



$$P(x, i) = \begin{cases} P(x, i-1), \\ \max\{P(x, i-1), \dots\} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5
0	1	0	0	0	0	0
1	1	0	0	0	0	0
2	1	0	0	0	0	+13
3						

Algorithmen und Datenstrukturen 2

Sommer 2023

Prof. Dr. Sándor Fekete

Fachgruppenrat Informatik

Studentische Vertretung der Informatik an der TU

Vielfältige Service-Angebote

- Blog: fginfo.tu-bs.de
- Prüfungs- und Lernmaterialien
- Wiki & Knowledge Base

Socializing

- Fachgruppenraum (IZ 149/150)
- Ersti-Einführung und Betreuung
 - Du willst mithelfen? Einladung zur Orga kommt über cs-studs!
- Spiele-, Grillabende, LAN-Partys
- Exkursionen (z.B. Chaos Communication Congress)



Studentische Vertretung der Informatik an der TU

Hochschulpolitik

- Vertretung der studentischen Interessen in Hochschulgremien
- Mitwirken an Weiterentwicklung und Verbesserung des Studiums
- Ansprechpartner*innen bei Problemen
 - Kommt einfach vorbei!

Kontakt:

- Mail: fginfo@tu-bs.de
- Sprechstunde: Dienstags, 15:45
- FG-Treffen: Dienstags, 16:45



Was machen wir?



- coole **Veranstaltungen** planen
 - Erstsemester-Woche
 - Grillen, Spieleabende, monatliche Treffen, ...
- Studierende **unterstützen** und vertreten
 - Du hast Fragen zu deinem Studiengang, einer Prüfung oder sonstigem? Kontaktiere uns einfach und wir helfen dir!
- Eigene, die Lehre ergänzende, Veranstaltungen anbieten



Hast du Lust mitzumachen?

Oder möchtest du einfach nur ein paar coole Leute (uns ;)) kennenlernen, die das gleiche studieren wie du?

Melde dich einfach (völlig unverbindlich) bei uns und schau mal bei einem Treffen vorbei. **Wir freuen uns auf dich!**

Schreib uns einfach eine Mail an fgwinfo@tu-braunschweig.de oder kontaktiere uns über Whatsapp, Instagram o.Ä.



$$P(x, i) = \begin{cases} P(x, i-1), \\ \max\{P(x, i-1), \dots\} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5
0	1	0	0	0	0	0
1	1	0	0	0	0	0
2	1	0	0	0	0	+13
3						

2 Dynamic Programming

*Algorithmen und Datenstrukturen 2
Sommer 2023*

Prof. Dr. Sándor Fekete

Beispiel

Beispiel

Beispiel 1.10.

Partition für $\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$

Gesamtsumme: 240

Beispiel

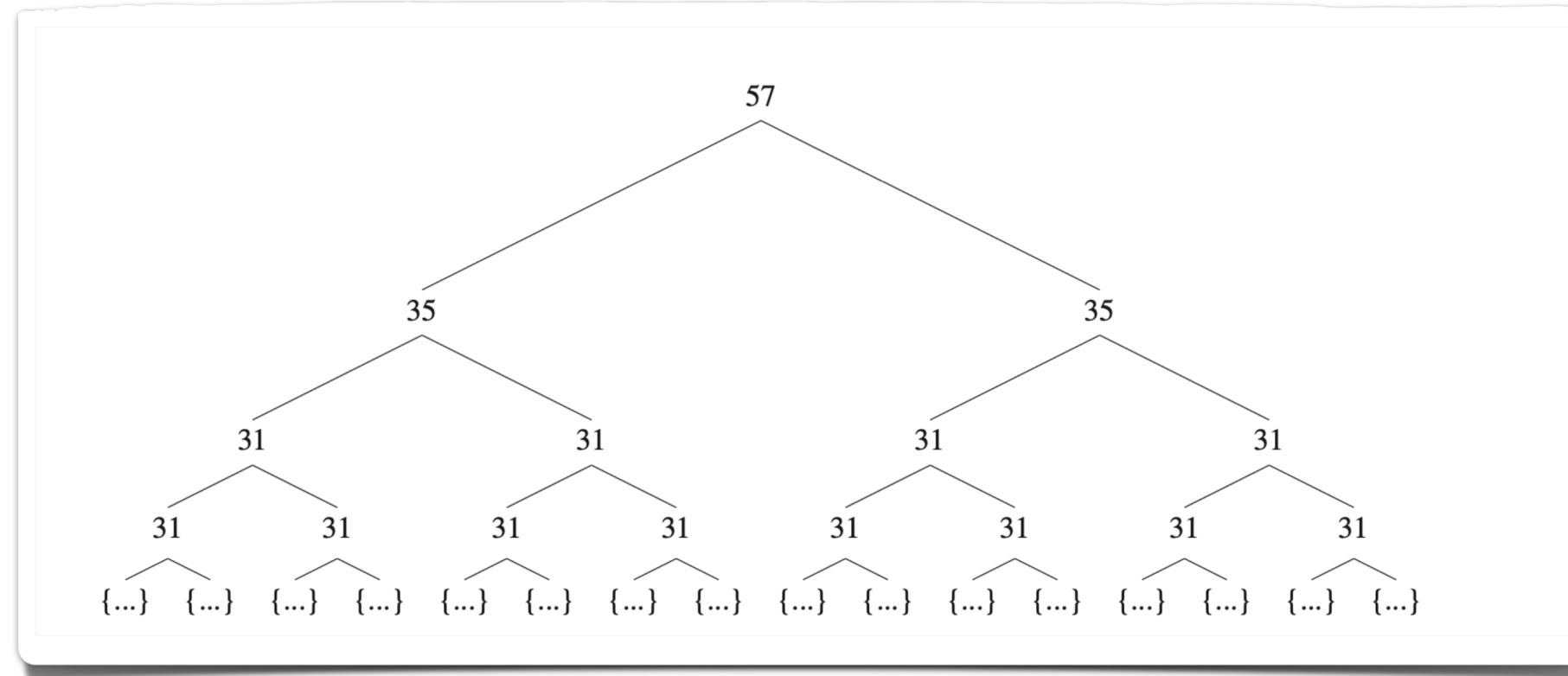
Beispiel 1.10.

Partition für $\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$

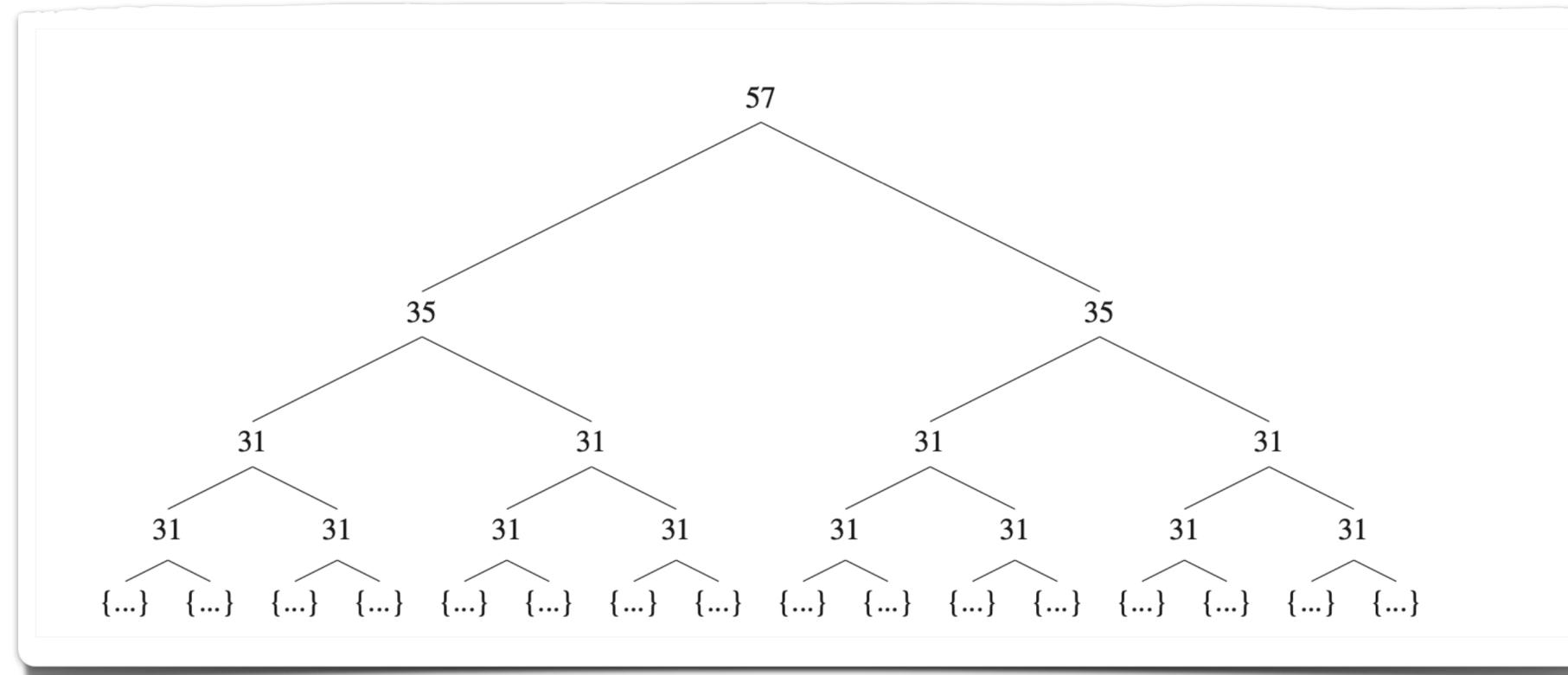
Gesamtsumme: 240

Finde Aufteilung in zwei gleiche Teilmengen!

Enumerationsprinzip

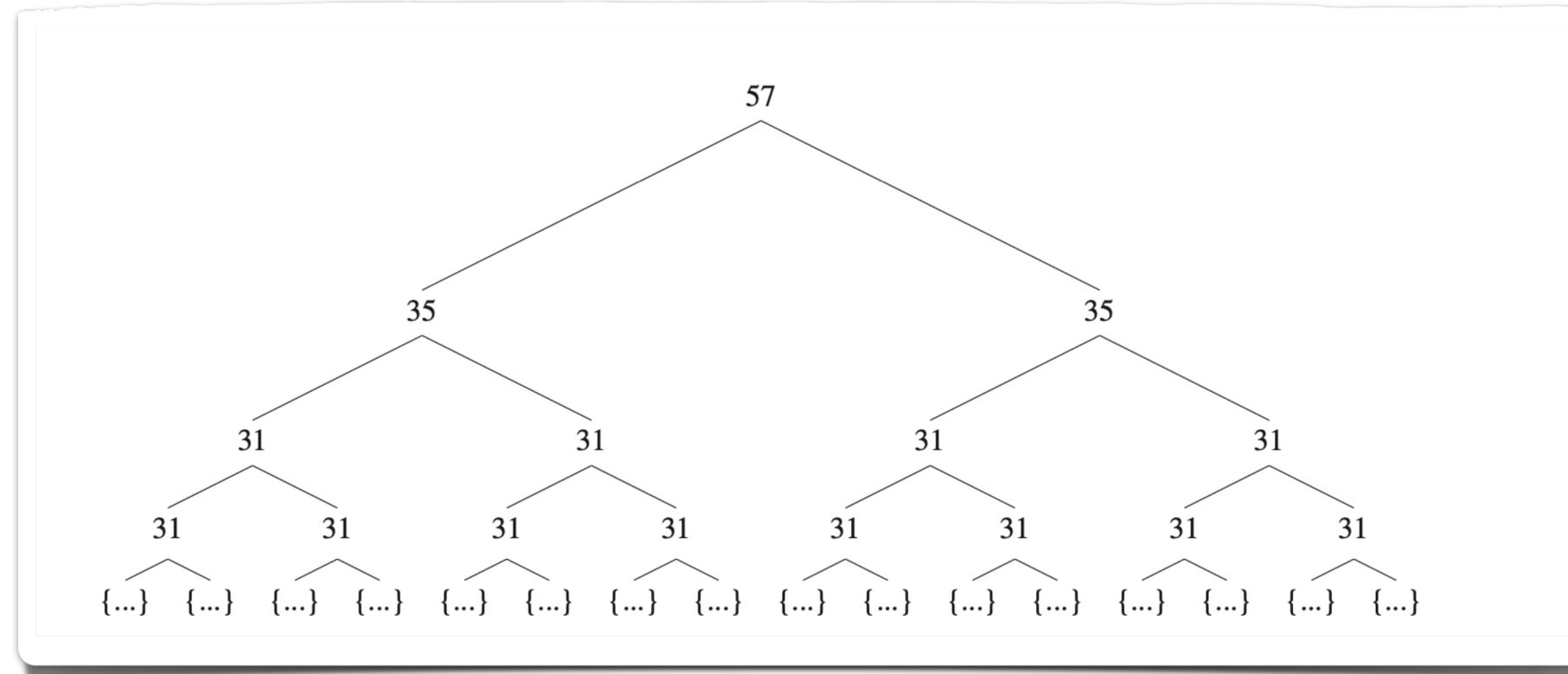


Enumerationsprinzip



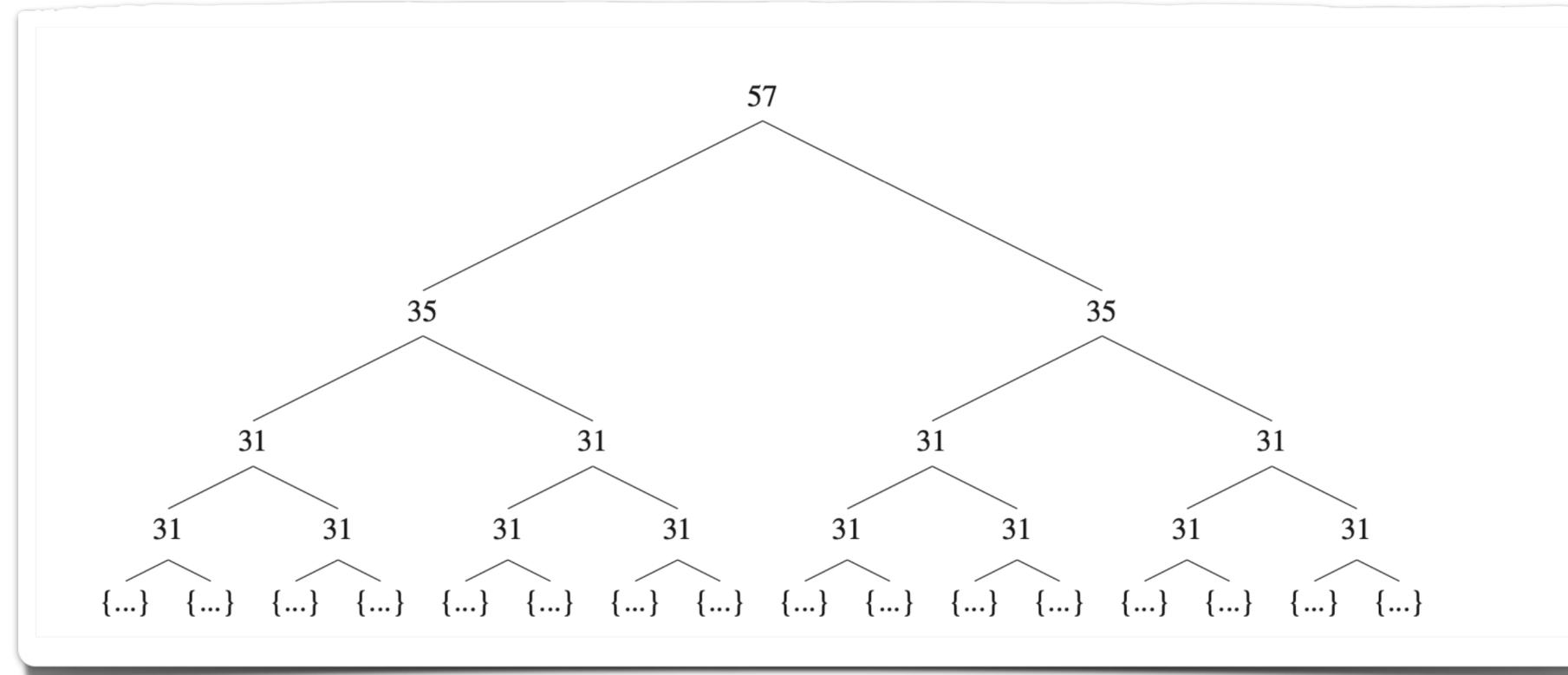
• Exponentiell viele Fälle!

Enumerationsprinzip



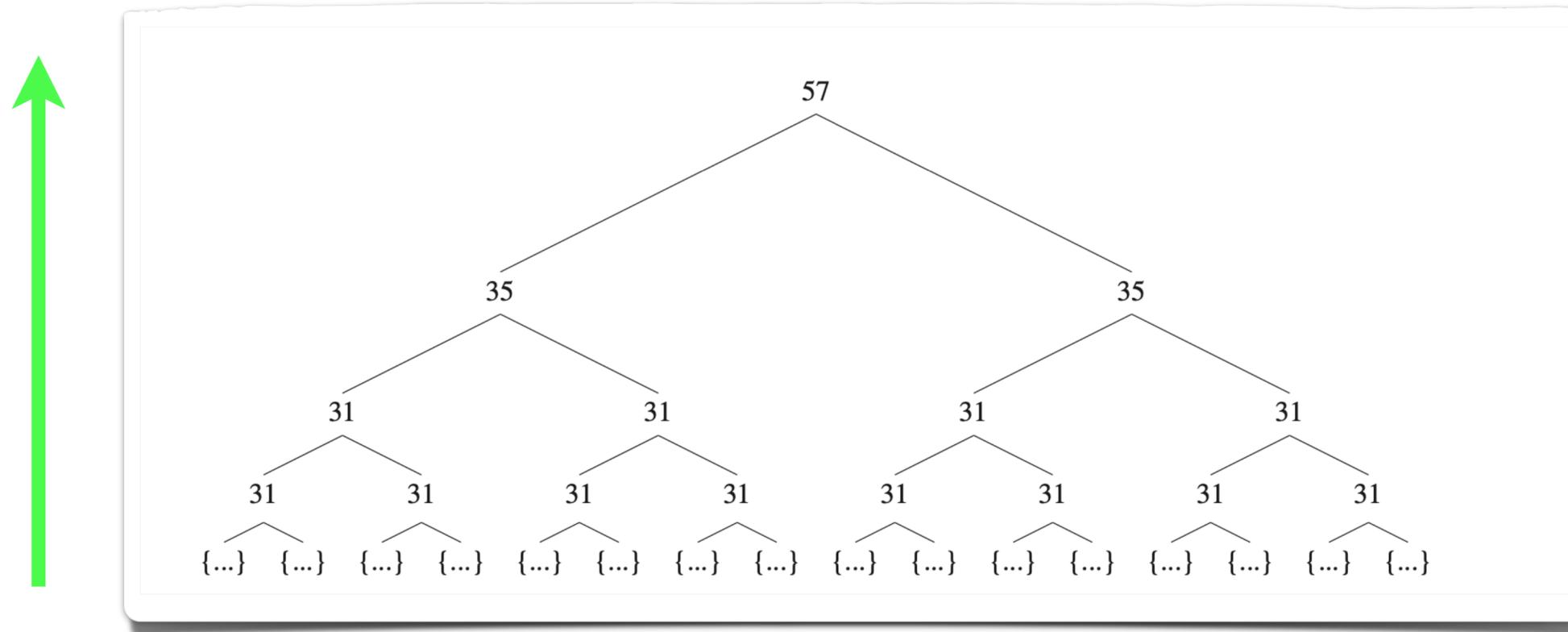
- Exponentiell viele Fälle!
- Wie geht das systematisch?

Enumerationsprinzip



- Exponentiell viele Fälle!
- Wie geht das systematisch?
- Wo kann man Arbeit sparen?

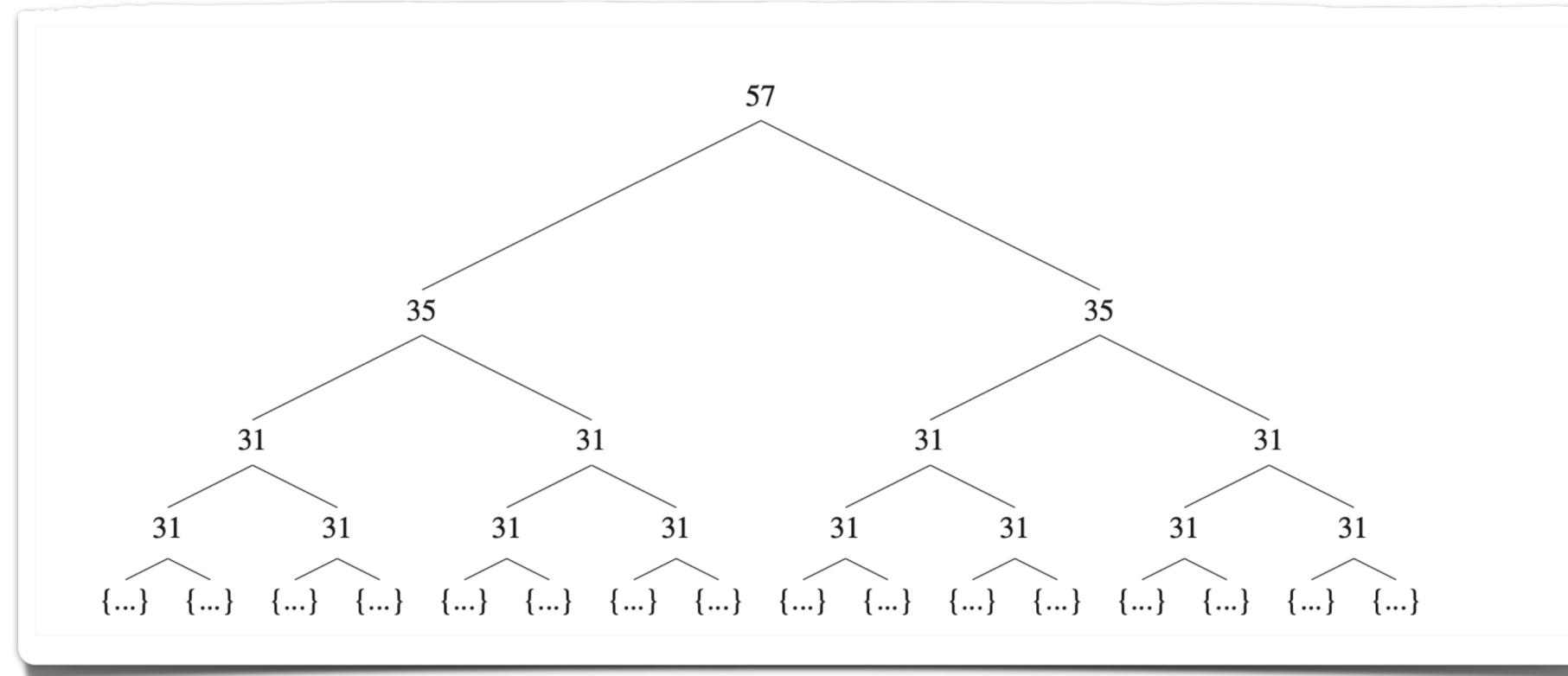
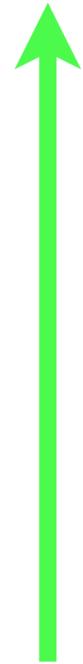
Enumerationsprinzip



- Exponentiell viele Fälle!
- Wie geht das systematisch?
- Wo kann man Arbeit sparen?

Enumerationsprinzip

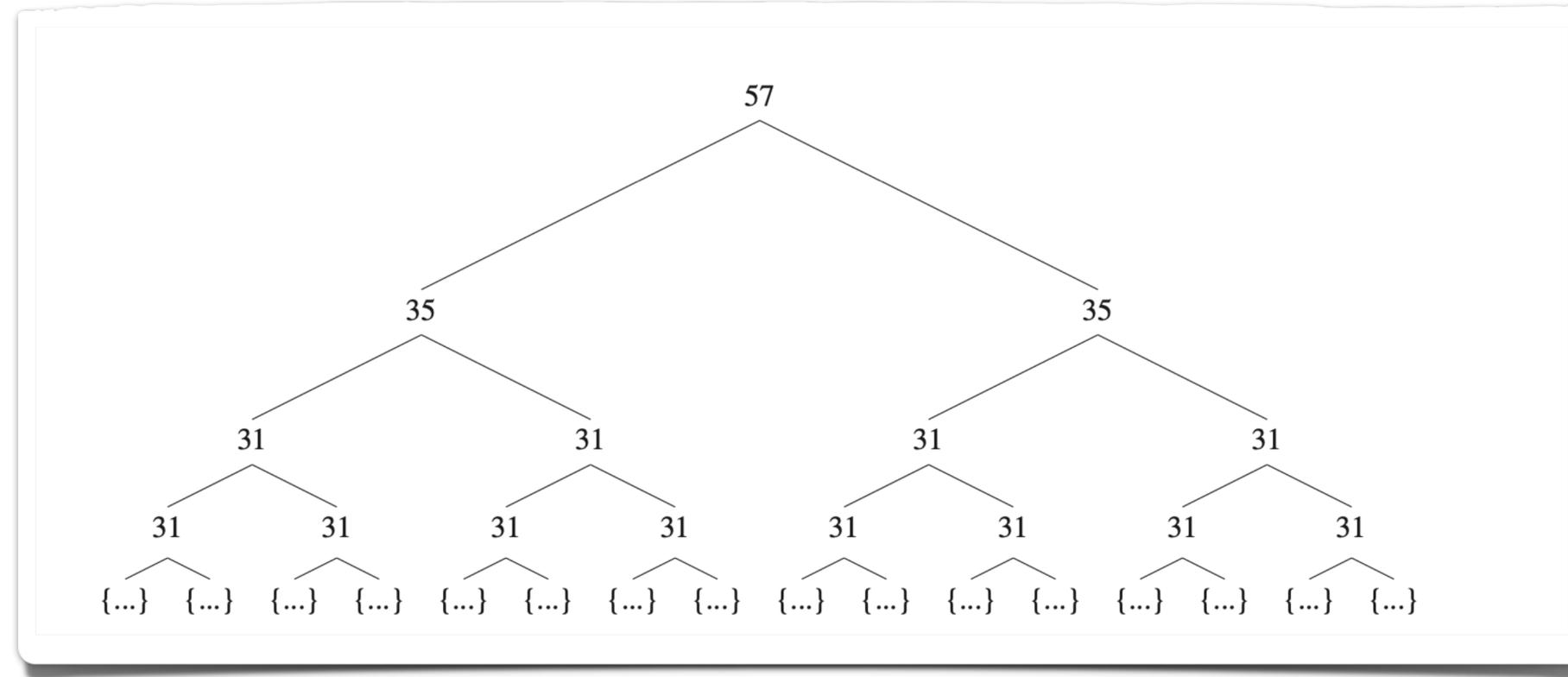
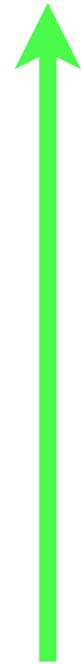
Dynamic
Programming



- Exponentiell viele Fälle!
- Wie geht das systematisch?
- Wo kann man Arbeit sparen?

Enumerationsprinzip

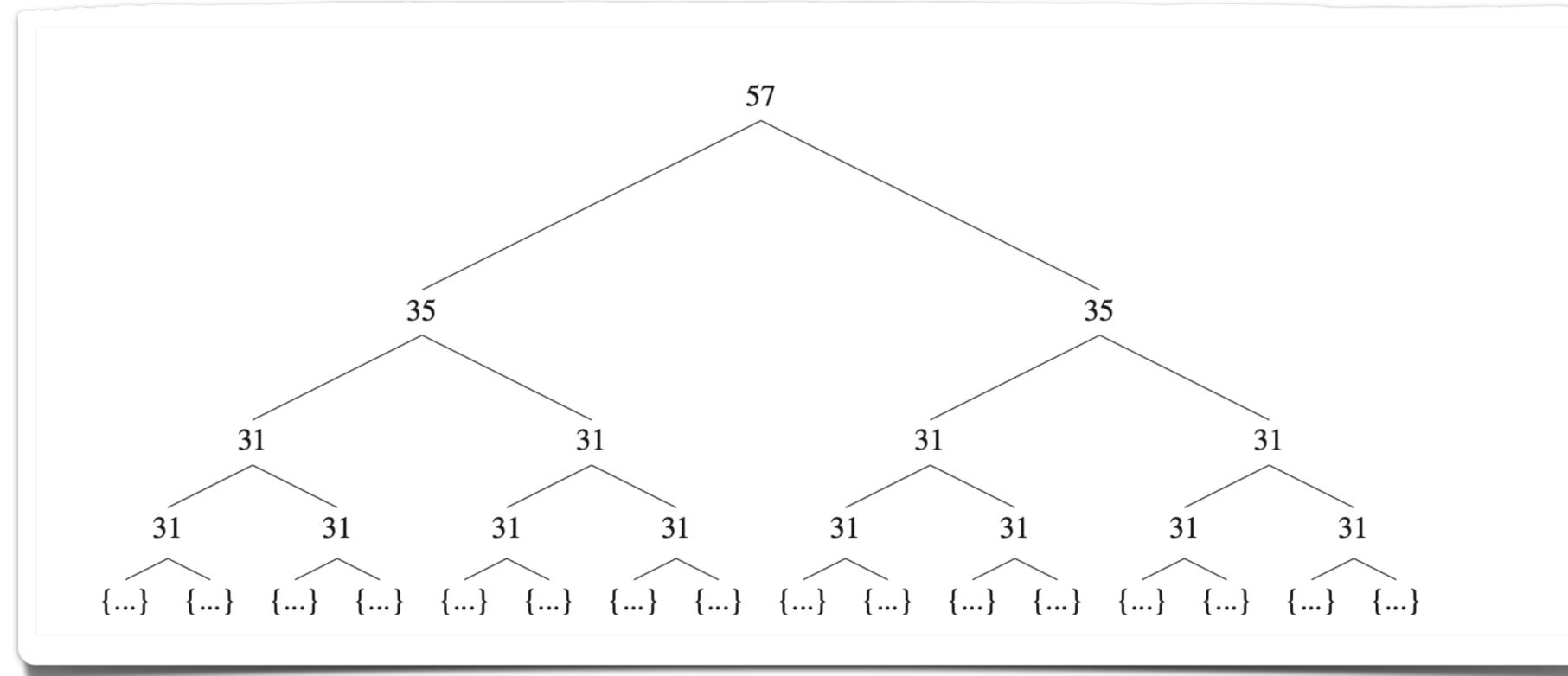
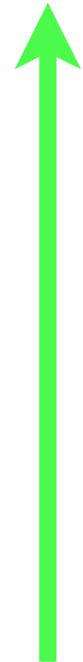
Dynamic
Programming



- Exponentiell viele Fälle!
- Wie geht das systematisch?
- Wo kann man Arbeit sparen?

Enumerationsprinzip

Dynamic
Programming



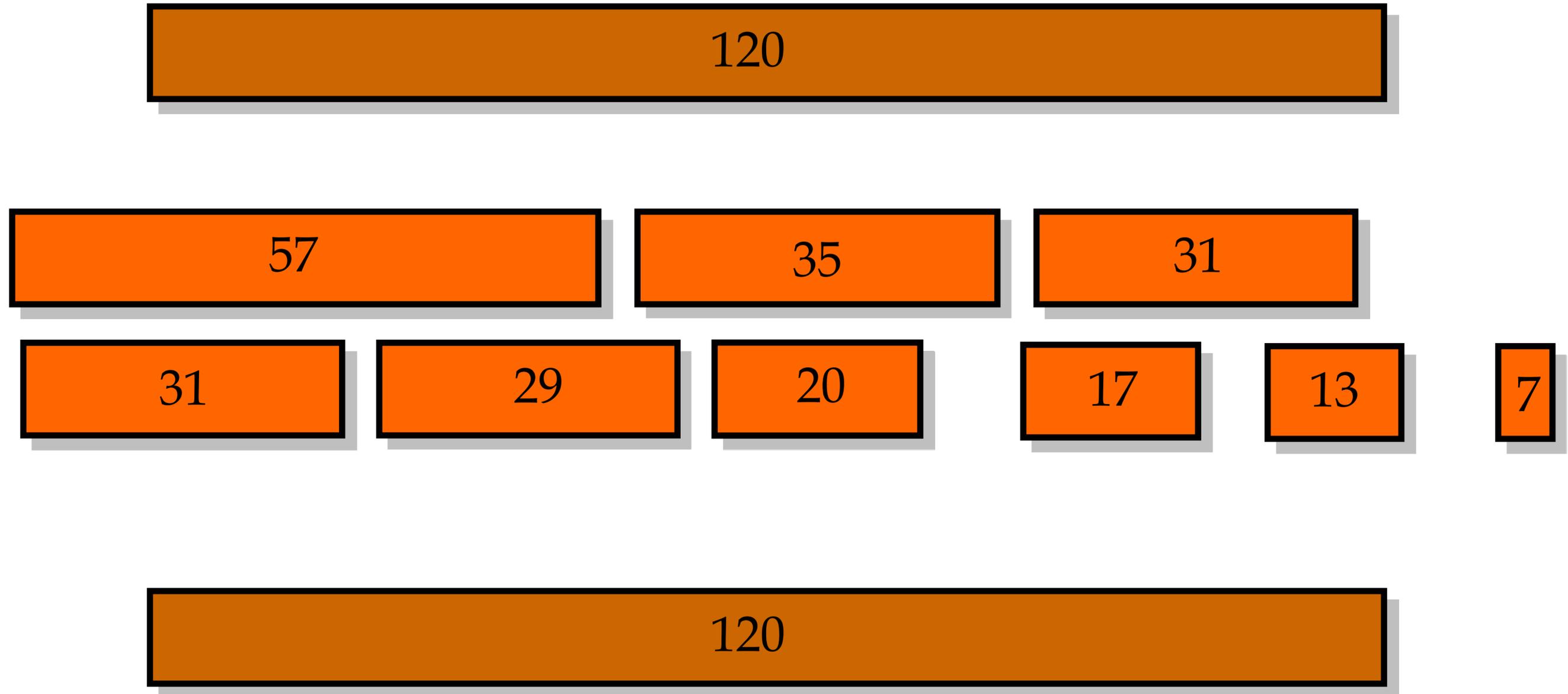
Branch and
Bound



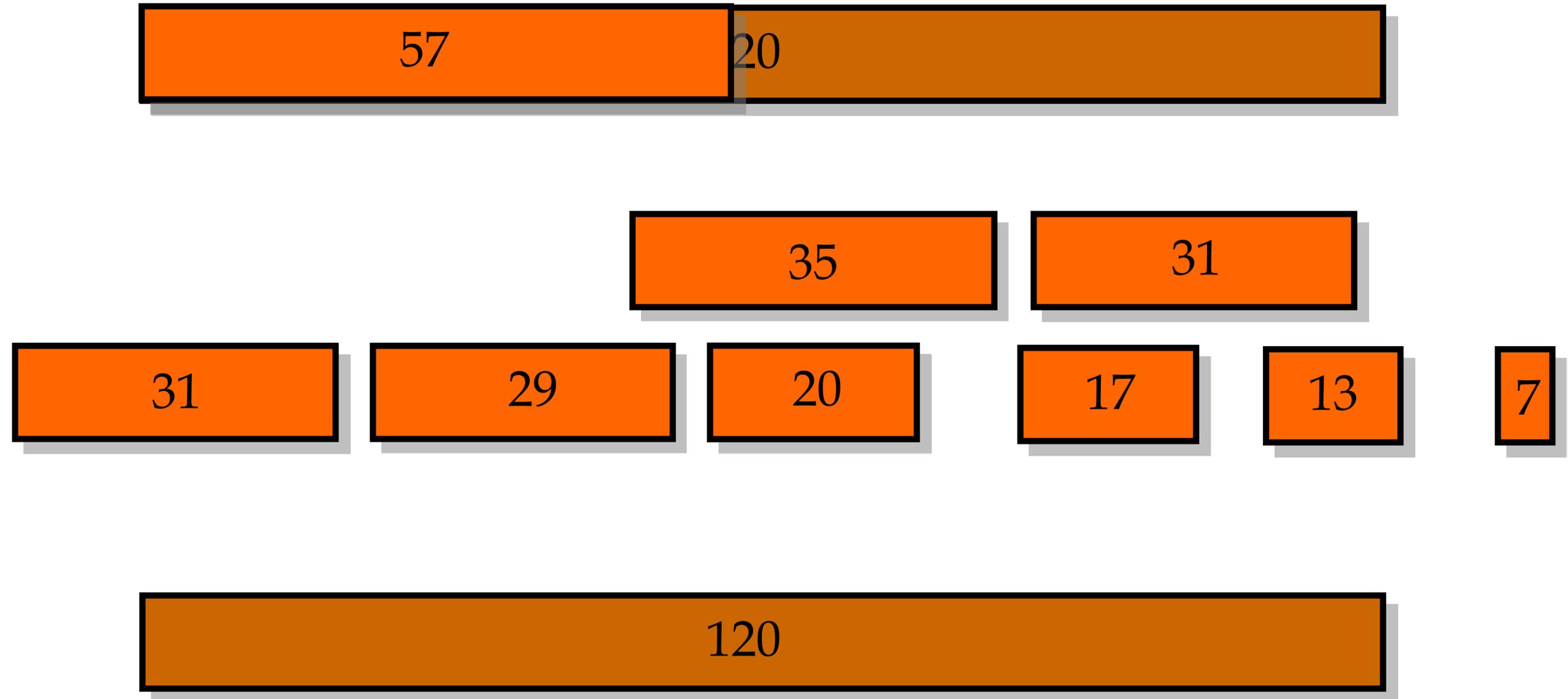
- Exponentiell viele Fälle!
- Wie geht das systematisch?
- Wo kann man Arbeit sparen?

Keine Lösung?!

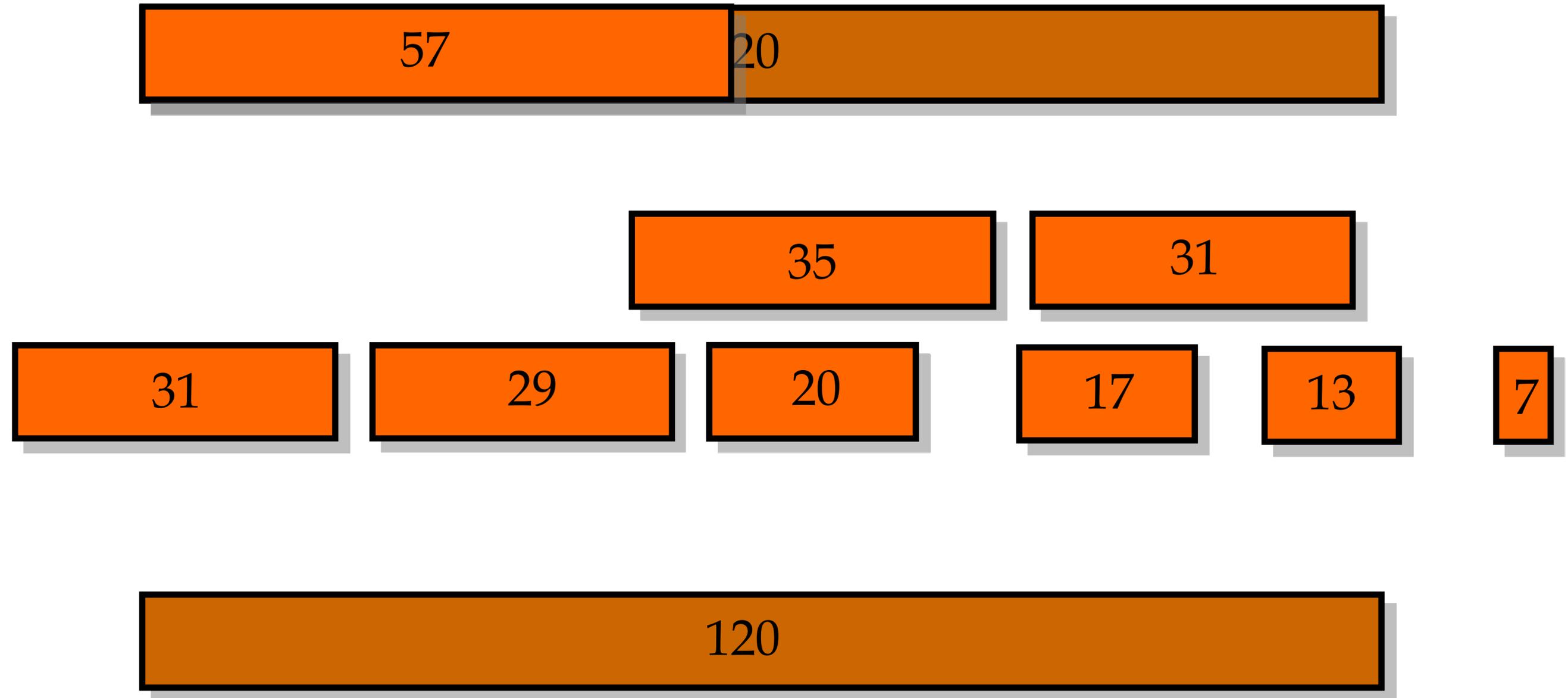
Keine Lösung?!



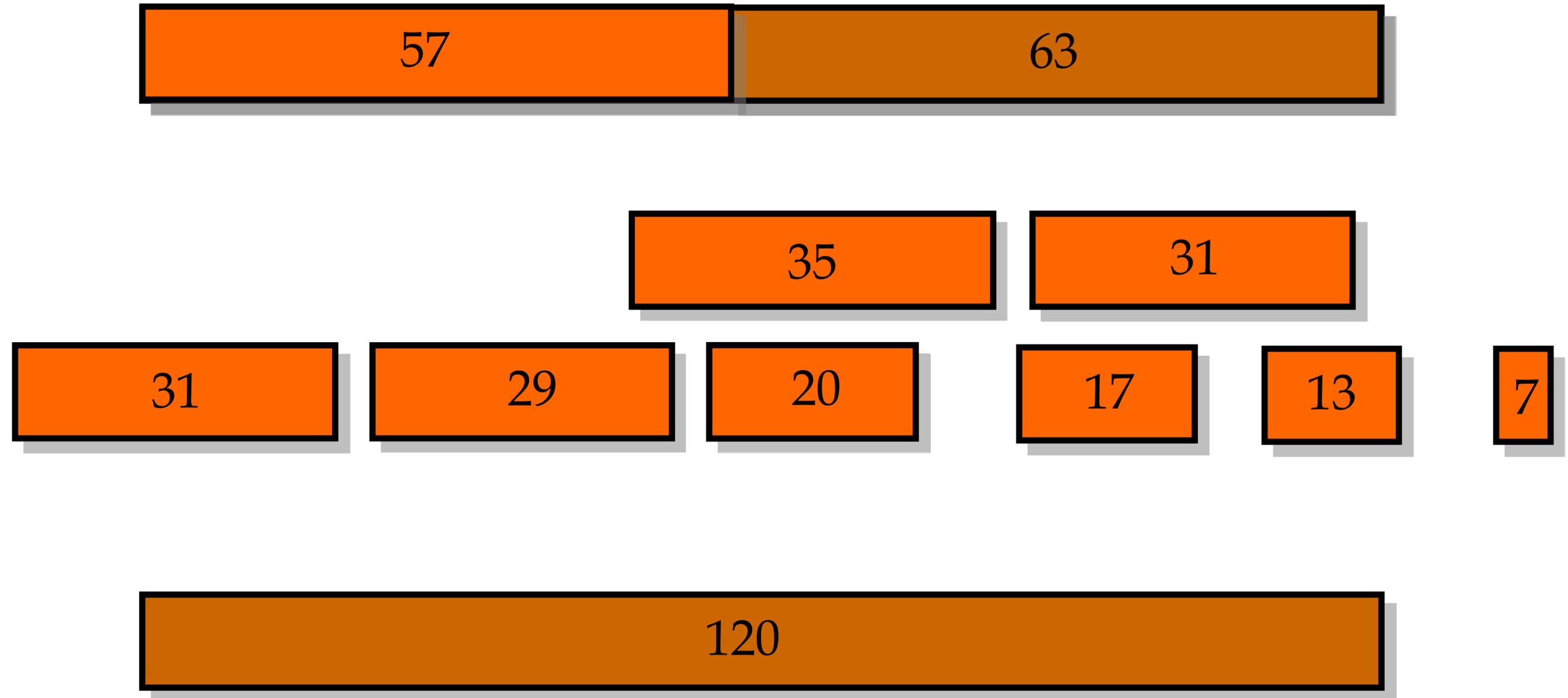
Keine Lösung?!



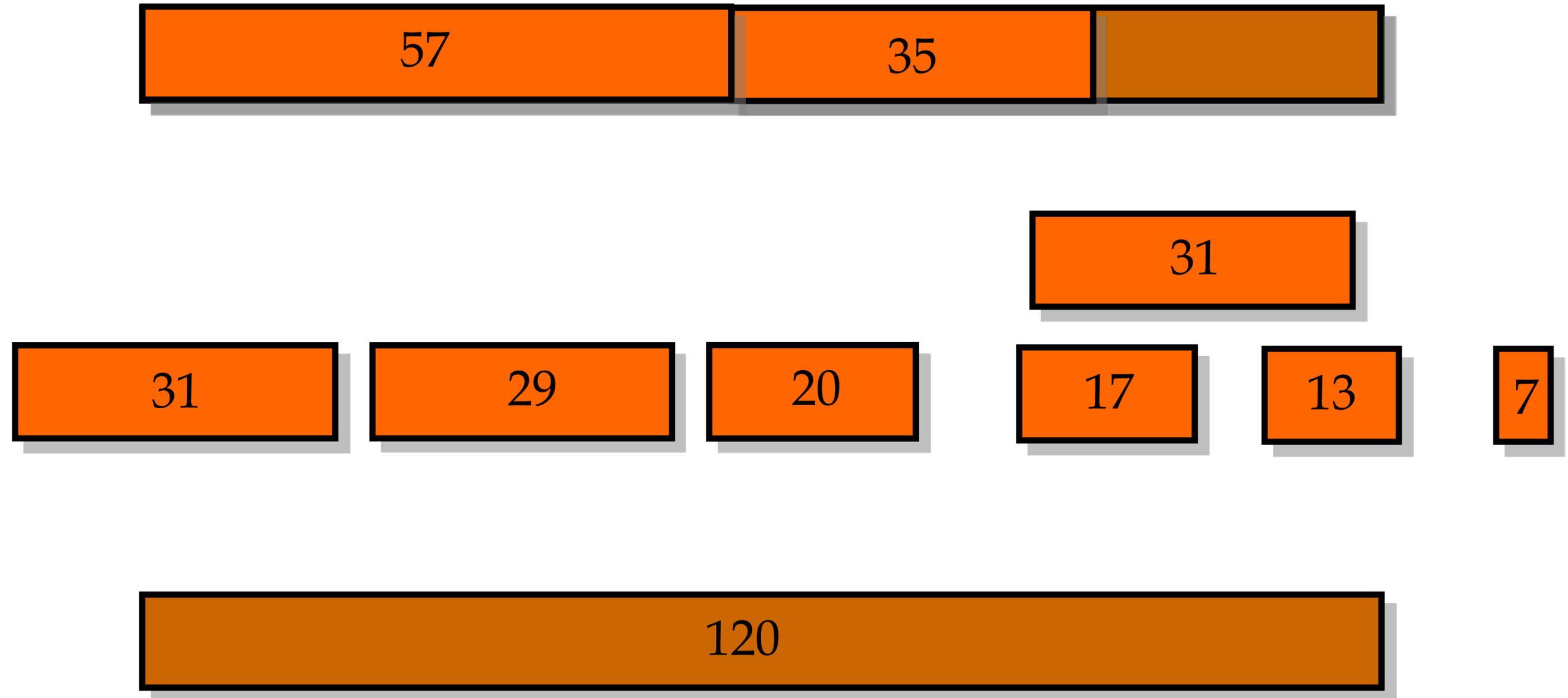
Keine Lösung?!



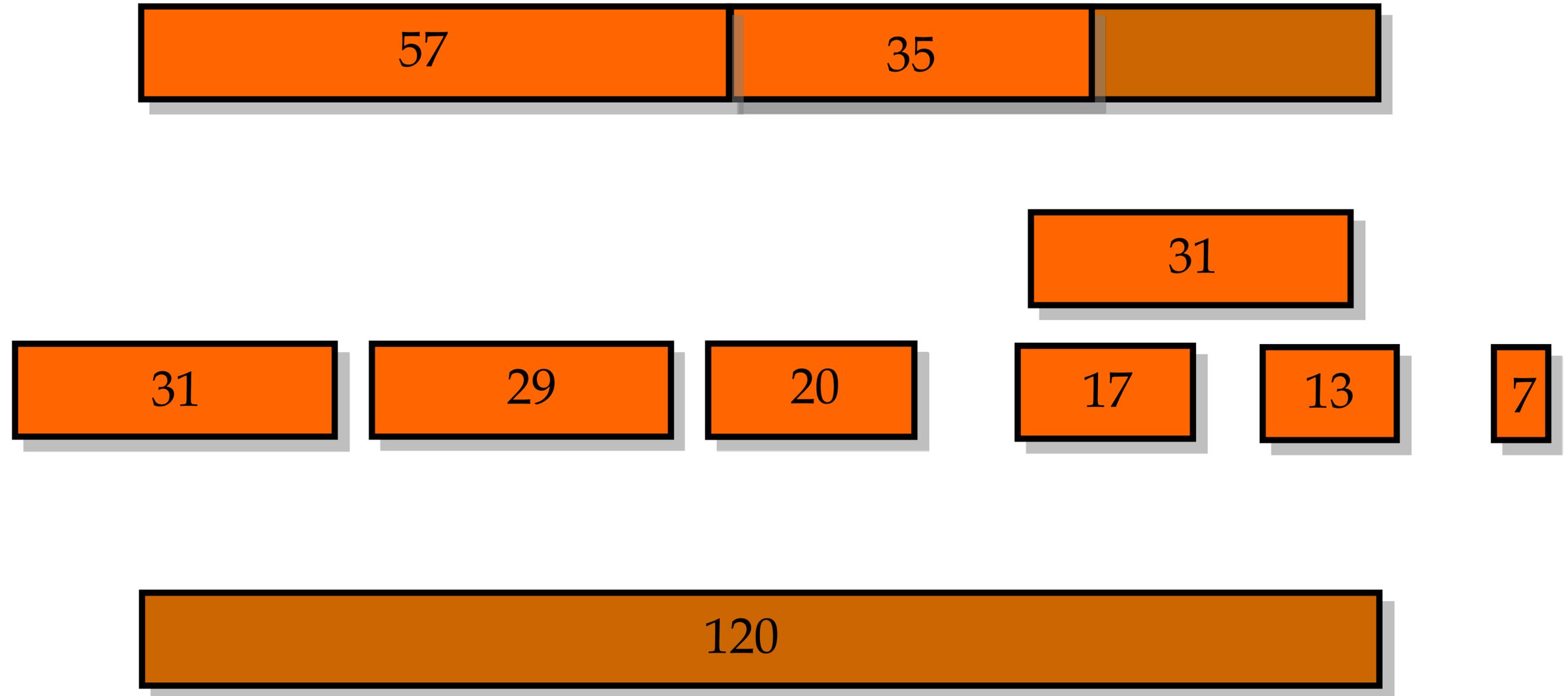
Keine Lösung?!



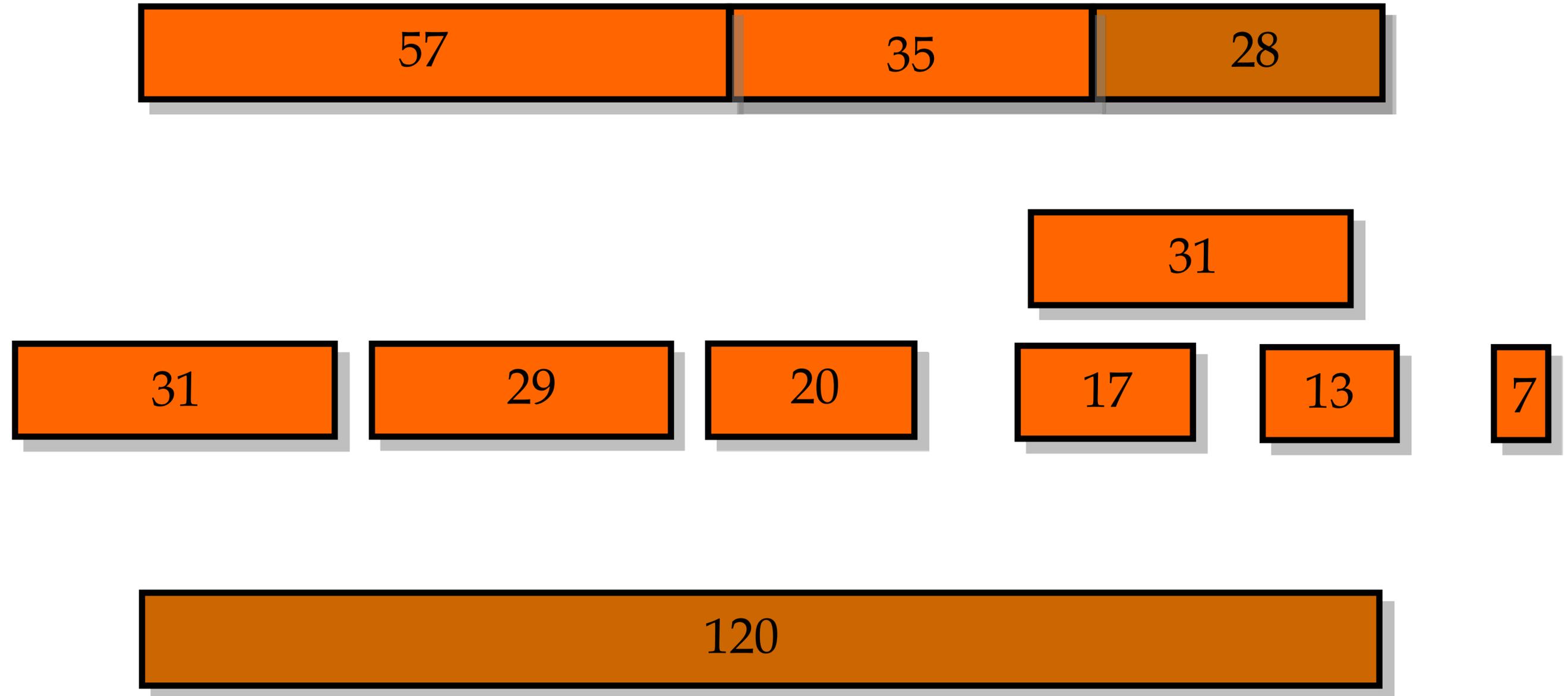
Keine Lösung?!



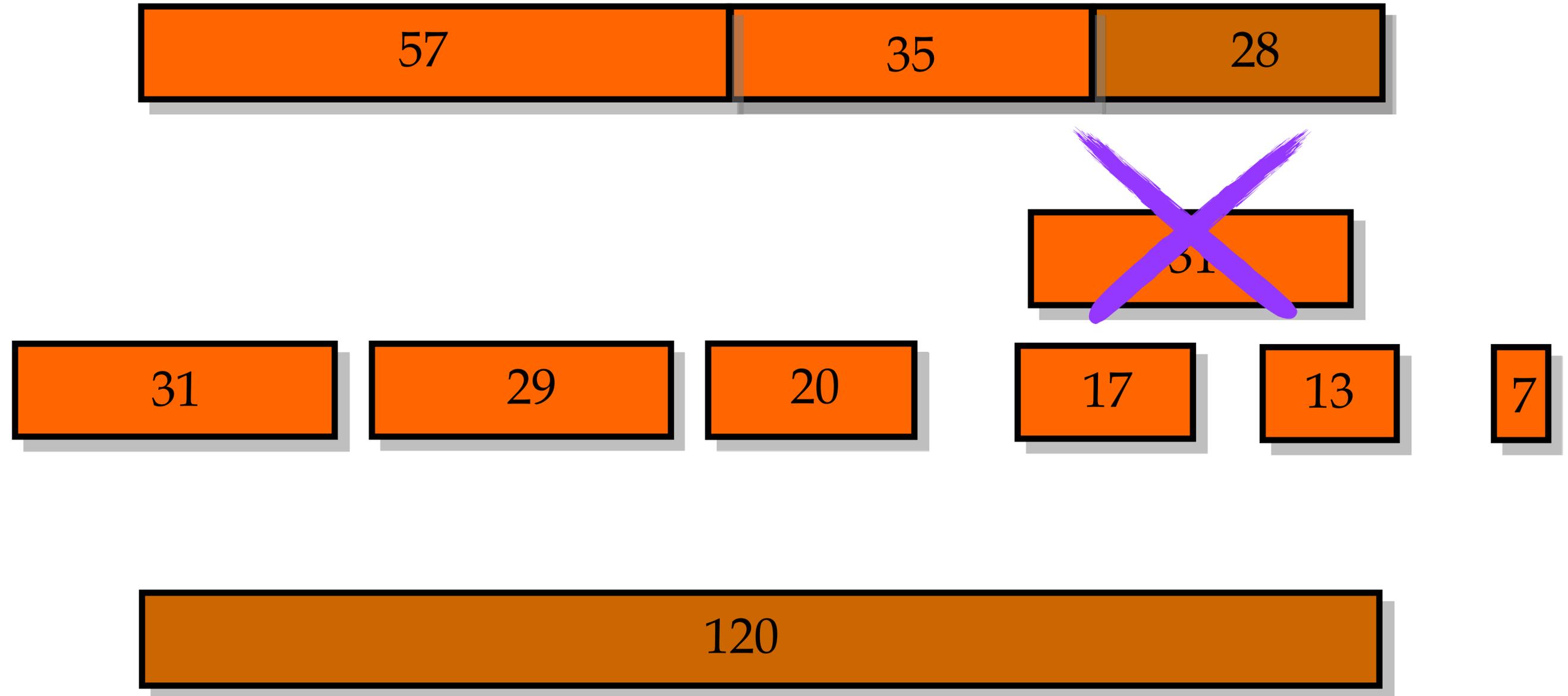
Keine Lösung?!



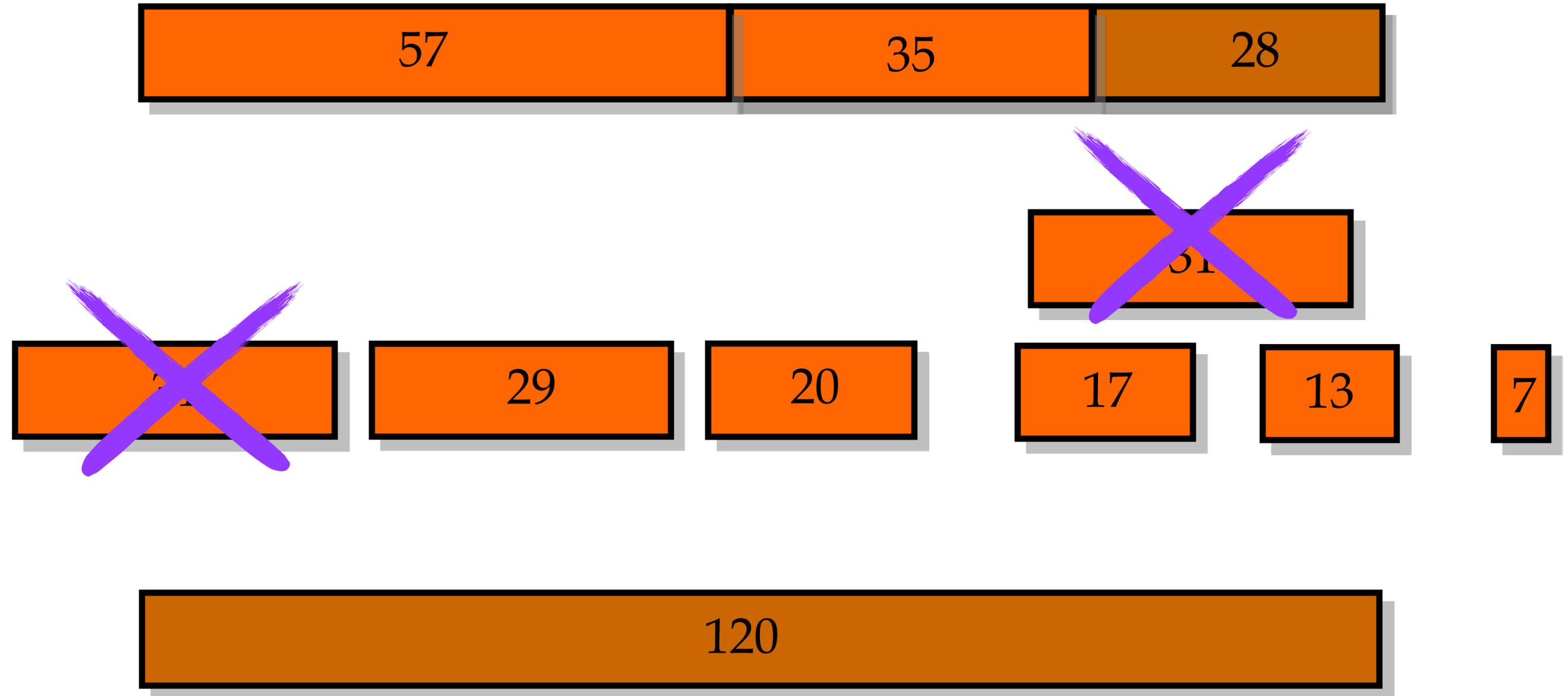
Keine Lösung?!



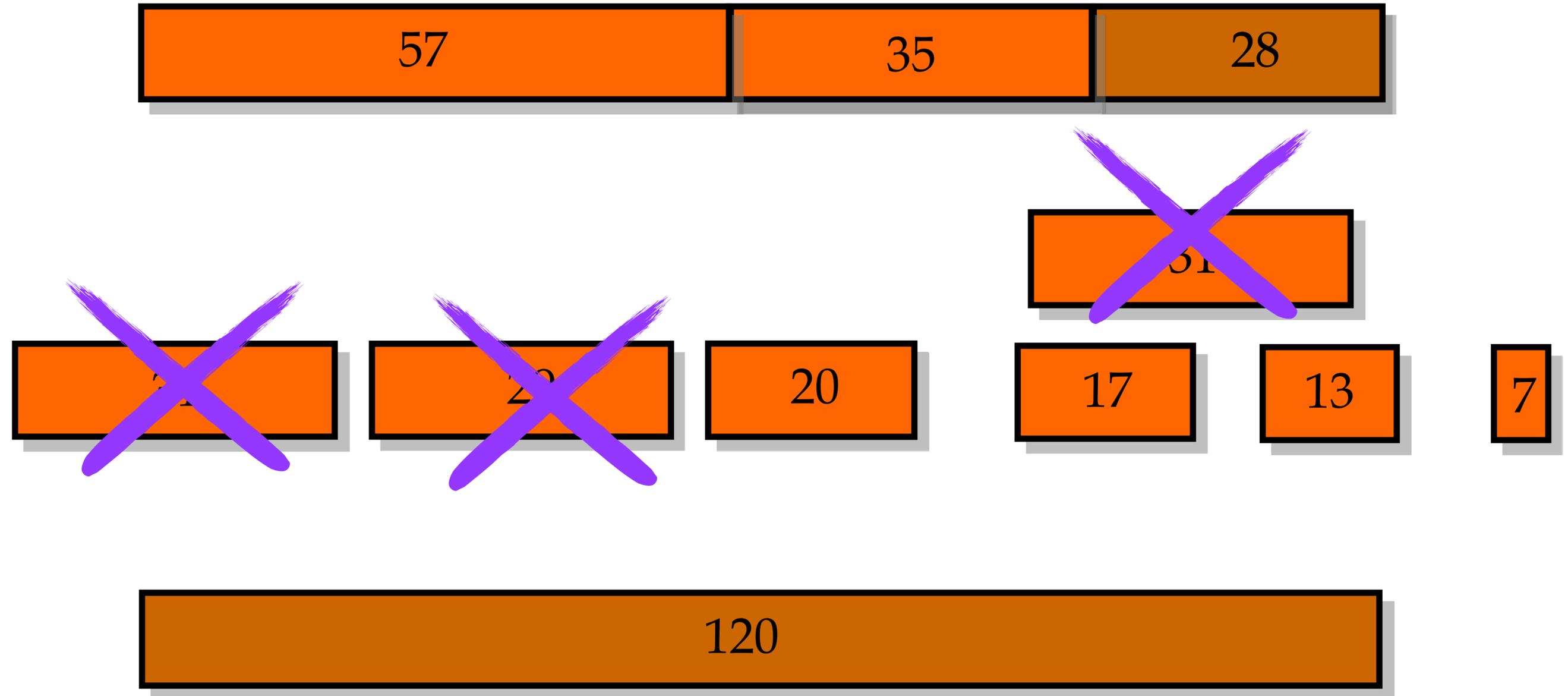
Keine Lösung?!



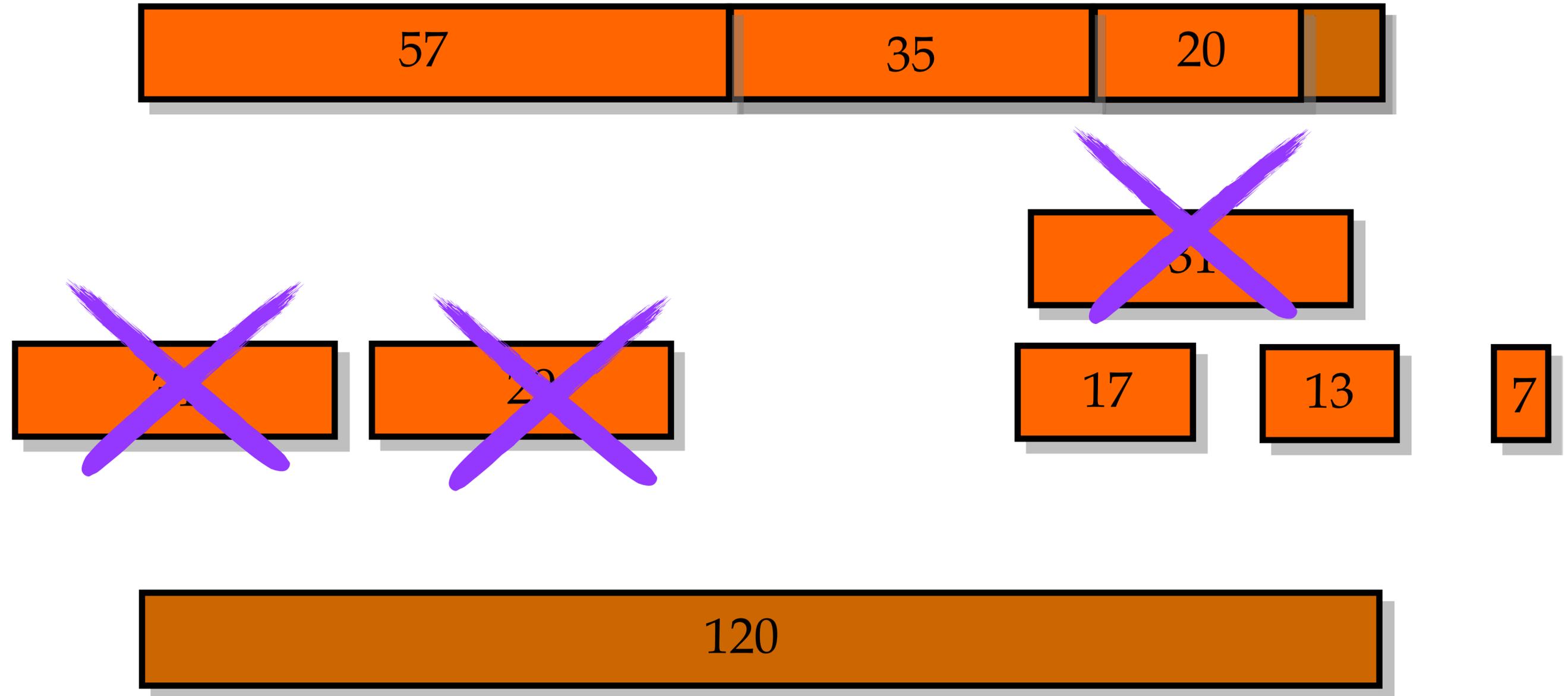
Keine Lösung?!



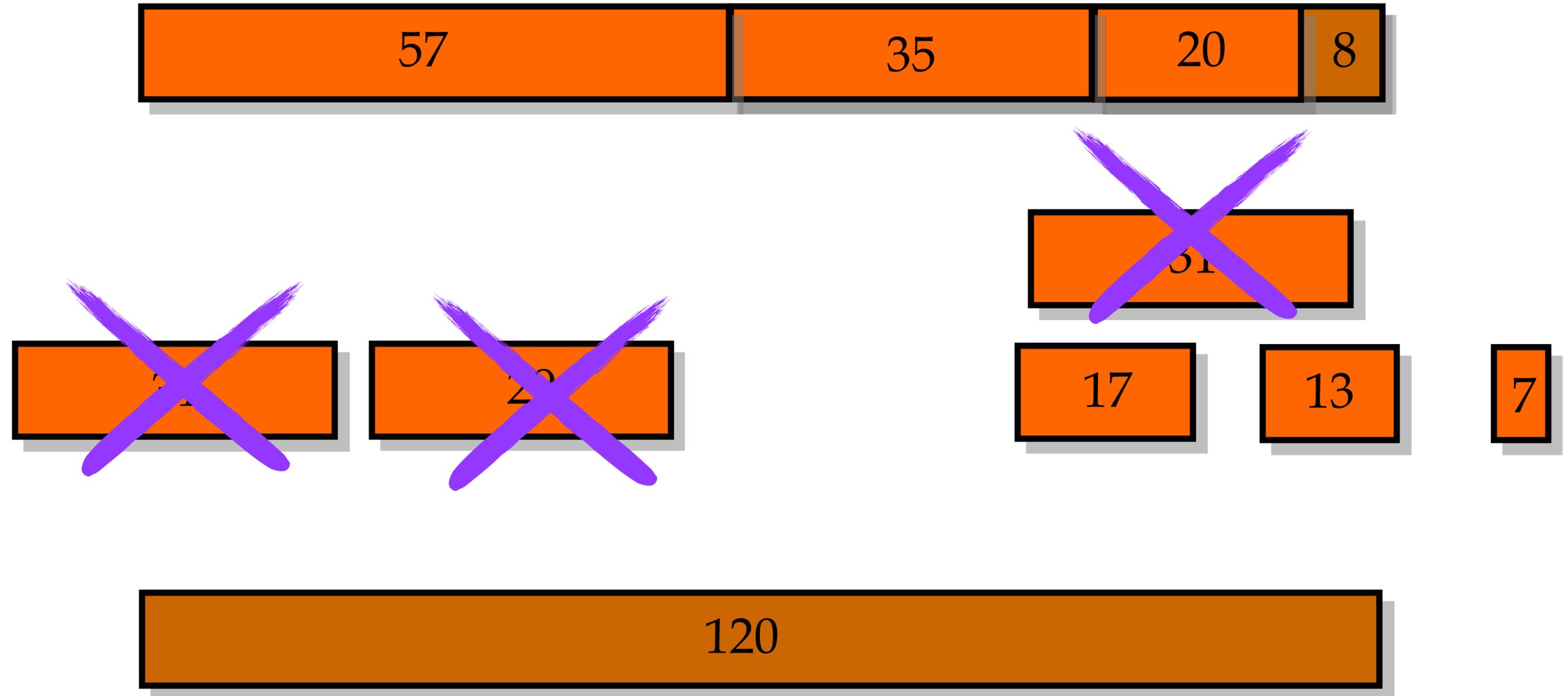
Keine Lösung?!



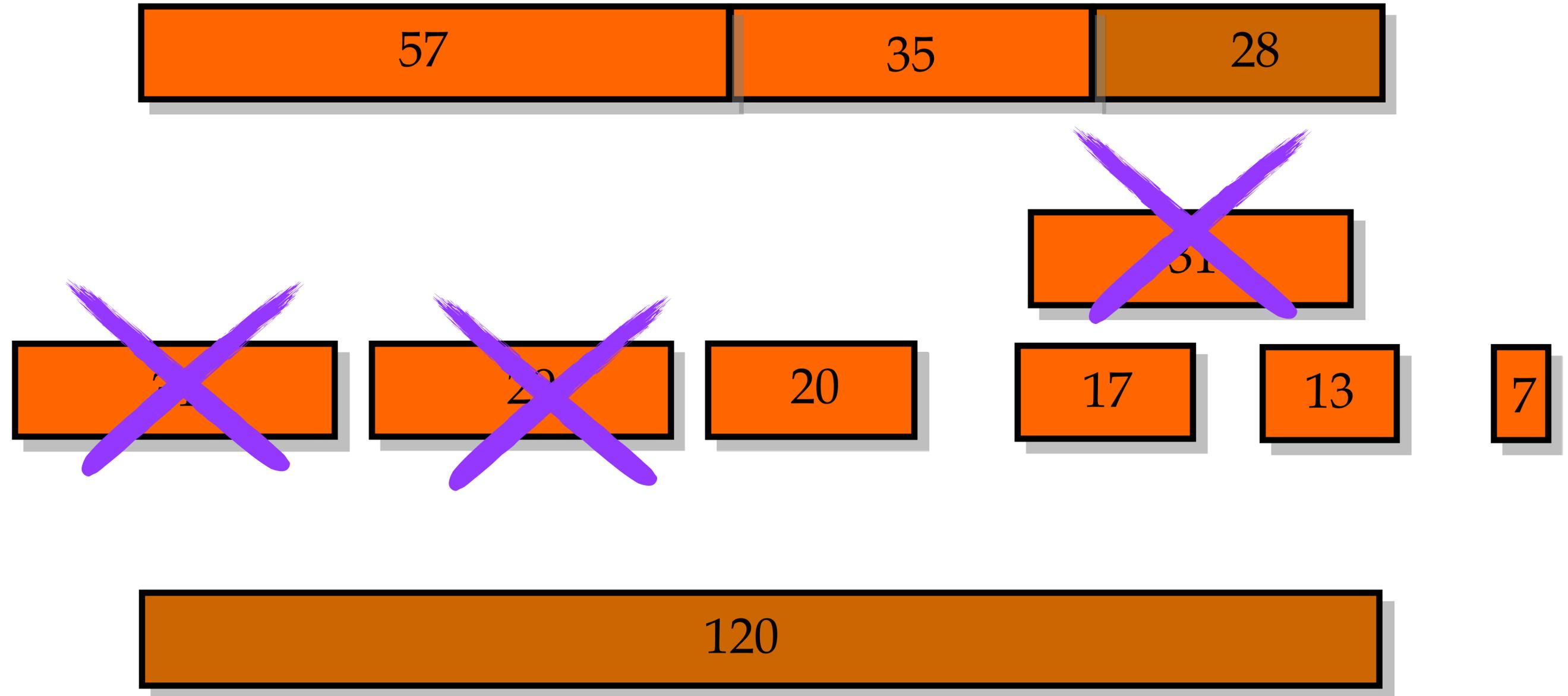
Keine Lösung?!



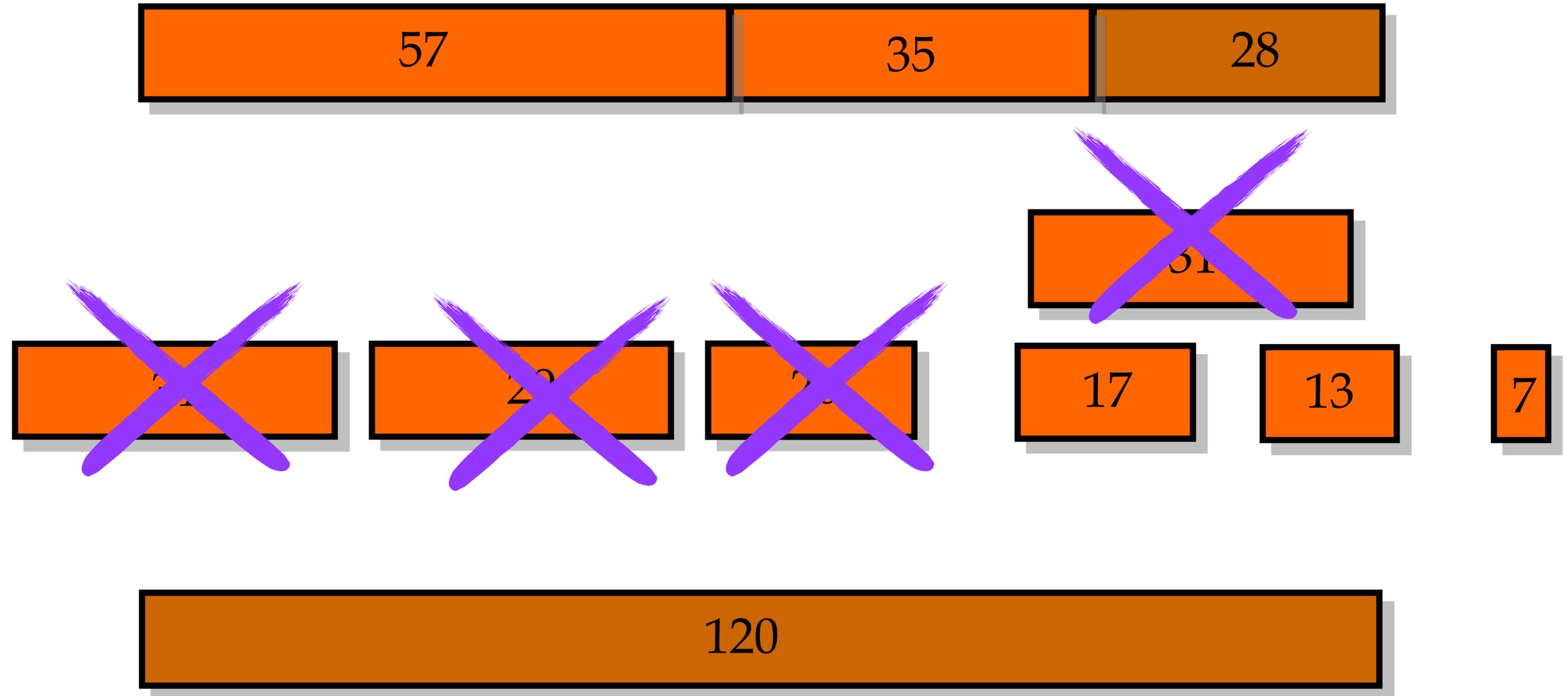
Keine Lösung?!



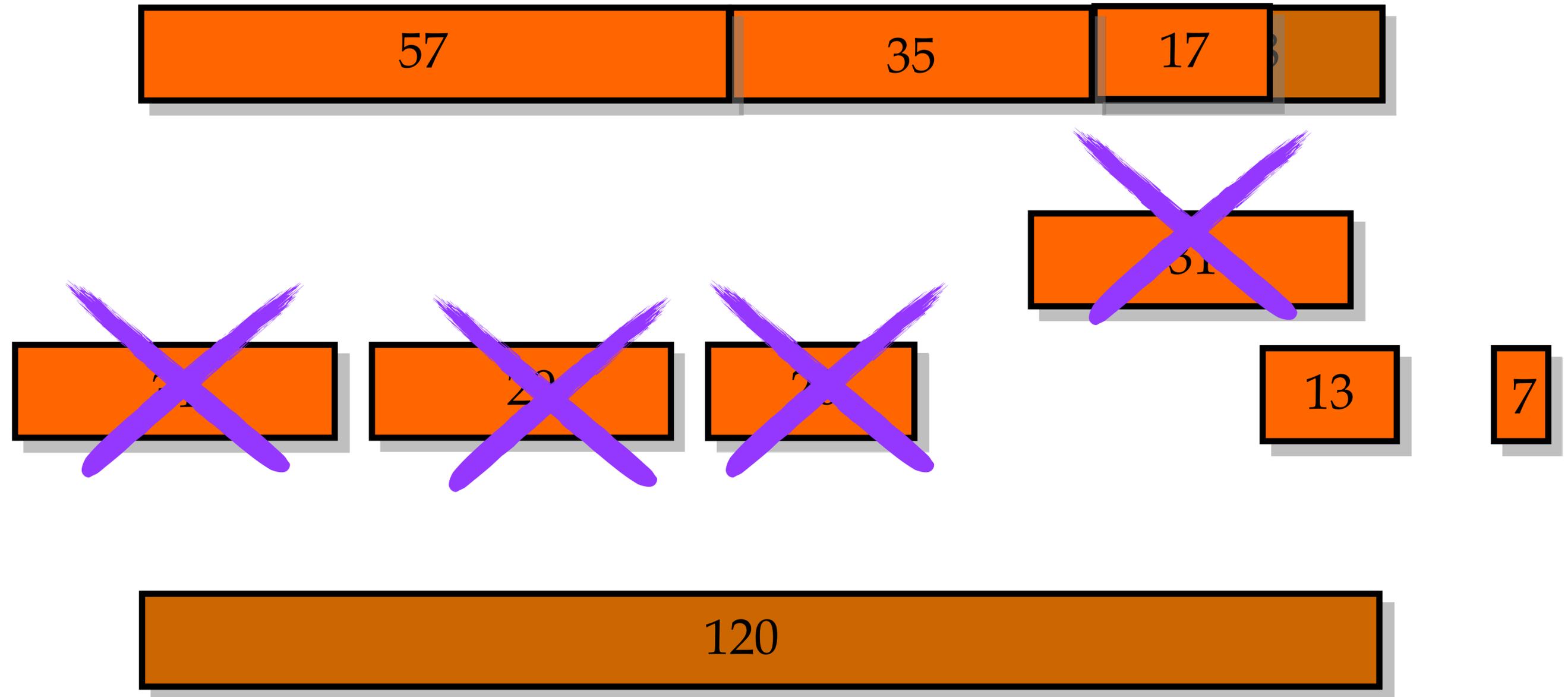
Keine Lösung?!



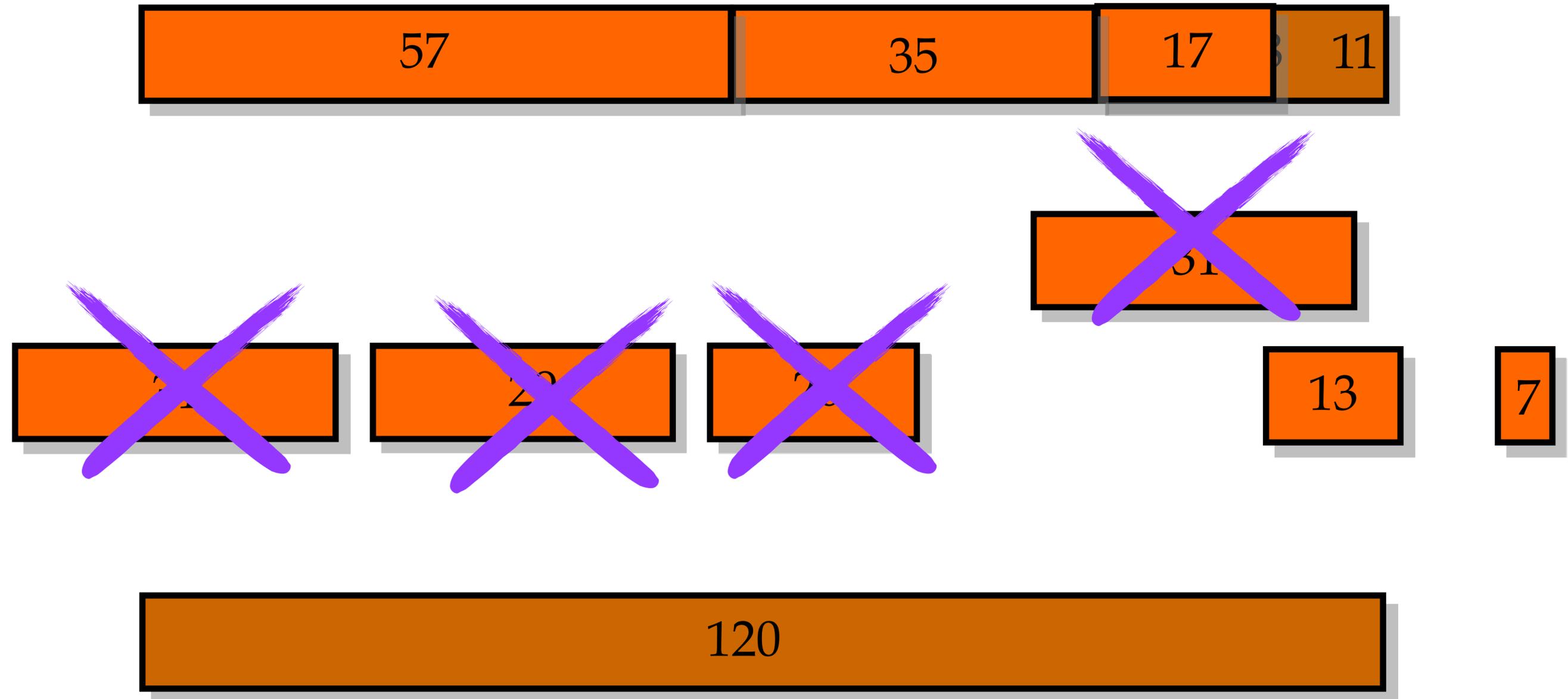
Keine Lösung?!



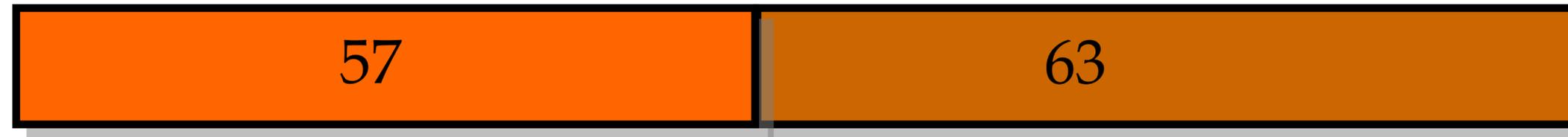
Keine Lösung?!



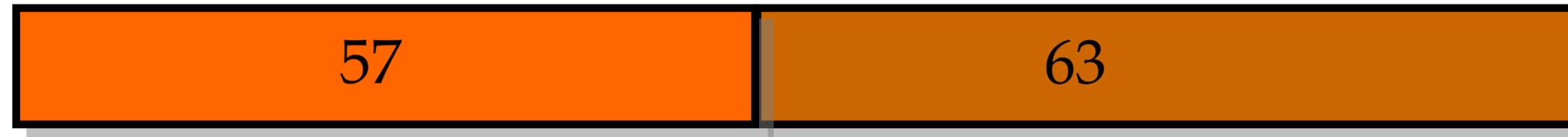
Keine Lösung?!



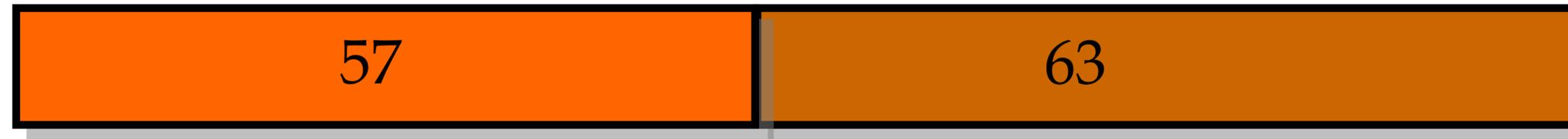
Keine Lösung?!



Keine Lösung?!



Keine Lösung?!



!



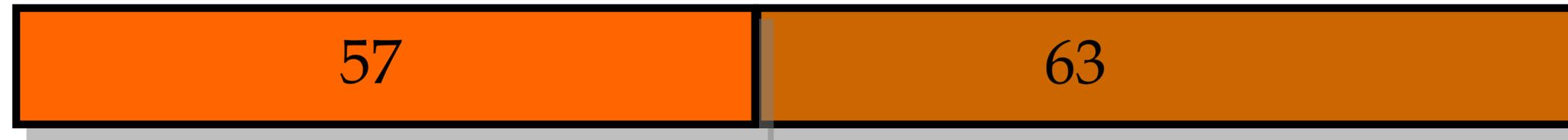
!

!

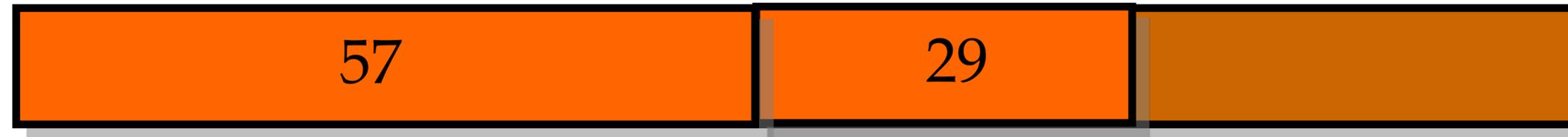
!



Keine Lösung?!



Keine Lösung?!



!

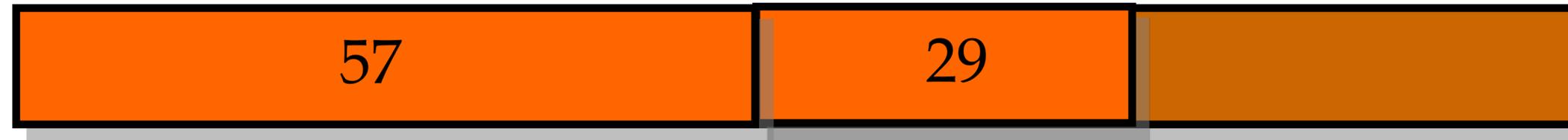


!

!

!

Keine Lösung?!



!

!



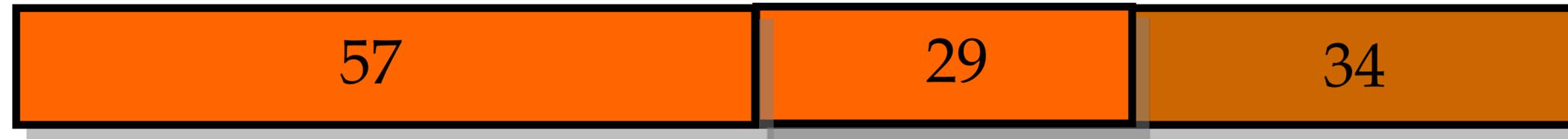
!

!

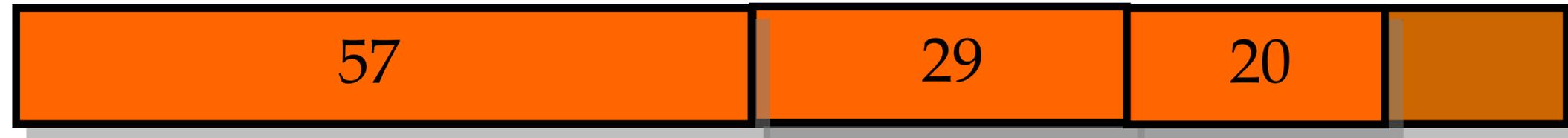
!



Keine Lösung?!



Keine Lösung?!



!

!

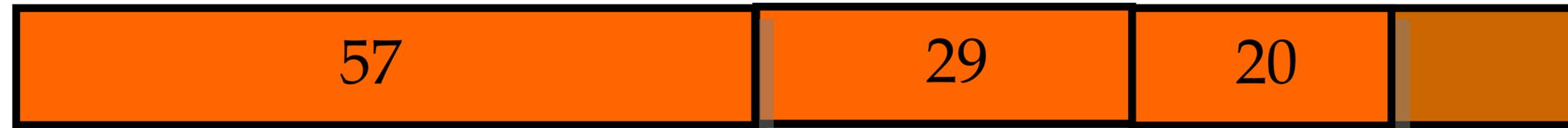


!

!

!

Keine Lösung?!



!

!

!



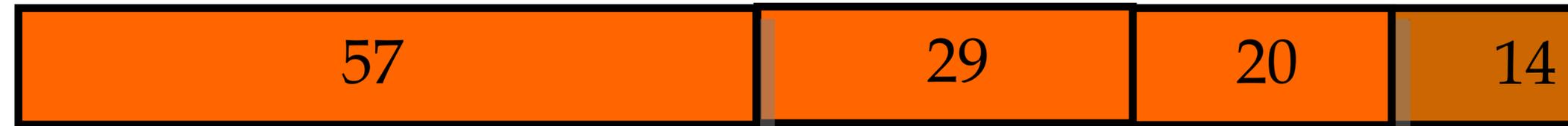
!

!

!



Keine Lösung?!



!

!

!



!

!

!



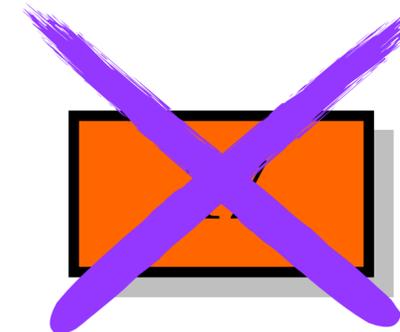
Keine Lösung?!



!

!

!



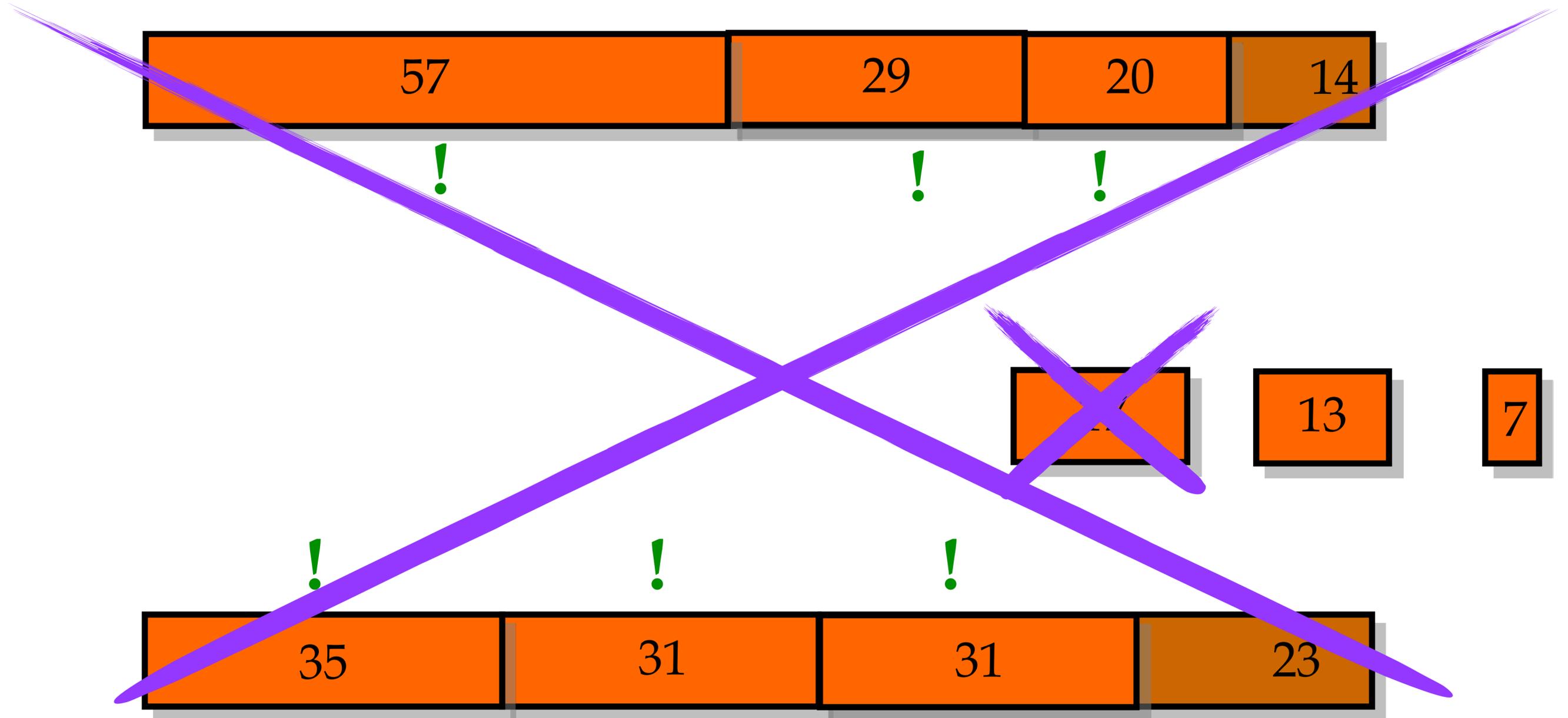
!

!

!

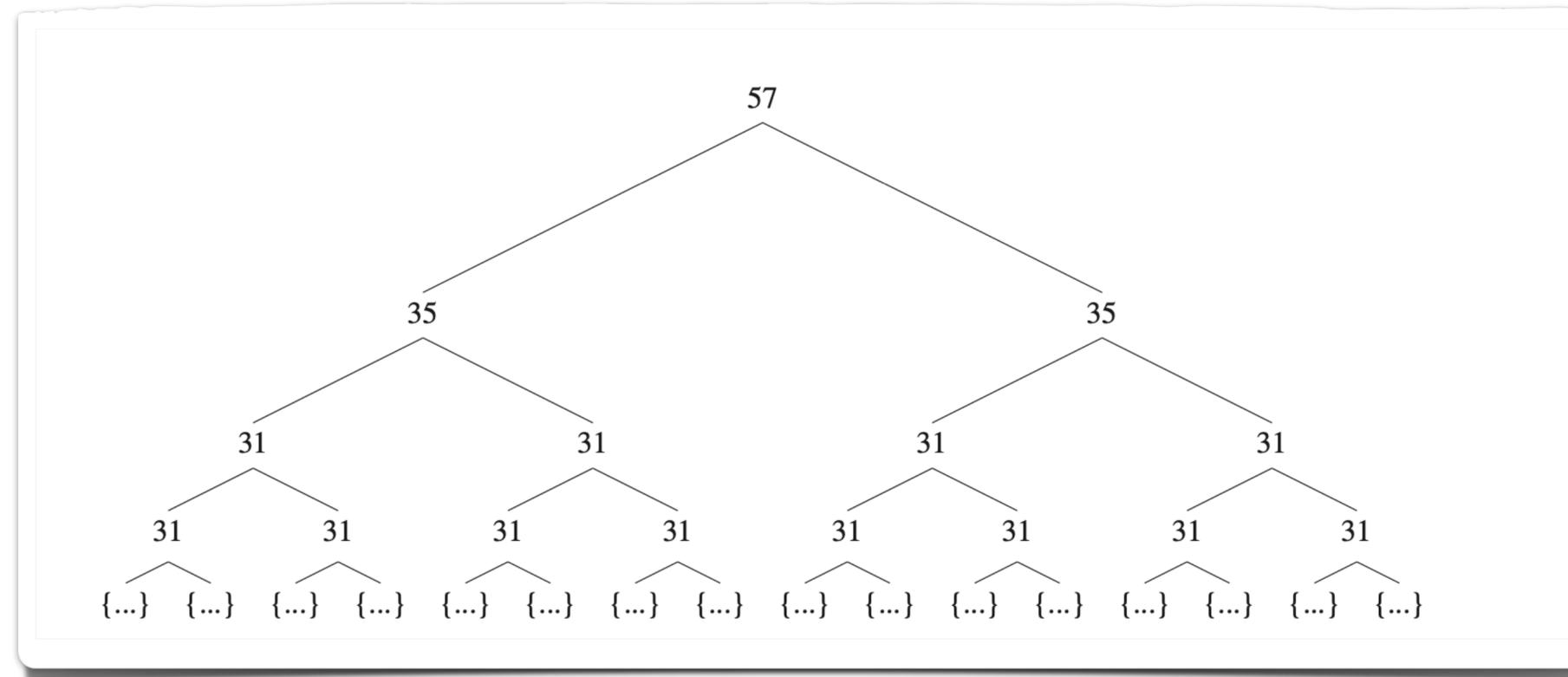


Keine Lösung?!

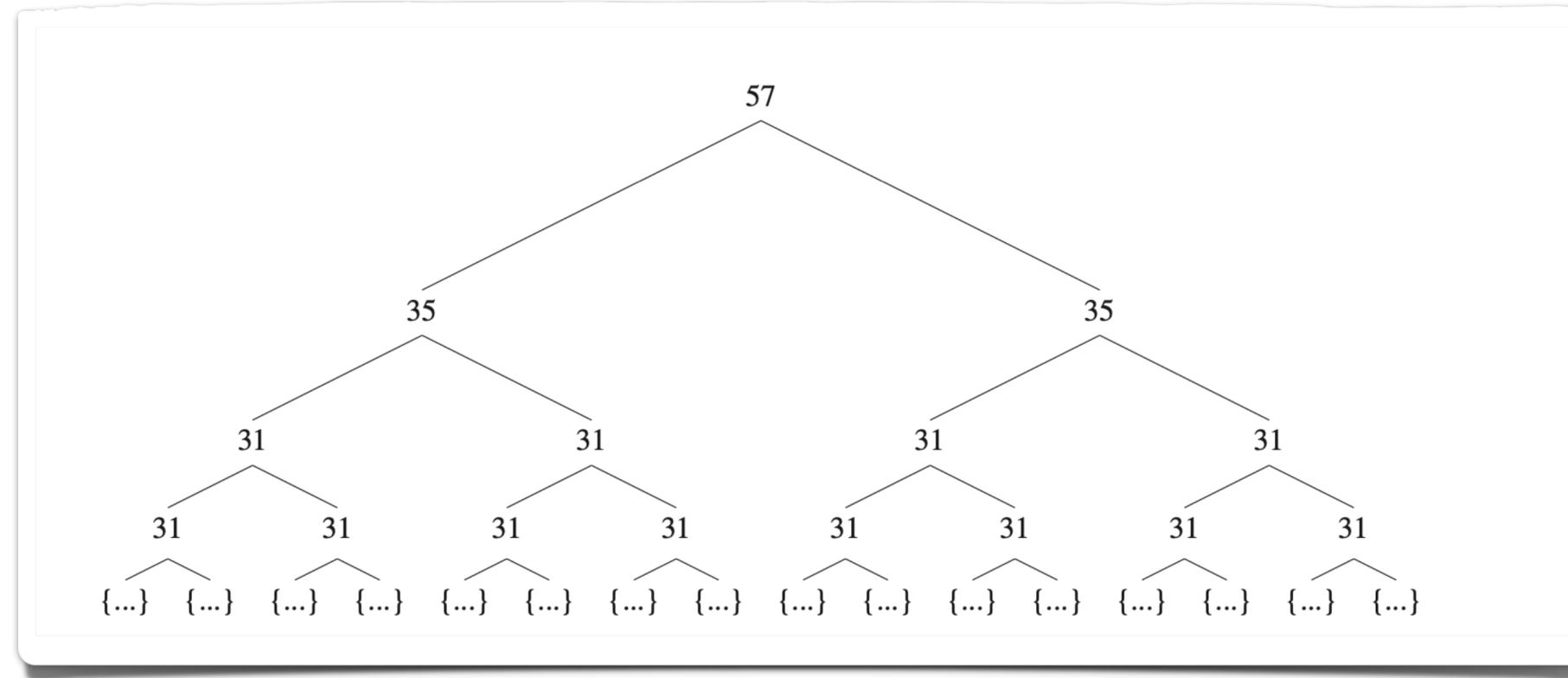


Ideen

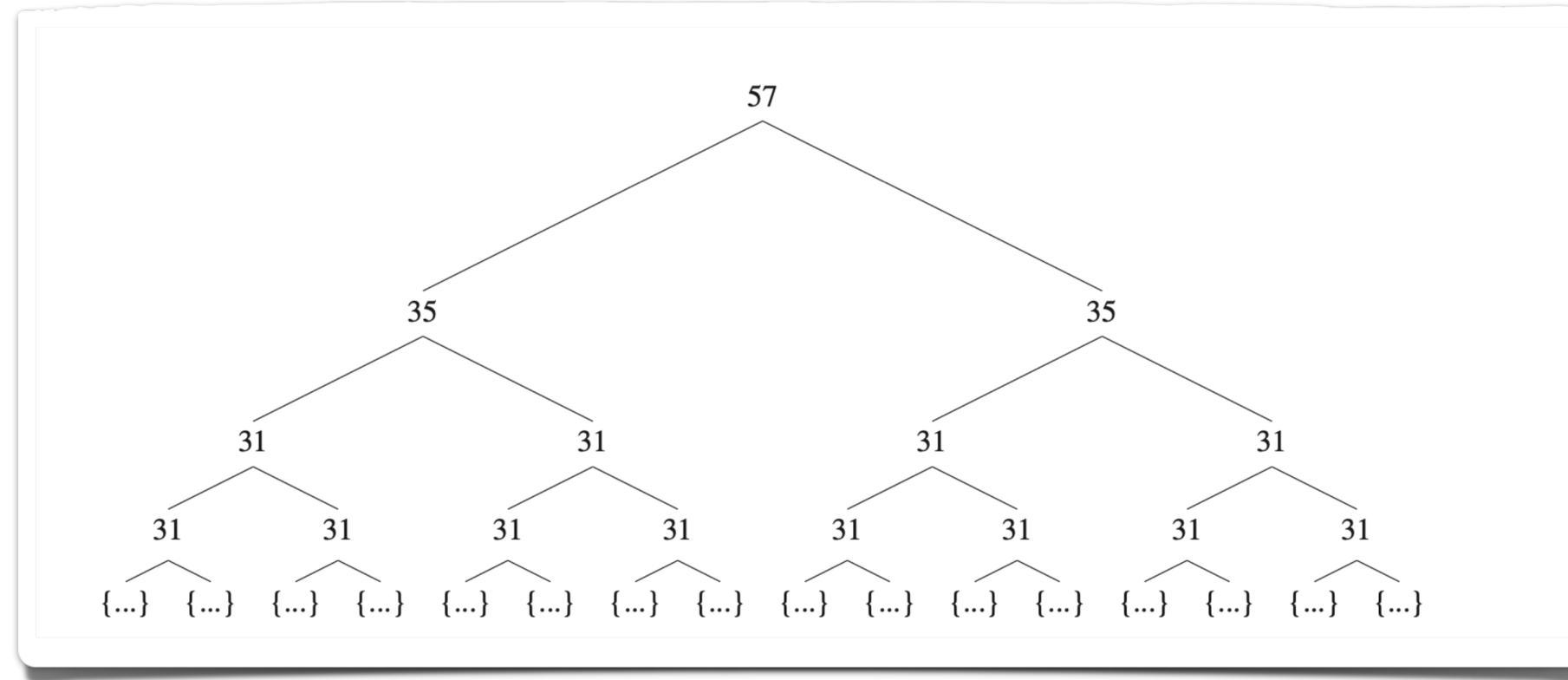
Ideen



Ideen

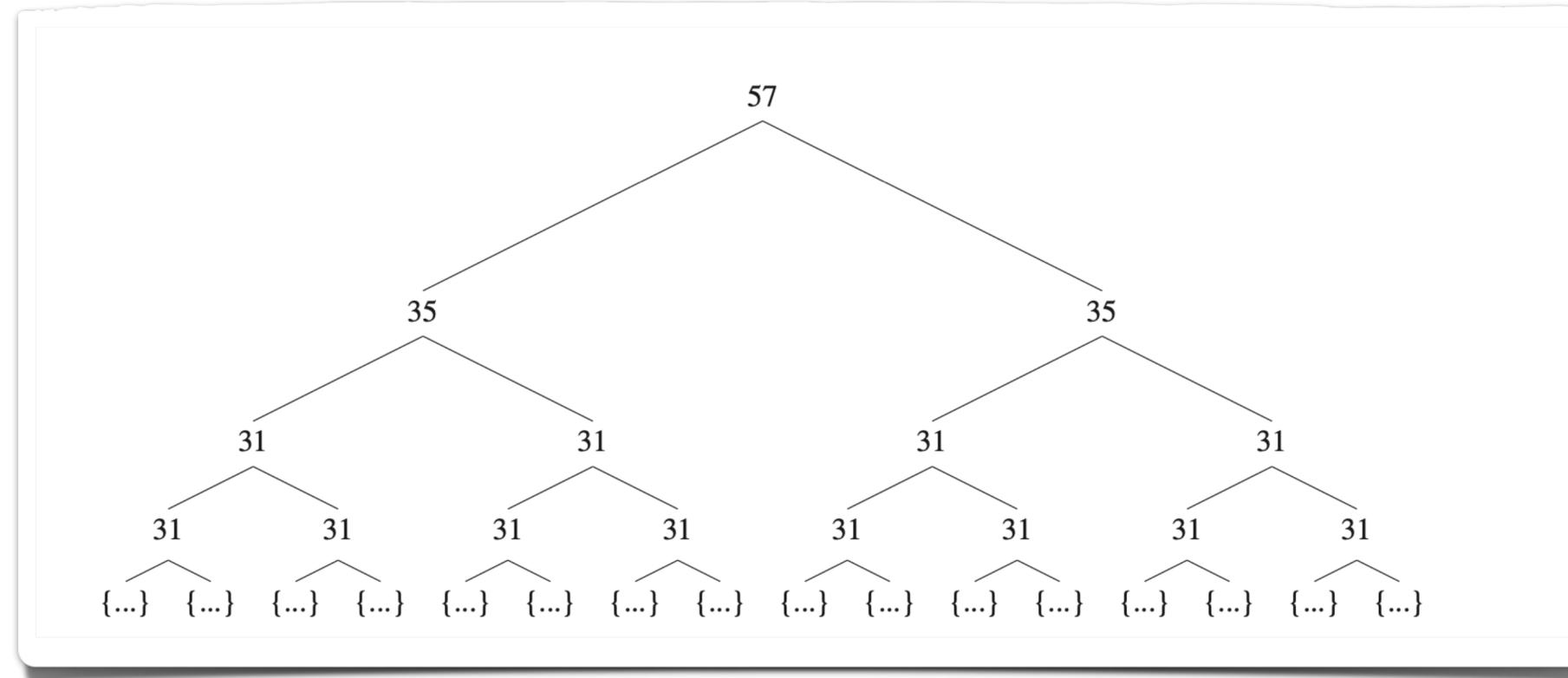


Ideen



- Zeige Unerreichbarkeit von Zahlen

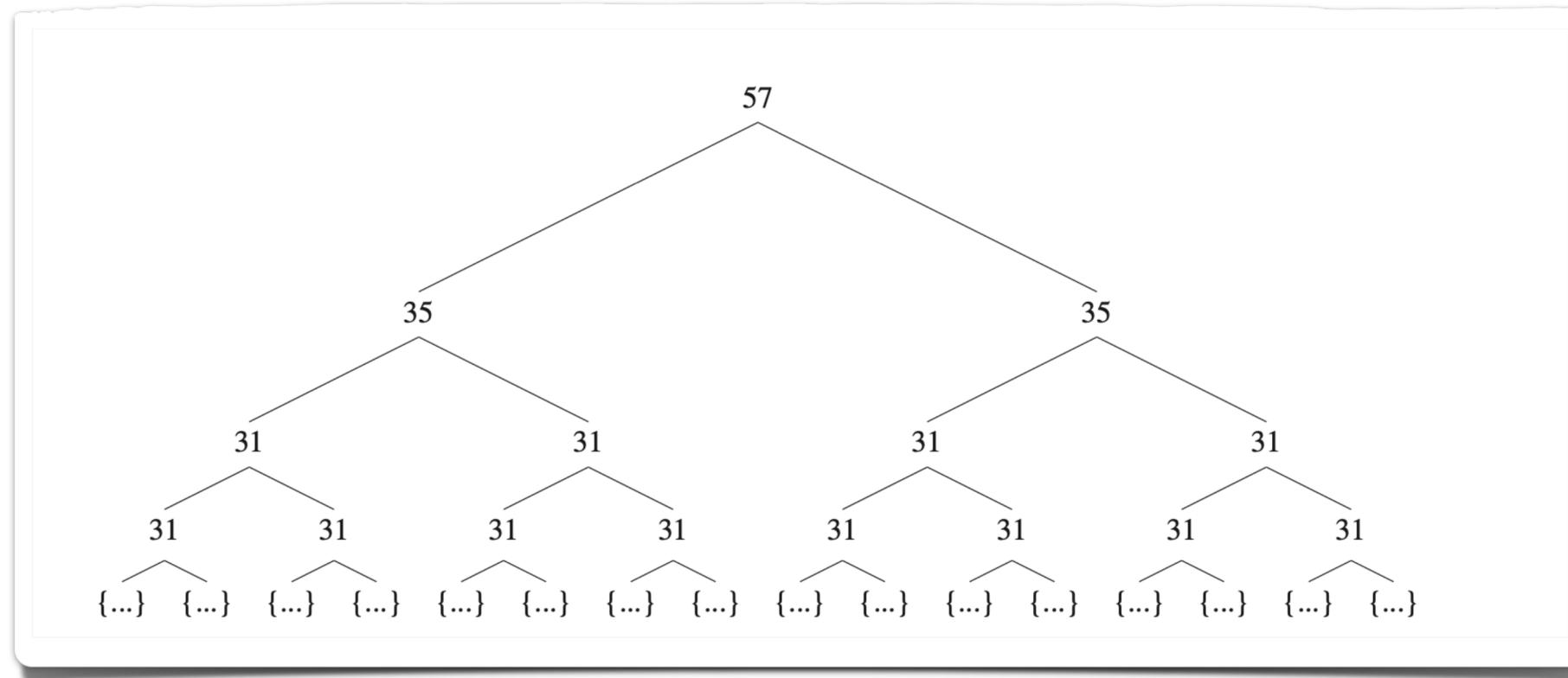
Ideen



- Zeige Unerreichbarkeit von Zahlen
- Rückwärts: Schließe Zahlen aus

Ideen

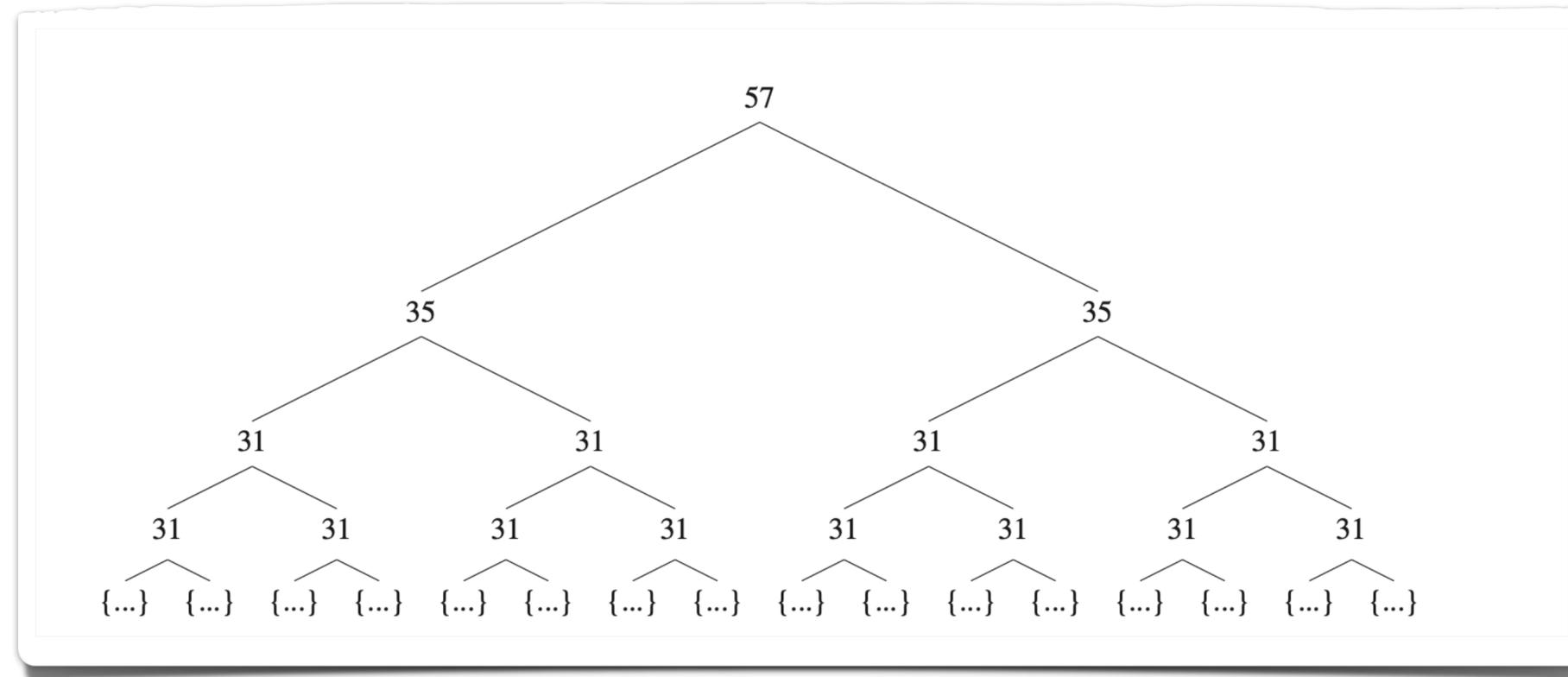
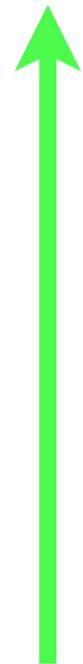
Dynamic
Programming



- Zeige Unerreichbarkeit von Zahlen
- Rückwärts: Schließe Zahlen aus

Ideen

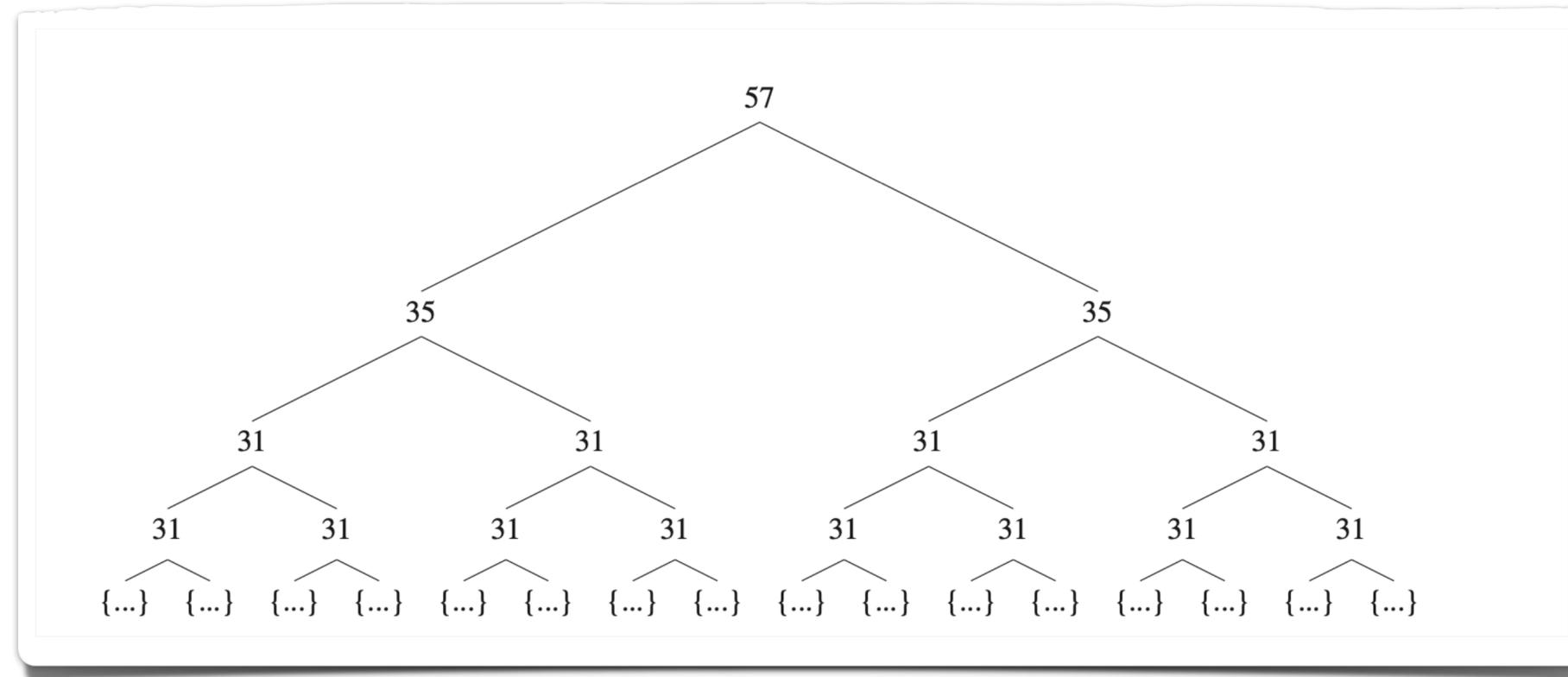
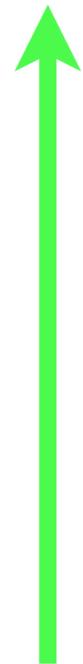
Dynamic
Programming



- Zeige Unerreichbarkeit von Zahlen
- Rückwärts: Schließe Zahlen aus

Ideen

Dynamic
Programming

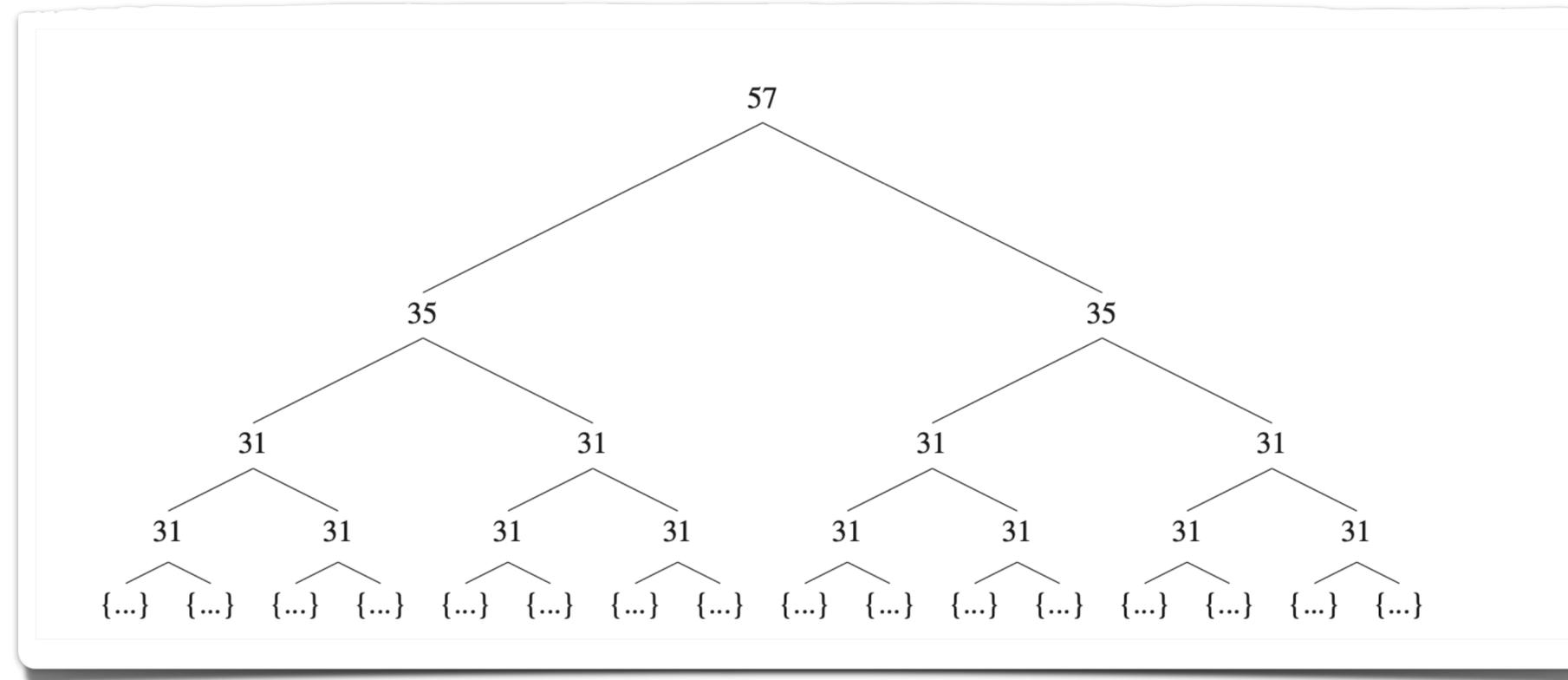
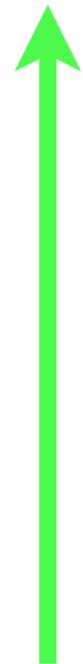


- Bestimme erreichbare Zahlen

- Zeige Unerreichbarkeit von Zahlen
- Rückwärts: SchlieÙe Zahlen aus

Ideen

Dynamic
Programming



- Bestimme erreichbare Zahlen
- Vorwärts: Nimm Zahlen hinzu

- Zeige Unerreichbarkeit von Zahlen
- Rückwärts: Schließe Zahlen aus

Beispiel

Beispiel

Beispiel 2.1

Beispiel

Beispiel 2.1

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

Beispiel

Beispiel 2.1

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

Bestimme alle Zahlen, die als Teilsummen erreichbar sind!

Beispiel

Beispiel 2.1

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

Bestimme alle Zahlen, die als Teilsummen erreichbar sind!

- Bestimme erreichbare Zahlen

Beispiel

Beispiel 2.1

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

Bestimme alle Zahlen, die als Teilsummen erreichbar sind!

- Bestimme erreichbare Zahlen
- Vorwärts: Nimm Zahlen hinzu

Beispiel

Beispiel 2.1

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

Bestimme alle Zahlen, die als Teilsummen erreichbar sind!

- Bestimme erreichbare Zahlen
- Vorwärts: Nimm Zahlen hinzu

Beispiel

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0																			
1																			
2																			
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0																				
1																				
2																				
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0																				
1																				
2																				
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1																		
1																			
2																			
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1																				
2																				
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1																				
2																				
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1																				
2																				
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1																				
2																				
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1																			
2																				
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0												
2																			
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0	
2																				
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0	
2																				
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1																		
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1																		
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1							1											
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1							1											
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	0	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	0	...	0

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1							1		1									
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0	
2	1							1		1										1
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	1	...	0
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0	
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0	
3																				

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3																			

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1																		

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1																		

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1							1											

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1							1		1									

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1							1	...	1									

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1							1		1							1		

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1							1	...	1							1	...	

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1							1		1							1		

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	0	1	0	0	1	...	0

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	1	...	0
3	1							1	...	1				1			1	...	

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	0	1	0	0	1	...	0

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	0	1	0	0	1	...	0

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	0	1	0	0	1	...	0

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	0	1	0	0	1	...	0

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1							1	...	1				1			1	...	

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1							1	...	1				1			1	...	

Beispiel

$$\{z_1, \dots, z_9\} = \{7, 13, 17, 20, 29, 31, 31, 35, 57\}$$

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	0	1	0	0	1	...	0

Allgemeiner

Allgemeiner

$$\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

Allgemeiner

$$\mathcal{P}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	1	...	0
3																			

Allgemeiner

$$\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$$\mathcal{S}(x, 0) = 0, \text{ für alle } x \in \{1, \dots, Z\}; \mathcal{S}(0, 0) = 1$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	1	...	0
3																			

Allgemeiner

$$\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$$\mathcal{S}(x, 0) = 0, \text{ für alle } x \in \{1, \dots, Z\}; \mathcal{S}(0, 0) = 1$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	1	...	0
3																			

Allgemeiner

$$\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$$\mathcal{S}(x, 0) = 0, \text{ für alle } x \in \{1, \dots, Z\}; \mathcal{S}(0, 0) = 1$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	1	...	0
3																			

$$\mathcal{S}(x, i-1) = 1 \Rightarrow \mathcal{S}(x, i) = 1$$

Allgemeiner

$$\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$$\mathcal{S}(x, 0) = 0, \text{ für alle } x \in \{1, \dots, Z\}; \mathcal{S}(0, 0) = 1$$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	1	...	0
3																			

$$\mathcal{S}(x, i-1) = 1 \Rightarrow \mathcal{S}(x, i) = 1$$

$$\mathcal{S}(x - z_i, i-1) = 1 \Rightarrow \mathcal{S}(x, i) = 1$$

Algorithmus

Algorithmus

Algorithmus 2.2. (Dynamic Programming für SUBSET SUM).

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

:

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) =$

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, \\ 0, \end{cases}$$

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \end{cases}$$

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	...	0
3	1	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

1: $\mathcal{S}(0, 0) := 1$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	...	0
3	1	0	0	0	0	0	0	0	...	1	0	0	0	0	0	0	0	...	0

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

1: $\mathcal{S}(0, 0) := 1$

2: **for** ($x=1$) **to** Z **do**

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	0	...	1
3																			

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

1: $\mathcal{S}(0, 0) := 1$

2: **for** ($x=1$) **to** Z **do**

3: $\mathcal{S}(x, 0) := 0;$

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	0	...	1
3																			

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

- 1: $\mathcal{S}(0, 0) := 1$
- 2: **for** ($x=1$) **to** Z **do**
- 3: $\mathcal{S}(x, 0) := 0;$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	...	1
3																			

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

- 1: $\mathcal{S}(0, 0) := 1$
- 2: **for** ($x=1$) **to** Z **do**
- 3: $\mathcal{S}(x, 0) := 0$;
- 4: **for** ($i = 1$) **to** n **do**

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	...	1
3																			

Note: The table above is a simplified representation of the DP table shown in the image. The image shows a table with columns 0, 1, 2, 3, 3, 5, 6, 7, ..., 13, 14, 15, 16, 17, 18, 19, 20, ..., 240. The row for i=0 is highlighted in purple. A blue arrow points to the first column (x=0). A green arrow points from the cell (i=1, x=1) to (i=2, x=13), with a red arrow pointing down from (i=1, x=1) to (i=2, x=1) and another red arrow pointing down from (i=1, x=7) to (i=2, x=7). The value +13 is written in green in the cell (i=2, x=13).

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

- 1: $\mathcal{S}(0, 0) := 1$
- 2: **for** ($x=1$) **to** Z **do**
- 3: $\mathcal{S}(x, 0) := 0;$

- 4: **for** ($i = 1$) **to** n **do**

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	...	1
3																			

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

$$\text{mit } \mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$$

1: $\mathcal{S}(0, 0) := 1$
 2: **for** ($x=1$) **to** Z **do**
 3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**
 5: **for** ($x = 0$) **to** $z_i - 1$ **do**

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	...	1
3																			

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

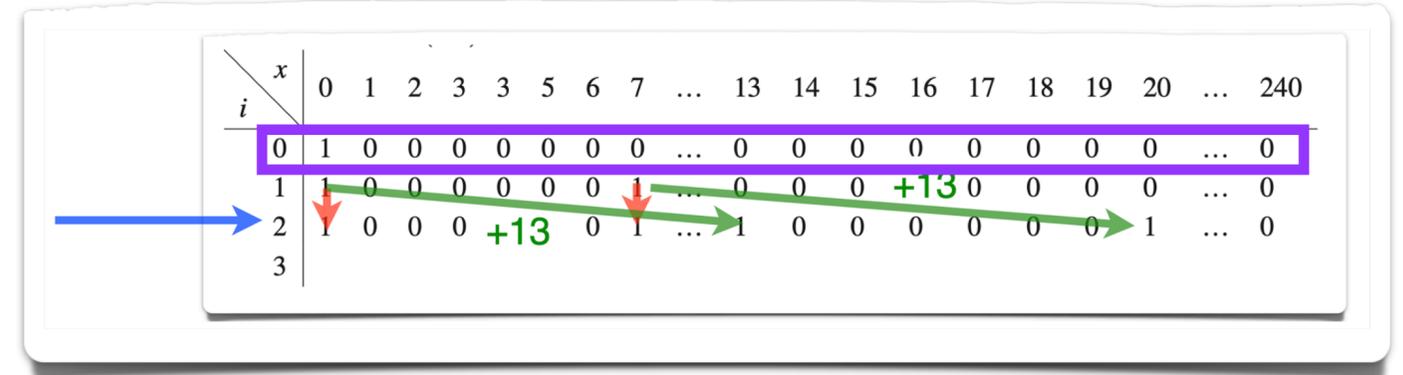
Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$
2: **for** ($x=1$) **to** Z **do**
3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**
5: **for** ($x = 0$) **to** $z_i - 1$ **do**
6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$



Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

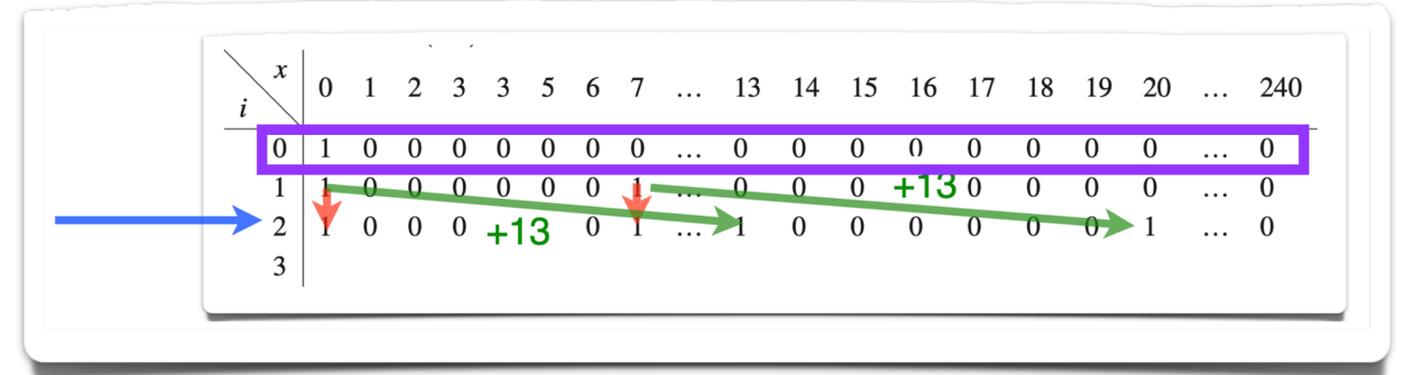
Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$
2: **for** ($x=1$) **to** Z **do**
3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**
5: **for** ($x = 0$) **to** $z_i - 1$ **do**
6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$
7: **for** ($x = z_i$) **to** Z **do**



Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$

2: **for** ($x=1$) **to** Z **do**

3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**

5: **for** ($x = 0$) **to** $z_i - 1$ **do**

6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$

7: **for** ($x = z_i$) **to** Z **do**

8: **if** ($(\mathcal{S}(x, i - 1) = 1)$ **OR** $(\mathcal{S}((x - z_i), i - 1) = 1)$) **then**

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	0	...	0	0	0	+13	0	0	0	0	...	0	
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	0	...	0	
3																				

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$

2: **for** ($x=1$) **to** Z **do**

3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**

5: **for** ($x = 0$) **to** $z_i - 1$ **do**

6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$

7: **for** ($x = z_i$) **to** Z **do**

8: **if** ($\mathcal{S}(x, i - 1) = 1$ **OR** ($\mathcal{S}((x - z_i), i - 1) = 1$)) **then**

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	0	...	0	0	0	+13	0	0	0	0	...	0	
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	0	...	0	
3																				

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$
 2: **for** ($x=1$) **to** Z **do**
 3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**
 5: **for** ($x = 0$) **to** $z_i - 1$ **do**
 6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$
 7: **for** ($x = z_i$) **to** Z **do**
 8: **if** ($\mathcal{S}(x, i - 1) = 1$ OR $\mathcal{S}((x - z_i), i - 1) = 1$) **then**

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	0	...	1
3																			

Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

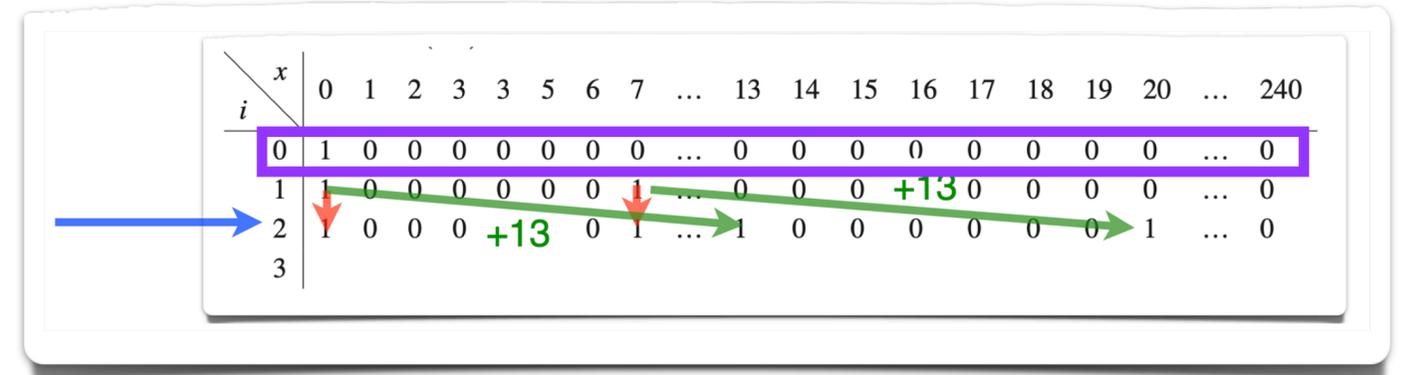
Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$
 2: **for** ($x=1$) **to** Z **do**
 3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**
 5: **for** ($x = 0$) **to** $z_i - 1$ **do**
 6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$
 7: **for** ($x = z_i$) **to** Z **do**
 8: **if** ($\mathcal{S}(x, i - 1) = 1$ **OR** ($\mathcal{S}((x - z_i), i - 1) = 1$)) **then**
 9: $\mathcal{S}(x, i) := 1;$



Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

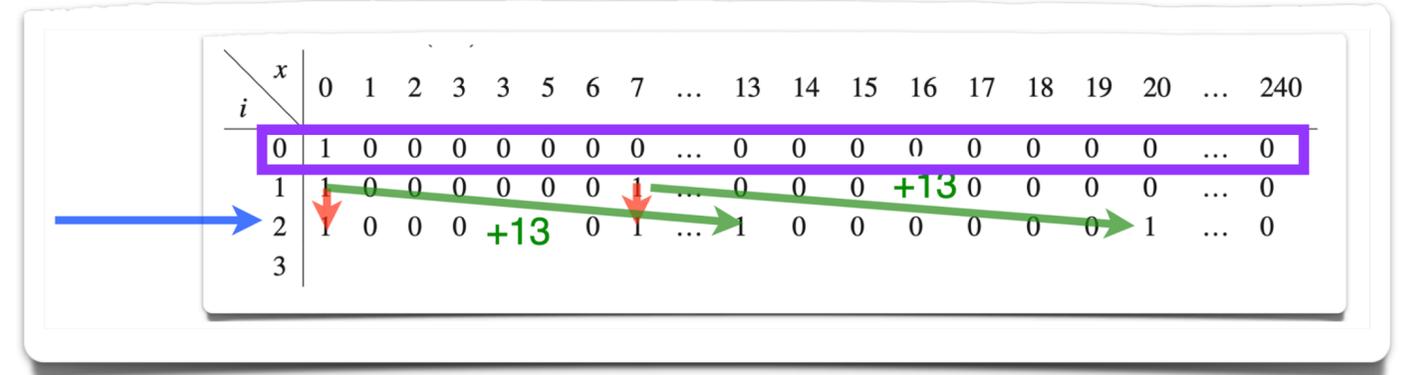
Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$
 2: **for** ($x=1$) **to** Z **do**
 3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**
 5: **for** ($x = 0$) **to** $z_i - 1$ **do**
 6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$
 7: **for** ($x = z_i$) **to** Z **do**
 8: **if** ($\mathcal{S}(x, i - 1) = 1$ **OR** ($\mathcal{S}((x - z_i), i - 1) = 1$)) **then**
 9: $\mathcal{S}(x, i) := 1;$
 10: **else**
 11: $\mathcal{S}(x, i) := 0;$



Algorithmus

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

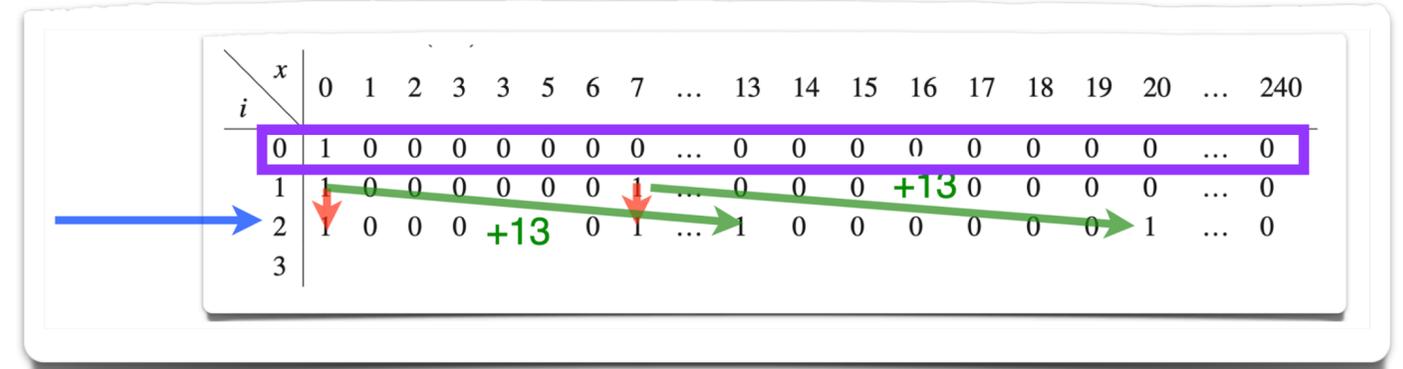
Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$
 2: **for** ($x=1$) **to** Z **do**
 3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**
 5: **for** ($x = 0$) **to** $z_i - 1$ **do**
 6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$
 7: **for** ($x = z_i$) **to** Z **do**
 8: **if** ($\mathcal{S}(x, i - 1) = 1$ **OR** ($\mathcal{S}((x - z_i), i - 1) = 1$)) **then**
 9: $\mathcal{S}(x, i) := 1;$
 10: **else**
 11: $\mathcal{S}(x, i) := 0;$

12: **return**



Korrektheit

Korrektheit

Satz 2.3 *Algorithmus 2.2* löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Korrektheit

Satz 2.3 *Algorithmus 2.2* löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis.

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis.

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0	
2																				
3																				

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0	
2																				
3																				

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

(I) z_i wird nicht verwendet

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0	
2																				
3																				

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

(I) z_i wird nicht verwendet

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0	
2																				
3																				

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

(I) z_i wird nicht verwendet

(II) z_i wird verwendet

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

(I) z_i wird nicht verwendet

(II) z_i wird verwendet

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0	
2																				
3																				

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

(I) z_i wird nicht verwendet $\Rightarrow x$ kann mit z_1, \dots, z_{i-1} erzeugt werden.

(II) z_i wird verwendet

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

(I) z_i wird nicht verwendet $\Rightarrow x$ kann mit z_1, \dots, z_{i-1} erzeugt werden.

(II) z_i wird verwendet

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0	
2																				
3																				

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

(I) z_i wird nicht verwendet $\Rightarrow x$ kann mit z_1, \dots, z_{i-1} erzeugt werden.

(II) z_i wird verwendet $\Rightarrow x - z_i$ kann mit z_1, \dots, z_{i-1} erzeugt werden.

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

(I) z_i wird nicht verwendet $\Rightarrow x$ kann mit z_1, \dots, z_{i-1} erzeugt werden.

(II) z_i wird verwendet $\Rightarrow x - z_i$ kann mit z_1, \dots, z_{i-1} erzeugt werden.

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2																			
3																			

Diagram illustrating the dynamic programming table for the Subset Sum problem. The table shows the state of the algorithm for different values of x (rows) and i (columns). The value $+13$ is highlighted in green, indicating the current state being processed. A blue arrow points to the row $x=1$, and a red arrow points to the column $i=7$. A green arrow points from the cell $(1, 7)$ to the cell $(1, 13)$, showing the transition from $x=1$ to $x=14$ by adding the element $z_7=13$.

Korrektheit

Satz 2.3 Algorithmus 2.2 löst das Problem SUBSET SUM. Die Laufzeit ist $O(Z \cdot n)$.

Beweis. Induktion über die Anzahl der Elemente n .

Induktionsanfang: Initialisierung korrekt für $n=0$.

Induktionssannahme: $\mathcal{S}(x, i-1)$ korrekt für alle i .

Induktionsschritt: Erzeugung von x mit z_1, \dots, z_i .

(I) z_i wird nicht verwendet $\Rightarrow x$ kann mit z_1, \dots, z_{i-1} erzeugt werden.

(II) z_i wird verwendet $\Rightarrow x - z_i$ kann mit z_1, \dots, z_{i-1} erzeugt werden.

$x \backslash i$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	0	...	1
3																			

Laufzeit

Laufzeit

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

```
1:  $\mathcal{S}(0, 0) := 1$ 
2: for ( $x=1$ ) to  $Z$  do
3:    $\mathcal{S}(x, 0) := 0$ ;
4: for ( $i = 1$ ) to  $n$  do
5:   for ( $x = 0$ ) to  $z_i - 1$  do
6:      $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1)$ ;
7:   for ( $x = z_i$ ) to  $Z$  do
8:     if ( $(\mathcal{S}(x, i - 1) = 1)$  OR  $(\mathcal{S}((x - z_i), i - 1) = 1)$ ) then
9:        $\mathcal{S}(x, i) := 1$ ;
10:    else
11:       $\mathcal{S}(x, i) := 0$ ;
12: return
```

Laufzeit

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

```
1:  $\mathcal{S}(0, 0) := 1$ 
2: for ( $x=1$ ) to  $Z$  do
3:    $\mathcal{S}(x, 0) := 0;$ 
4: for ( $i = 1$ ) to  $n$  do
5:   for ( $x = 0$ ) to  $z_i - 1$  do
6:      $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$ 
7:   for ( $x = z_i$ ) to  $Z$  do
8:     if ( $(\mathcal{S}(x, i - 1) = 1)$  OR  $(\mathcal{S}((x - z_i), i - 1) = 1)$ ) then
9:        $\mathcal{S}(x, i) := 1;$ 
10:    else
11:       $\mathcal{S}(x, i) := 0;$ 
12: return
```

Laufzeit

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$

2: **for** ($x=1$) **to** Z **do**

3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**

5: **for** ($x = 0$) **to** $z_i - 1$ **do**

6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$

7: **for** ($x = z_i$) **to** Z **do**

8: **if** ($(\mathcal{S}(x, i - 1) = 1)$ **OR** $(\mathcal{S}((x - z_i), i - 1) = 1)$) **then**

9: $\mathcal{S}(x, i) := 1;$

10: **else**

11: $\mathcal{S}(x, i) := 0;$

12: **return**

Laufzeit

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$

2: **for** ($x=1$) **to** Z **do**

3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**

5: **for** ($x = 0$) **to** $z_i - 1$ **do**

6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$

7: **for** ($x = z_i$) **to** Z **do**

8: **if** ($(\mathcal{S}(x, i - 1) = 1)$ **OR** $(\mathcal{S}((x - z_i), i - 1) = 1)$) **then**

9: $\mathcal{S}(x, i) := 1;$

10: **else**

11: $\mathcal{S}(x, i) := 0;$

12: **return**

Laufzeit

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$

2: **for** ($x=1$) **to** Z **do**

3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**

5: **for** ($x = 0$) **to** $z_i - 1$ **do**

6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$

7: **for** ($x = z_i$) **to** Z **do**

8: **if** ($(\mathcal{S}(x, i - 1) = 1)$ **OR** $(\mathcal{S}((x - z_i), i - 1) = 1)$) **then**

9: $\mathcal{S}(x, i) := 1;$

10: **else**

11: $\mathcal{S}(x, i) := 0;$

12: **return**

Laufzeit

Algorithmus 2.2 . (Dynamic Programming für SUBSET SUM).

Eingabe: Zahlen z_1, \dots, z_n mit $\sum_{i=1}^n z_i = Z$

Ausgabe: Boolesche Funktion $\mathcal{S} : \{0, \dots, Z\} \times \{0, \dots, n\} \rightarrow \{0, 1\}$

mit $\mathcal{S}(x, i) = \begin{cases} 1, & \text{falls } x \text{ mit } z_1, \dots, z_i \text{ erzeugt werden kann} \\ 0, & \text{sonst} \end{cases}$

1: $\mathcal{S}(0, 0) := 1$

2: **for** ($x=1$) **to** Z **do**

3: $\mathcal{S}(x, 0) := 0;$

4: **for** ($i = 1$) **to** n **do**

5: **for** ($x = 0$) **to** $z_i - 1$ **do**

6: $\mathcal{S}(x, i) := \mathcal{S}(x, i - 1);$

7: **for** ($x = z_i$) **to** Z **do**

8: **if** ($(\mathcal{S}(x, i - 1) = 1)$ **OR** $(\mathcal{S}((x - z_i), i - 1) = 1)$) **then**

9: $\mathcal{S}(x, i) := 1;$

10: **else**

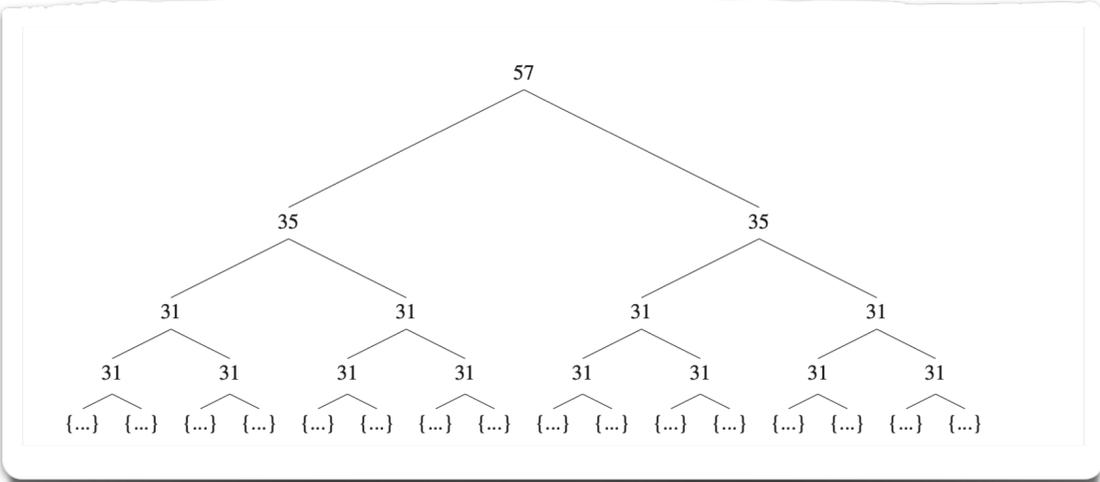
11: $\mathcal{S}(x, i) := 0;$

12: **return**

$O(nZ)$

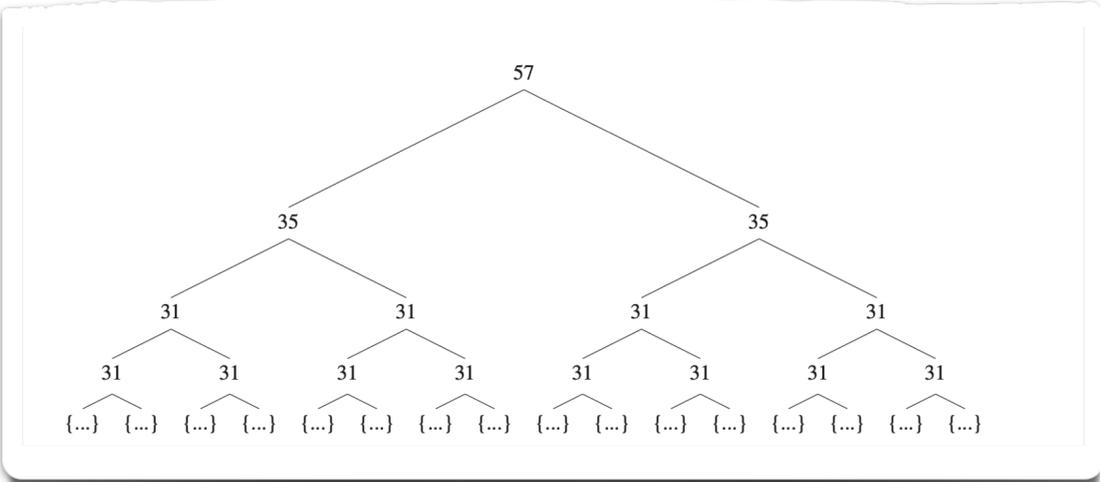
Effizienz?!

Effizienz?!



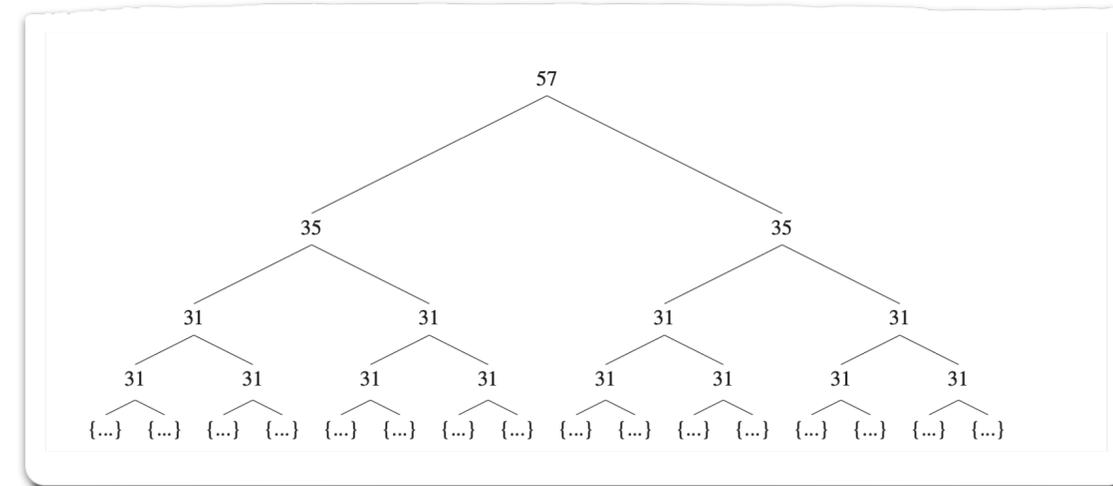
Effizienz?!

$$2^n$$



Effizienz?!

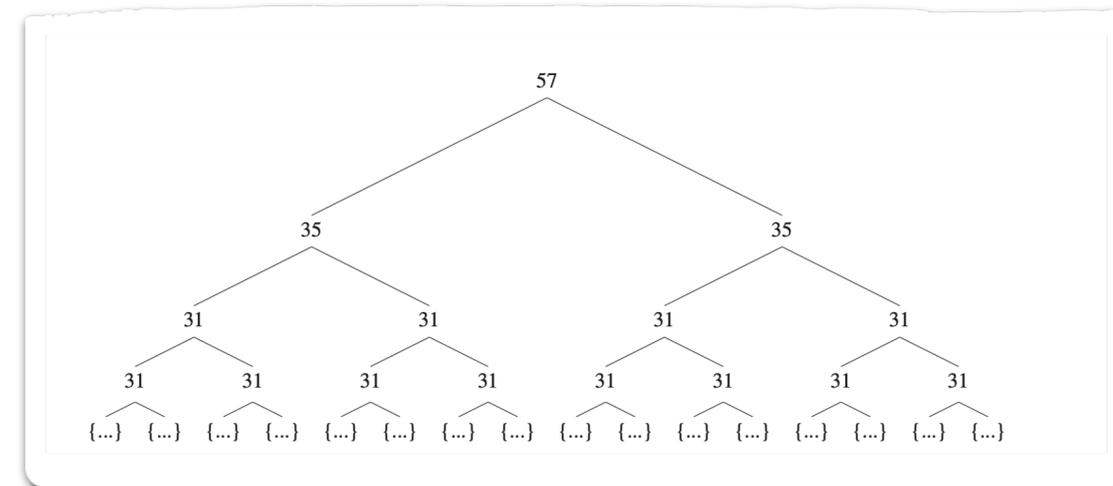
$$2^n$$



- Exponentiell viele Fälle!

Effizienz?!

$$2^n$$

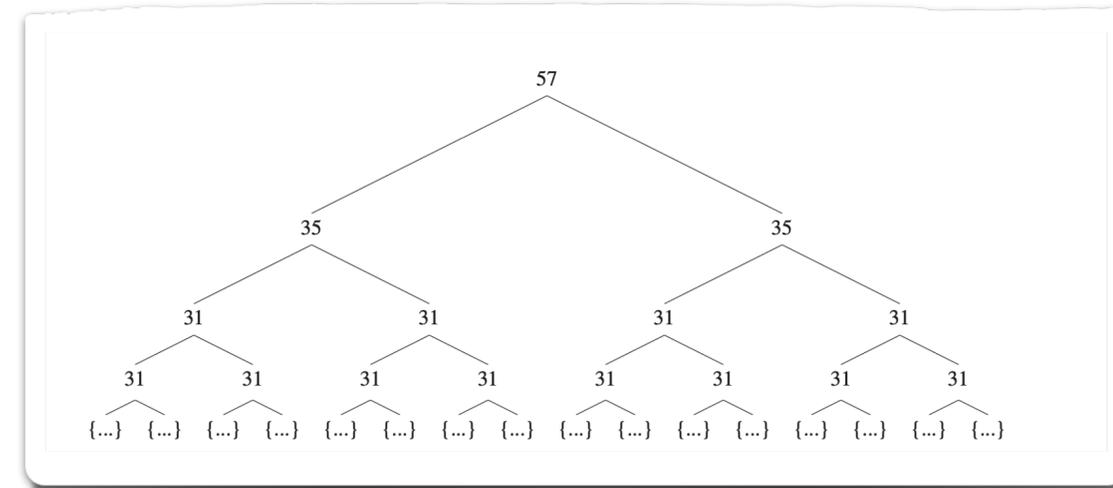


- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Effizienz?!

$$O(nZ)$$

$$2^n$$



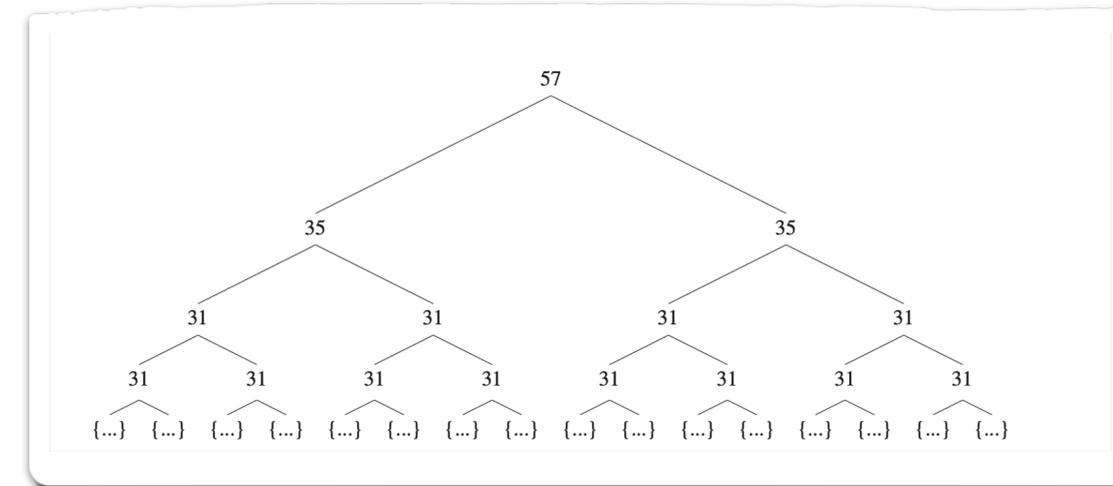
- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1																				
2																				
3																				

$$2^n$$



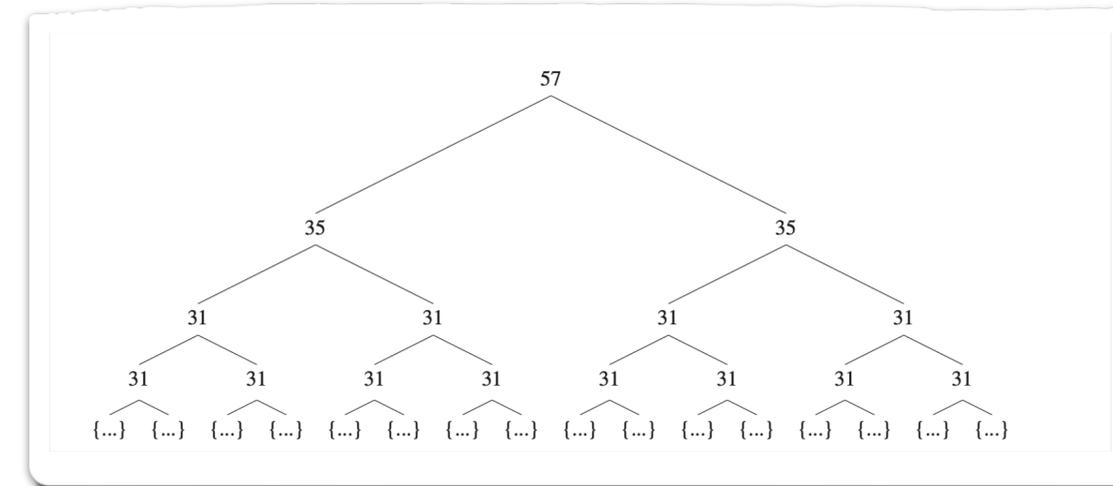
- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240	
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0	
1																				
2																				
3																				

$$2^n$$



1

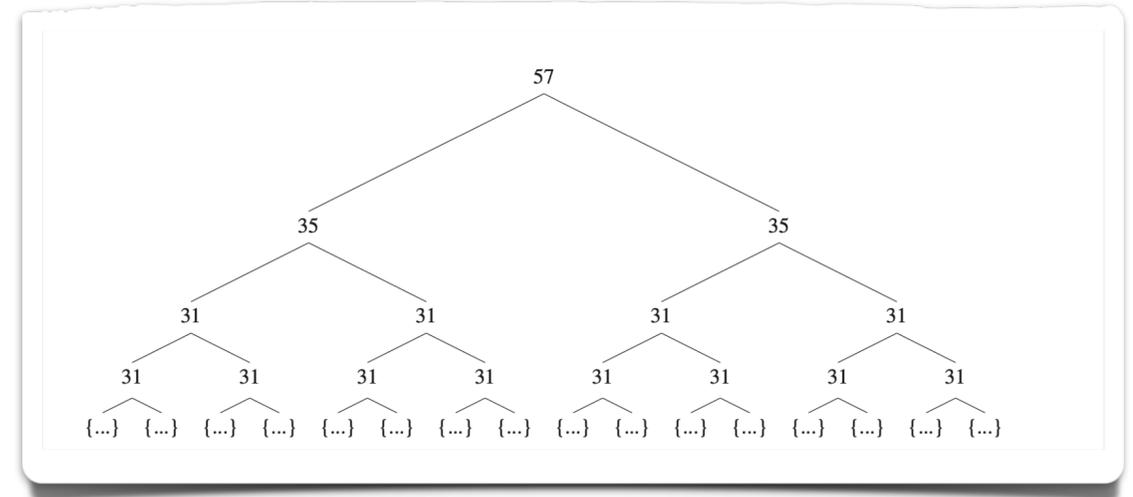
- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	+7	0	0	...	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	1										
2																			
3																			

$$2^n$$



1

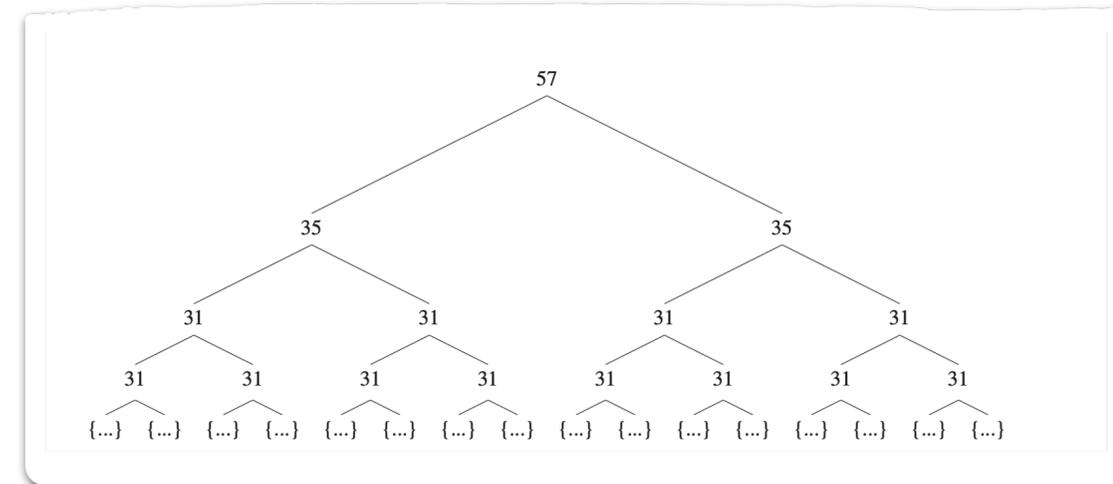
- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	+7	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
2																			
3																			

$$2^n$$



2

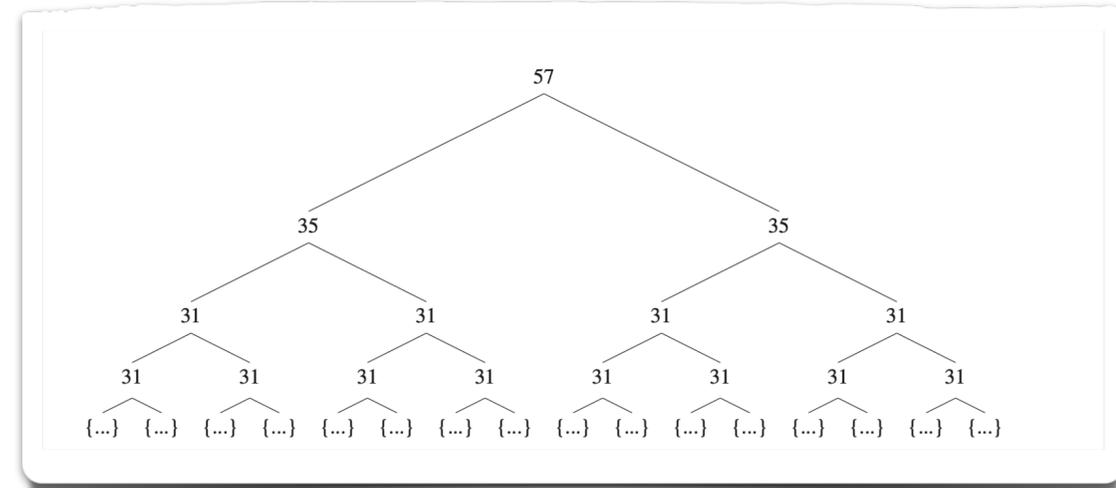
- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	1	...	0
3																			

$$2^n$$



2

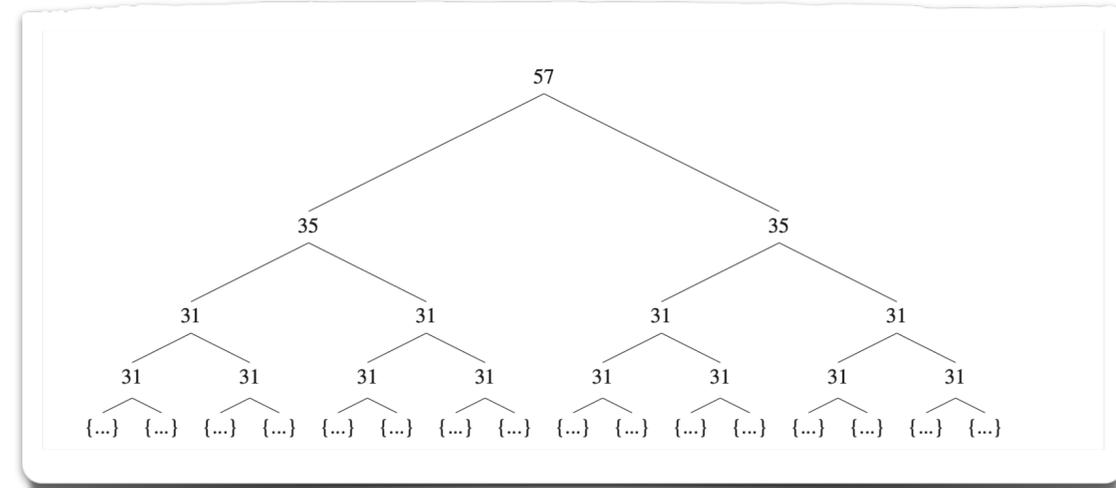
- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	+13	0	0	0	0	...	0
2	1	0	0	0	+13	0	1	...	1	0	0	0	0	0	0	0	1	...	0
3																			

$$2^n$$



4

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

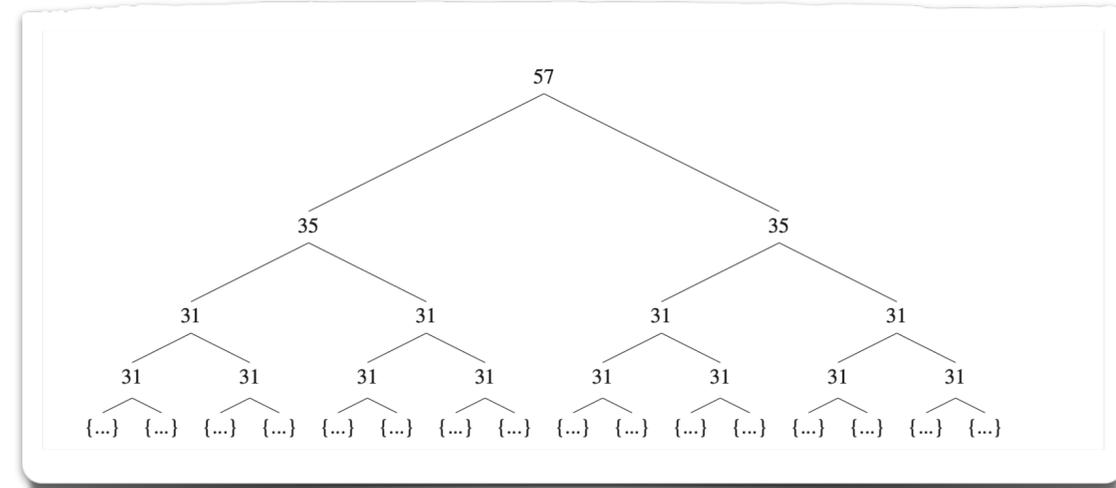
Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	1	0	0	0	1	...	0

Annotations: A blue arrow points to row 3. Red arrows point to the '1' in column 7 of rows 2 and 3, and to the '1' in column 20 of row 3. Green arrows point from the '1' in column 7 of row 2 to the '1' in column 17 of row 3, and from the '1' in column 20 of row 3 to the right. A green '+17' is written below the '1' in column 17 of row 3.

$$2^n$$



4

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

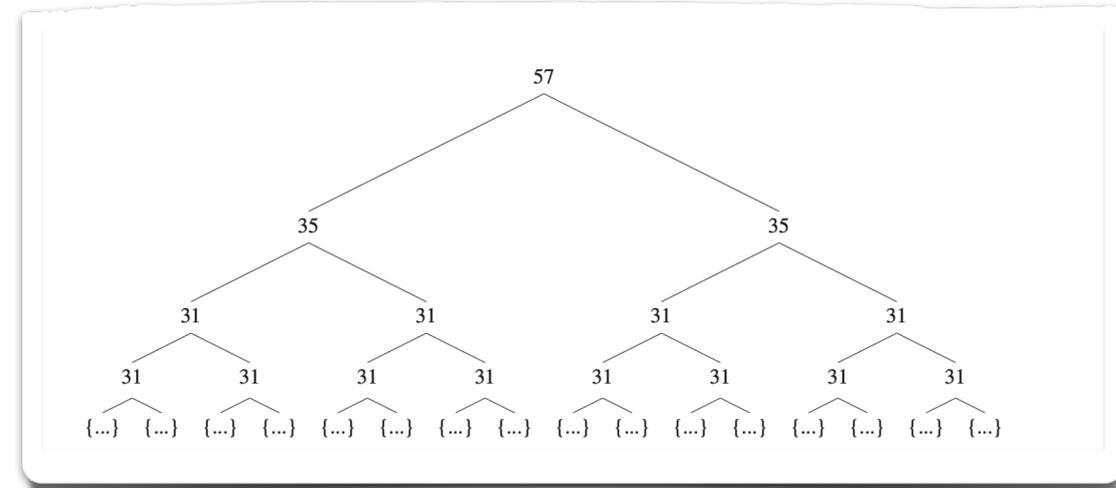
Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	1	0	0	0	1	...	0

Annotations: A blue arrow points to row 3. Red arrows point to the '1' in column 7 of rows 2 and 3, and to the '1' in column 20 of row 3. A green arrow points from the '1' in row 2, column 7 to the '1' in row 3, column 17. A green '+17' is written below the '1' in row 3, column 17. Above the table, the numbers 24, 30, and 37 are written above columns 20, 24, and 28 respectively.

$$2^n$$



8

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Effizienz?!

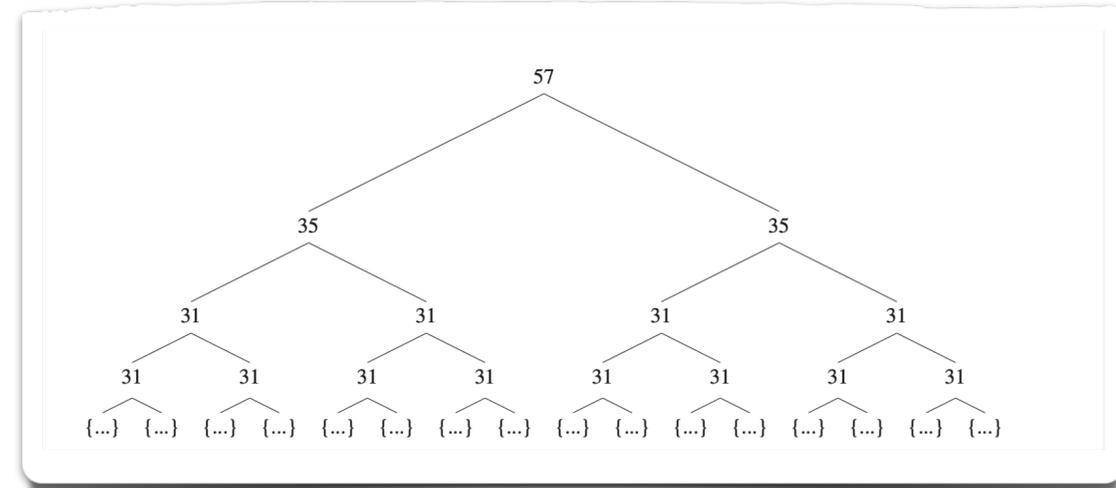
$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	1	0	0	0	1	...	0

→ (at row 3, column 0)
 ↓ (at row 2, column 0)
 ↓ (at row 2, column 7)
 ↓ (at row 2, column 10)
 ↓ (at row 2, column 17)
 → (at row 2, column 10)
 → (at row 2, column 17)
 → (at row 2, column 20)
 → (at row 2, column 240)

+17

$$2^n$$



16

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Effizienz?!

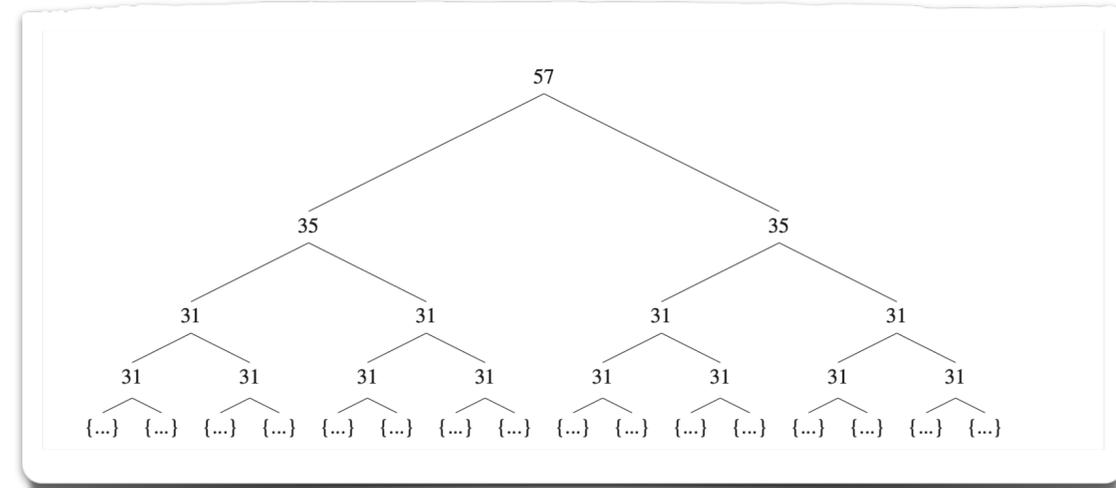
$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	1	0	0	0	1	...	0

→ (at i=3, x=0)
 ↓ (at i=2, x=0)
 ↓ (at i=2, x=7)
 ↓ (at i=2, x=13)
 ↓ (at i=2, x=20)
 → (at i=2, x=17)
 → (at i=2, x=240)

+17

$$2^n$$



32

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

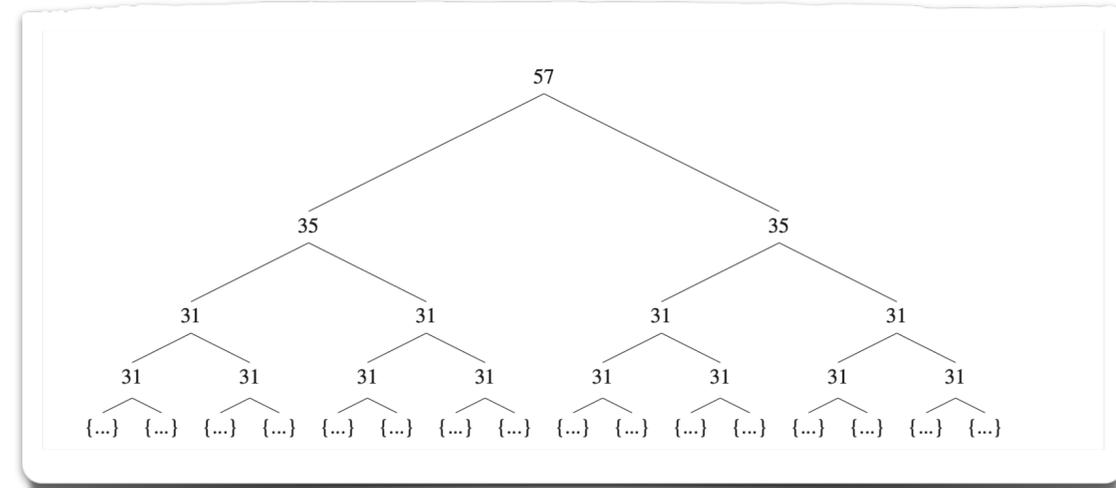
Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	1	0	0	0	1	...	0

Annotations: A blue arrow points to row 3. Red arrows point to the '1' in column 7 of rows 2 and 3, and to the '1' in column 20 of row 3. Green arrows point from the '1' in column 7 of row 2 to the '1' in column 17 of row 3, and from the '1' in column 20 of row 3 to the right. A green '+17' is written below the '1' in column 17 of row 3.

$$2^n$$



64

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

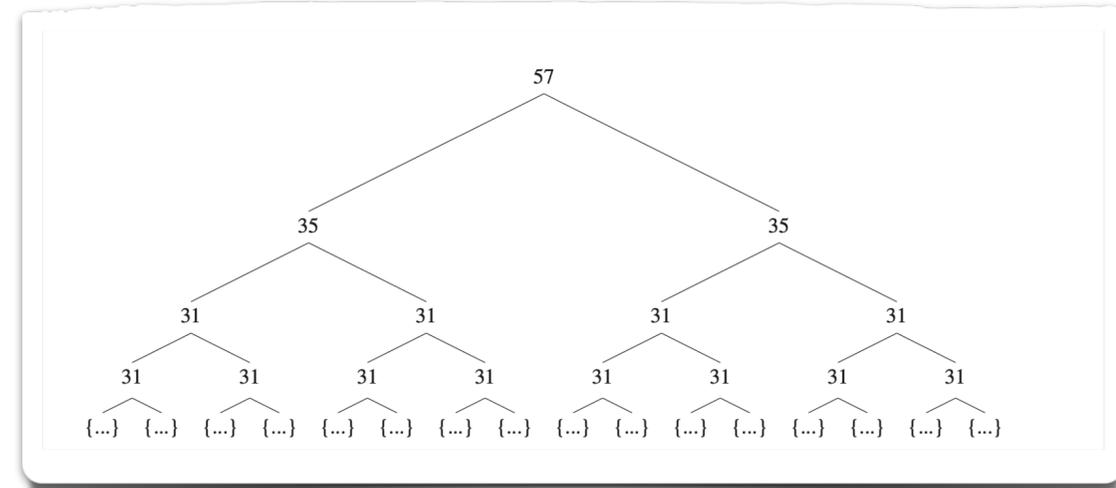
Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	1	0	0	0	1	...	0

Annotations: A blue arrow points to row 3. A green arrow points from row 2 to row 3. Red arrows point to the '1' in row 2, column 8 and row 3, column 10. A green '+17' is written below row 3, column 12.

$$2^n$$



128

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

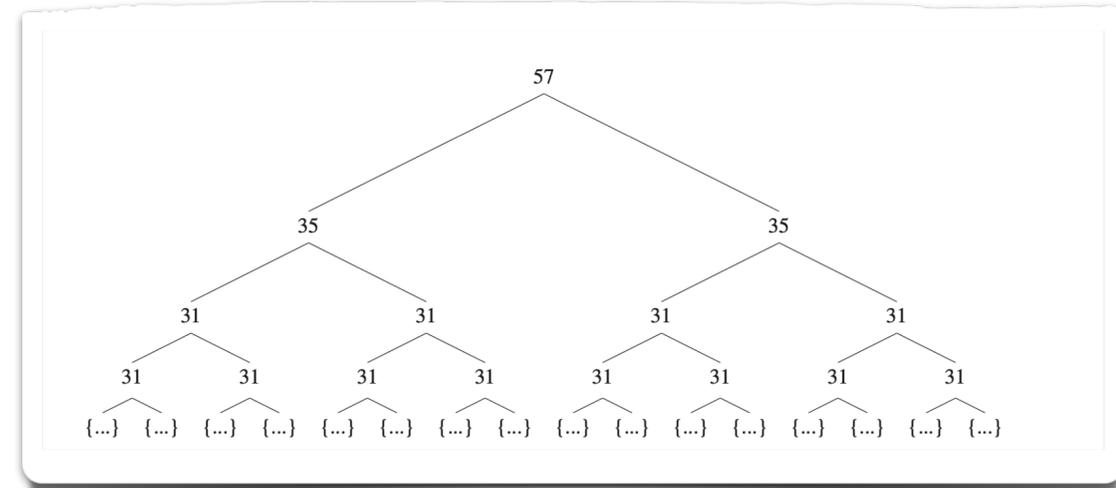
Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0
3	1	0	0	0	0	0	0	1	...	1	0	0	1	0	0	0	1	...	0

→ (at i=3, x=0)
 ↓ (at i=2, x=0)
 ↓ (at i=2, x=7)
 ↓ (at i=2, x=13)
 ↓ (at i=2, x=20)
 → (at i=2, x=20)
 → (at i=3, x=17)
 → (at i=3, x=240)

$$2^n$$



256?!

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

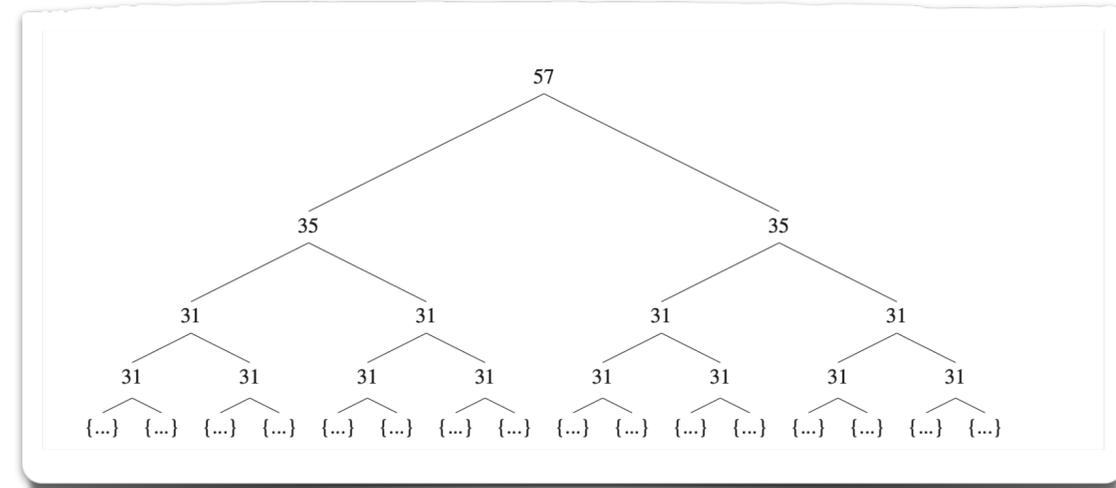
Effizienz?!

$$O(nZ)$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	24	30	37	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0			
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0			
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0			
3	1	0	0	0	0	0	0	1	...	1	0	0	1	0	0	0	1	...	0			

Note: A blue arrow points to row 3. Red arrows point to the first '1' in each row. A green arrow points from the first '1' in row 2 to the '1' in row 3 at column 13. A green '+17' is written below the '1' in row 3 at column 13.

$$2^n$$



256?!

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Z klein im Vergleich zu 2^n : viele Duplikate gespart

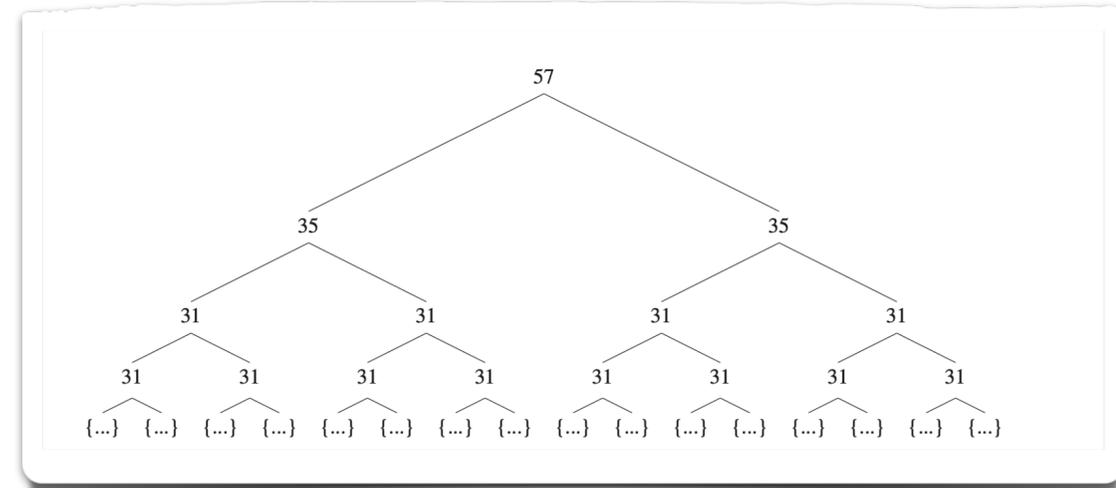
Effizienz?!

$$O(nZ)$$

$$2^n$$

$i \backslash x$	0	1	2	3	3	5	6	7	...	13	14	15	16	17	18	19	20	...	24	30	37	240
0	1	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	...	0			
1	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0	0	...	0			
2	1	0	0	0	0	0	0	1	...	1	0	0	0	0	0	0	1	...	0			
3	1	0	0	0	0	0	0	1	...	1	0	0	1	0	0	0	1	...	0			

Note: A blue arrow points to row 3. Red arrows point to the first '1' in each row. A green arrow points from the first '1' in row 2 to the '1' in row 3 at column 17, with '+17' written below it.



256?!

- Exponentiell viele Fälle!
- Wo sparen wir Arbeit?

Z klein im Vergleich zu 2^n : viele Duplikate gespart
 Z groß im Vergleich zu 2^n : viel unnötige Arbeit

Fragen

Fragen

- Wie lässt sich das Prinzip verallgemeinern?

Fragen

- Wie lässt sich das Prinzip verallgemeinern?
- Wie geht das für Knapsack?

Fragen

- Wie lässt sich das Prinzip verallgemeinern?
- Wie geht das für Knapsack?
- Wie geht das für andere Probleme?

Vielen Dank!

s.fekete@tu-bs.de