arbeitsblatt)

Capacity	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	3
3	0	0	0	3
4	0	5	5	5
5	0	5	6	6
6	0	5	6	8
7	0	5	6	9
8	0	5	6	9
9	0	5	11	11
		$v_1 = 5$ $w_1 = 4$	$v_2 = 6$ $w_2 = 5$	$v_3 = 3$ $w_3 = 2$

Ein einfaches Beispiel

Welche Items wurden ausgewählt?

Capacity	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	3
3	0	0	0	3
4	0	5	5	5
5	0	5	6	6
6	0	5	6	8
7	0	5	6	9
8	0	5	6	9
9	0	5	11	11
		$v_1 = 5$ $w_1 = 4$	$v_2 = 6$ $w_2 = 5$	$v_3 = 3$ $w_3 = 2$

Rucksack fasst **9 kg**

5 €
4 kg

Item 1

6 €
5 kg

Item 2

3 €
2 kg

Item 3

Ein einfaches Beispiel

Welche Items wurden ausgewählt? **Backtracing!!**

Rucksack fasst **9 kg**

Capacity	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	3
3	0	0	0	3
4	0	5	5	5
5	0	5	6	6
6	0	5	6	8
7	0	5	6	9
8	0	5	6	9
9	0	5	11	11
		$v_1 = 5$ $w_1 = 4$	$v_2 = 6$ $w_2 = 5$	$v_3 = 3$ $w_3 = 2$

5 €
4 kg

Item 1

6 €
5 kg

Item 2

3 €
2 kg

Item 3

Item 1 und Item 2

Weitere Beispiele

Siehe Übungsblatt

Übungen

a)

Item 1	Item 2	Item 3	
5 €	6 €	3 €	Rucksack fasst **9 kg**
4 kg	5 kg	2 kg	

mathematisch:

Maximiere $5x_1 + 6x_2 + 3x_3$

Bedingung $4x_1 + 5x_2 + 2x_3 \le 9$ mit $x_i \in \{0, 1\}$ $i \in \{1, 2, 3\}$

Item / Capacity	0	1 (v_1 = w_1 =)	2 (v_2 = w_2 =)	3 (v_3 = w_3 =)
0				
1				
2				
3				
4				
5				
6				
7				
8				
9				

b) Maximiere $16x_1 + 19x_2 + 23x_3 + 28x_4$

Bedingung $2x_1 + 3x_2 + 4x_3 + 5x_4 \le 7$ mit $x_i \in \{0, 1\}$ $i \in \{1 \ldots 7\}$

Item / Capacity	0	1 (v_1 = w_1 =)	2 (v_2 = w_2 =)	3 (v_3 = w_3 =)
0				
1				
2				
3				
4				
5				
6				
7				

Auszuwählen sind die Items ____________ ,
der Gesamtwert der Items im Rucksack beträgt dann: ___________