# Übungsblatt 01

## Algorithmen I - Sommersemester 2024

**Abgabe im ILIAS bis 26.04.2024, 18:00 Uhr**

Die Abgabe erfolgt alleine oder zu zweit als eine PDF-Datei über das Übungsmodul in der Gruppe eures Tutoriums im ILIAS. Beschriftet die Abgabe deutlich mit Matrikelnummer und Name.

- Achtet bei handschriftlichen Abgaben auf Lesbarkeit.
- Achtet darauf, ob Algorithmen in Worten oder Pseudocode beschrieben werden sollen.
- Es werden immer asymptotisch möglichst effiziente Algorithmen erwartet, wenn nicht anders angegeben.
- Wenn Korrektheits- oder Laufzeitanalysen gefordert sind, behandelt diese separat von der Algorithmenbeschreibung.

**Gesamtpunkte**: 20

### Aufgabe 1 - O(rdnungs-)-Kalkül (3 Punkte)

Ordne die folgenden Funktionen so an, dass $f \in \mathcal{O}(g)$ gilt, genau dann, wenn $f$ links von $g$ eingeordnet ist. Begründungen sind nicht notwendig.

(i) $2^{n \cdot \log_2(n)}$

(ii) $n!$

(iii) $\log_2(\sqrt{n})$

(iv) $\sqrt{\log_3(n)}$

(v) $\frac{n^2}{\log_2(n)}$

(vi) $\pi^{\ln(n)}$

Hinweis: Achte auf die Basen der Logarithmen.

## Lösung 1

Die richtige Reihenfolge ist: (iv), (iii), (vi), (v), (ii), (i)
Also:

$$\sqrt{\log_3(n)}, \quad \log_2(\sqrt{n}), \quad \pi^{\ln(n)}, \quad \frac{n^2}{\log_2(n)}, \quad n!, \quad 2^{n\cdot\log_2(n)}$$

**Begründung (war nicht gefordert).** Wir vereinfachen zuerst unter Anwendung von Potenz- und Logarithmusregeln:

(i)

$$\begin{aligned} 2^{n\cdot\log_2(n)} &= \left(2^{\log_2(n)}\right)^n \\ &= n^n \end{aligned}$$

(ii)

$$n!$$

(iii)

$$\log_2(\sqrt{n}) = \frac{1}{2}\cdot\log_2(n) \in \Theta(\log_2(n))$$

(iv)

$$\sqrt{\log_3(n)} = \sqrt{\frac{\log_2(n)}{\log_2(3)}} \in \Theta(\sqrt{\log_2(n)})$$

(v) Hier ist nichts zu vereinfachen, wir können uns allerdings zwei grobe Abschätzungen überlegen, um ein Gefühl für die Funktion zu bekommen.

- Es gilt $\frac{n^2}{\log_2(n)} \in \mathcal{O}(n^2)$, da wir die schrumpfende Funktion $\frac{1}{\log_2(n)}$ weglassen.
- Aus $\sqrt{n} \in \Omega(\log_2(n))$ folgt, dass $\frac{n^2}{\log_2(n)} \in \Omega(\frac{n^2}{\sqrt{n}}) = \Omega(n\cdot\sqrt{n}) = \Omega\left(n^{\frac{3}{2}}\right)$

(vi)

$$\begin{aligned} \pi^{\ln(n)} &= \pi^{\frac{\log_\pi(n)}{\log_\pi(e)}} \\ &= \pi^{\log_\pi(n)\cdot\frac{1}{\log_\pi(e)}} \\ &= n^{\frac{1}{\log_\pi(e)}} \\ &= n^{\ln(\pi)} \\ &\approx n^{1,14} \end{aligned}$$

Somit verbleiben zwei Fälle, die nicht direkt aus der bekannten Hierarchie des $\mathcal{O}$-Kalküls folgen. Zum einen gilt $n! \in \mathcal{O}(n^n)$. Dies sieht man, indem man die Faktoren paarweise vergleicht. Außerdem gilt (iv) vor (iii), da $\Theta(\sqrt{\log_2(n)})$ langsamer als logarithmisch wächst.

## Aufgabe 2 - Beweisen oder nicht beweisen, das ist hier die Frage (5 Punkte)

Beweise oder widerlege die folgenden Aussagen. Gib hierfür jeweils $n_0$ und $c$ aus der formalen Definition an und zeige, dass mit diesen Werten die Ungleichung der Definition gilt. Alternativ, nutze die alternative Definition über Grenzwerte, um eine Aussage zu widerlegen.
Beispiel: Für $f(n) \in \mathcal{O}(g(n))$ finde $n_0$ und $c$ mit $\forall n > n_0 : f(n) \leq c \cdot g(n)$ und zeige, das dies für deine Werte gilt.

a) $7 \cdot n^{42} + 4 \cdot n^{11} + 13 \cdot n^2 - 12 \cdot n + 7 \in \mathcal{O}(n^{42})$ (1 Punkt)

b) $\log(n) \cdot \log(n) \in \mathcal{O}(\log(\log(n)))$ (1 Punkt)

c) $2024 \cdot n^2 \in \mathcal{O}(3^n)$ (1 Punkt)

d) $\sqrt[5]{n^7} \in \Omega(\log^7(n))$ (2 Punkte)

## Lösung 2

a) Wir zeigen die Aussage.
Wähle $c = 31$ und $n_0 = 1$, denn dann gilt für alle $n > n_0$

$$\begin{aligned}
&7 \cdot n^{42} + 4 \cdot n^{11} + 13 \cdot n^2 - 12 \cdot n + 7 \\
&\leq 7 \cdot n^{42} + 4 \cdot n^{42} + 13 \cdot n^{42} + 7 \cdot n^{42} \\
&= 31 \cdot n^{42} \\
&= c \cdot n^{42}.
\end{aligned}$$

Hierbei haben wir benutzt, dass $n^a \geq n^b$ mit $a \geq b$ für alle $n \geq 1$.

b) Wir widerlegen die Aussage.
Wir nutzen zuerst, dass $m \geq \log(m)$ für alle $m > 1$ gilt. Daraus folgt $\log(n) \geq \log(\log(n))$ für alle $n > 1$. Falls $\log(n) \cdot \log(n) \in \mathcal{O}(\log(\log(n)))$, dann müsste also auch $\log(n) \cdot \log(n) \in \mathcal{O}(\log(n))$ gelten.

Um dies zu widerlegen, betrachten wir den Grenzwert $\lim_{n\to\infty} \frac{\log(n)\cdot\log(n)}{\log(n)} = \lim_{n\to\infty} \log(n) = \infty$.
Somit gilt $\log(n) \cdot \log(n) \in \omega(\log(n))$ und auch $\log(n) \cdot \log(n) \in \omega(\log(\log(n)))$.

c) Wir zeigen die Aussage.
Wähle $c = 2024$ und $n_0 = 1$, denn dann gilt für alle $n > n_0$

$$2024 \cdot n^2 \leq c \cdot 3^n = 2024 \cdot 3^n \Leftrightarrow n^2 \leq 3^n.$$

d) Wir zeigen die Aussage.
Wähle $c = (\frac{1}{5})^7$ und $n_0 = 1$, denn dann gilt für alle $n > n_0$ :

$$\begin{aligned} c \cdot \log^7(n) &= \left(\frac{1}{5}\right)^7 \cdot \log^7(n) \\ &= \left(\frac{1}{5} \cdot \log(n)\right)^7 \\ &= \left(\log(n^{\frac{1}{5}})\right)^7 \\ &= \log^7(\sqrt[5]{n}). \end{aligned}$$

Da für alle $m > 1$ gilt, dass $m \geq \log(m)$, und außerdem $\sqrt[5]{n^7} = n^{\frac{7}{5}} = (\sqrt[5]{n})^7$, folgt für alle $n > n_0$:

$$(\sqrt[5]{n})^7 \geq \log^7(\sqrt[5]{n})$$

## Aufgabe 3 - Die Mär vom großen und bösen ~~Wolf~~ Summenterm (4 Punkte)

In dieser Aufgabe sollst du einige wichtige Techniken zum Abschätzen von Summen im $\mathcal{O}$-Kalkül kennen lernen.
Zeige für jede der folgenden Summen, dass diese in der gegebenen Komplexitätsklasse liegt.

a) $f_1(n) := \sum_{i=0}^{\sqrt{n}} \frac{2^8}{4^i} \in \Theta(1)$ (1 Punkt)

b) $f_2(n) := \sum_{i=0}^{\log_3 n} \frac{3^i}{2^8} \in \Theta(n)$ (1 Punkt)

c) $f_3(n) := \sum_{i=0}^{\sqrt{n}} \frac{i^2}{2^8} \in \mathcal{O}(n \cdot \sqrt{n})$ (1 Punkt)

d) $f_4(n) := \sum_{i=\sqrt{n}}^{2\cdot\sqrt{n}} \left(\frac{n}{\log_6(2^n \cdot 3^n)}\right)^i \in \Theta(\sqrt{n})$ (1 Punkt)

## Lösung 3

a) Hierbei handelt es sich um eine geometrische Reihe, da $\sum_{i=0}^{\sqrt{n}} \frac{2^8}{4^i} = 2^8 \cdot \sum_{i=0}^{\sqrt{n}} \frac{1}{4^i}$.

Für geometrische Reihen gilt $\lim_{n\to\infty} \sum_{i=0}^{n} q^i = \frac{1}{1-q}$, wenn $q < 1$.

Somit gilt hier $\lim_{n\to\infty} \sum_{i=0}^{\sqrt{n}} \frac{1}{4^i} = \frac{1}{1-\frac{1}{4}} = \frac{4}{3}$. Also ist $\lim_{n\to\infty} f_1(n) = 2^8 \cdot \frac{4}{3}$.

Da wir das Wachstum von $f_1(n)$ durch eine Konstante beschränken können, gilt $f_1(n) \in \Theta(1)$. Hierbei ist zu beachten, dass die Funktion wächst und sich der Grenze daher von unten annährt.

b) Es gilt $\sum_{i=0}^{\log_3 n} \frac{3^i}{2^8} = \frac{1}{2^8} \cdot \sum_{i=0}^{\log_3 n} 3^i$ und der letzte Term in jeder Teilsumme ist immer mindestens das doppelte der restlichen Terme der Teilsumme zusammen. Das heißt die Summe wird durch ihren letzten Term dominiert. Somit haben wir $f_2(n) \in \Theta(\frac{1}{2^8} \cdot 3^{\log_3(n)}) = \Theta(n)$.

c) Da $\sum_{i=0}^{\sqrt{n}} \frac{i^2}{2^8} = \frac{1}{2^8} \cdot \sum_{i=0}^{\sqrt{n}} i^2$ genügt es die Summe ohne den Vorfaktor zu betrachten. Wir schätzen als nächstes die Summe nach oben ab. Die Summanden werden mit wachsendem i immer größer. Daher können wir alle Terme durch den Term für $i = \sqrt{n}$ abschätzen. Somit ist:

$$\begin{aligned}
\sum_{i=0}^{\sqrt{n}} i^2 &\leq \sum_{i=0}^{\sqrt{n}} (\sqrt{n})^2 \\
&= \sum_{i=0}^{\sqrt{n}} n \\
&= (\sqrt{n} + 1) \cdot n \in \mathcal{O}(n \cdot \sqrt{n})
\end{aligned}$$

d) Wir vereinfachen die Summanden unter Anwendung der Potenzregeln.

$$\begin{aligned}
\sum_{i=\sqrt{n}}^{2\cdot\sqrt{n}} \left(\frac{n}{\log_6(2^n \cdot 3^n)}\right)^i &= \sum_{i=\sqrt{n}}^{2\cdot\sqrt{n}} \left(\frac{n}{\log_6(6^n)}\right)^i \\
&= \sum_{i=\sqrt{n}}^{2\cdot\sqrt{n}} \left(\frac{n}{n}\right)^i \\
&= \sum_{i=\sqrt{n}}^{2\cdot\sqrt{n}} 1^i \\
&= \sum_{i=\sqrt{n}}^{2\cdot\sqrt{n}} 1 \\
&= \sqrt{n} + 1
\end{aligned}$$

Es ist also $f_4(n) \in \Theta(\sqrt{n})$.

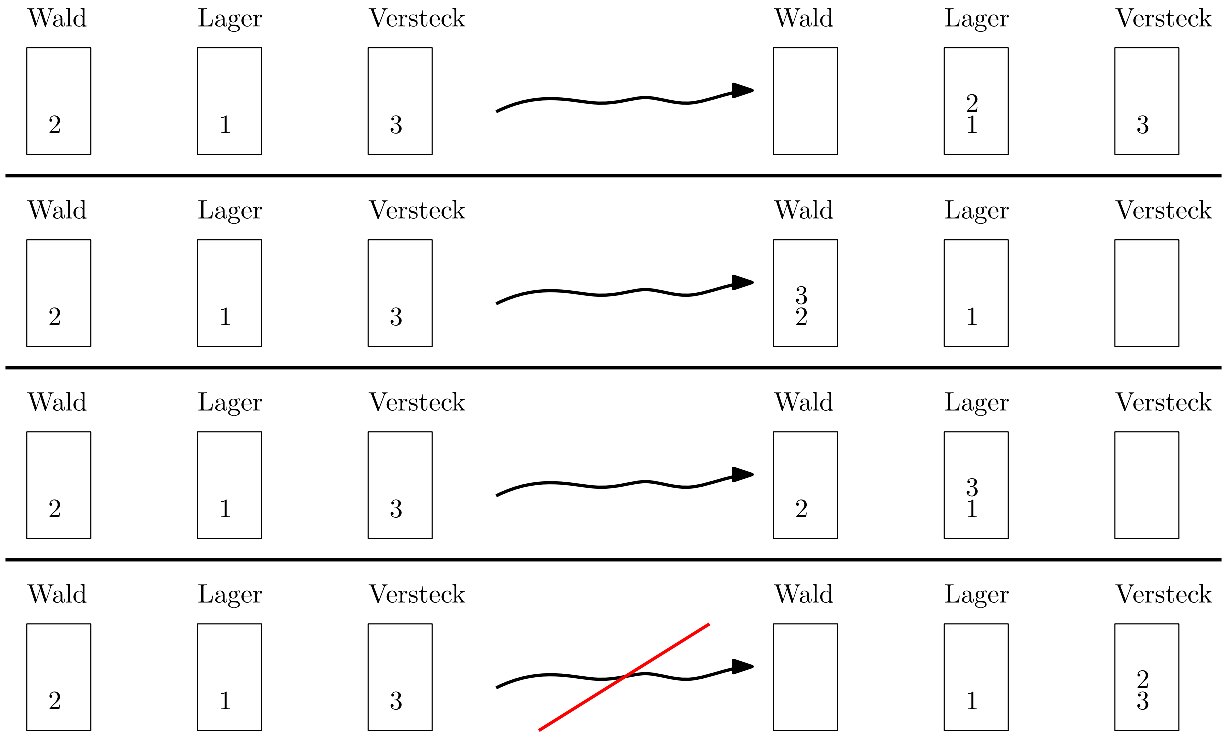
## **Aufgabe 4 - Hols Holz!** (8 Punkte)

Nach mehreren gescheiterten Versuchen, die Weltherrschaft zu übernehmen, nutzte Dr. Meta die vorlesungsfreie Zeit, um eine neue Strategie zu entwickeln. Während er die Details seines Plans noch geheim hält, hat er enthüllt, dass er für die Umsetzung erhebliche Mengen an Holz in seinem Versteck benötigt. Dafür hat er seine treuen Biber beauftragt, das benötigte Holz aus dem Wald zu seinem Versteck zu transportieren.

Jeder der $n$ Baumstämme ist mit einer eindeutigen Qualitätszahl von 1 bis $n$ markiert. Um die Qualität des Holzes zu bewahren, muss jeder Holzstapel stets so sortiert sein, dass die Stämme mit niedrigerer Qualitätszahl unten und jene mit höherer oben liegen. Das bedeutet, dass ein Baumstamm immer auf einen anderen gelegt werden muss. Die Holzfällbiber haben das gefällte Holz bereits entsprechend sortiert und gestapelt.

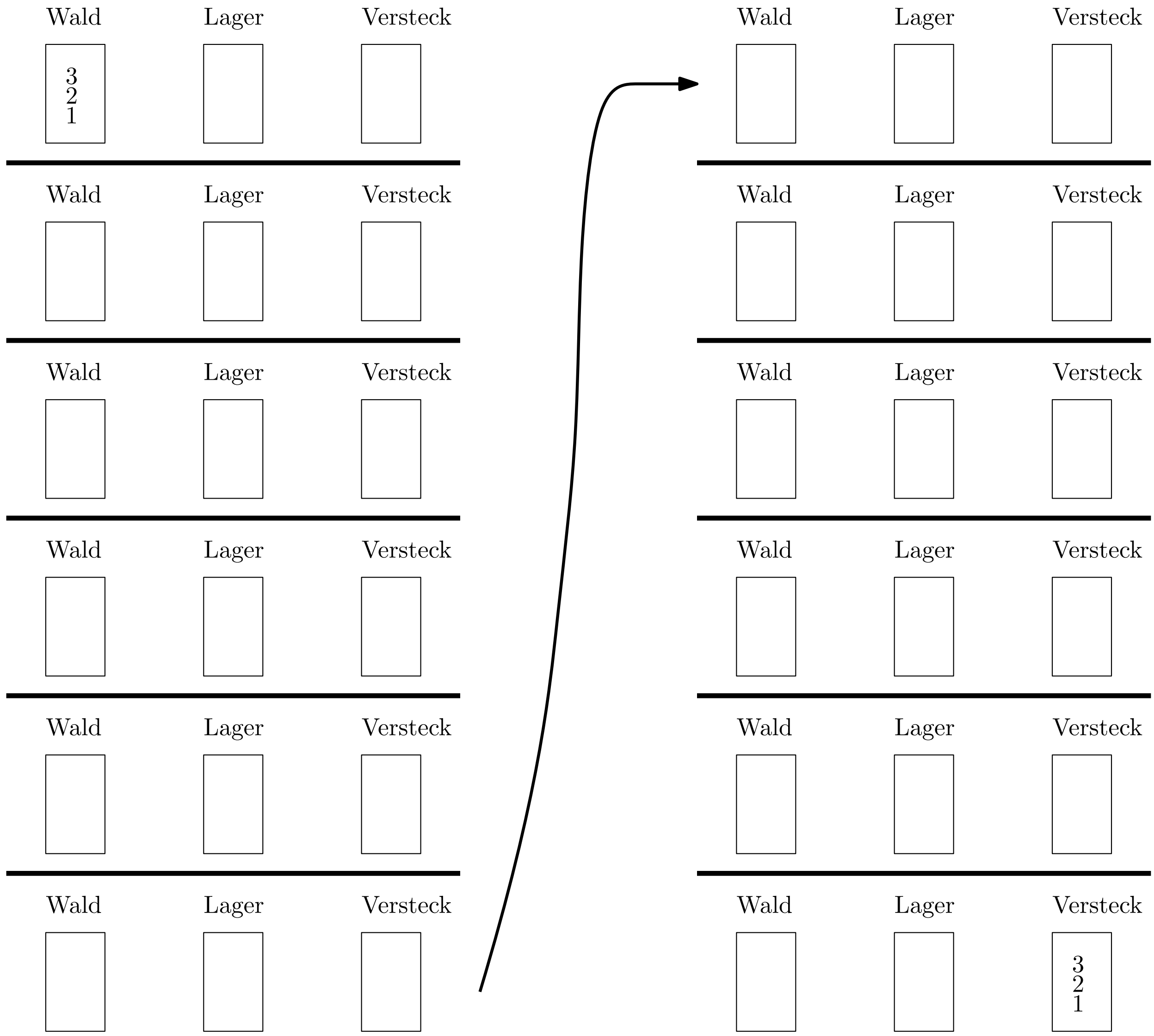
Dr. Meta stellt den Bibern für diese Aufgabe nur einen alten Lastkraftwagen und ein kleines Zwischenlager zur Verfügung. Der LKW kann pro Fahrt nur einen Baumstamm befördern. Sowohl im Wald, im Geheimversteck, als auch im Zwischenlager kann nur ein einziger Stapel Holz platziert werden. Es kann jeweils nur der oberste Baumstamm eines Stapels transportiert werden, welcher dann auf den Stapel am Zielort gelegt wird.

Die Aufgabe besteht darin, die Biber dabei zu unterstützen, das Holz mit möglichst wenigen Fahrten in das Versteck zu bringen. Dabei zählen nur die Fahrten, bei denen tatsächlich ein Baumstamm transportiert wird.




Ein gültiger Zustand, sowie alle durch gültige Fahrten erreichbaren Zustände. Außerdem ein Zustand der nicht durch eine gültige Fahrt erreicht werden kann. Jeder Baumstamm wird durch seine Qualitätszahl repräsentiert.

a) Gib für $n = 3$ eine Folge von Fahrten an, die alle Baumstämme mit minimaler Fahrtenzahl ins Versteck bewegt. Trage dafür den Zustand nach jeder Fahrt für alle drei Lagerorte (Wald (W), (Zwischen-)Lager (L), Versteck (V)) in der untenstehenden Darstellung ein. (1.5 Punkte) Hinweis: Es sind mehr Zustände als nötig angegeben. Streiche die nicht benötigten durch.




Darstellung des Start- und des Endzustandes der geforderten Folge von Fahrten, so wie freie Felder für Zwischenzustände.

b) Angenommen die Biber könnten die obersten $n-1$ Baumstämme eines Stapels auf einmal bewegen. Gib nun für allgemeine und ausreichend große $n$ an, wie du das Problem mit minimaler Fahrtzahl lösen kannst.
Du musst nicht zeigen, dass deine Lösung minimal ist. (1.5 Punkte)

c) Gib einen rekursiven Algorithmus, der das ursprüngliche Problem mit möglichst wenigen Fahrten löst, in Pseudocode an. Du kannst dafür die Erkenntnisse aus b) in abgewandelter Form und die Methode FAHREN(S : Lagerort, Z : Lagerort), die den obersten Baumstamm von S nach Z fährt, verwenden.
Deine Lösung soll hierbei die Methodensignatur HOLSHOLZ(S : Lagerort, Z : La-

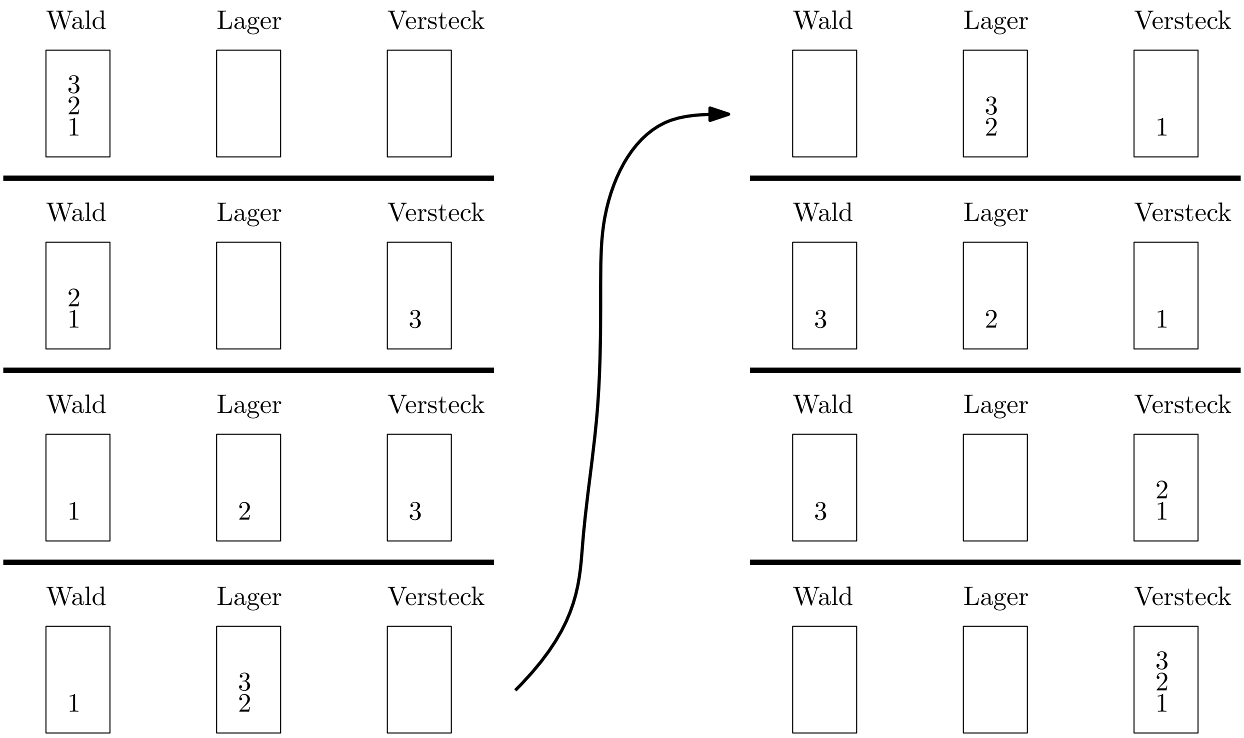
gerort, H : Lagerort, $n : \mathbb{N}_+$) haben, wobei die Methode $n$ Baumstämme von S nach Z unter Nutzung von H als Ablage transportiert. (2 Punkte)

d) Erkläre kurz warum es bei deinem Algorithmus zu keinen Problemen kommt, also warum nie ein Baumstamm auf einem qualitativ hochwertigeren gestapelt wird. (1 Punkt)

e) Gebe die von deinem Algorithmus benötigte Anzahl an Fahrten mit einem Baumstamm an. Stelle dafür zuerst eine Rekurrenzgleichung auf und analysiere diese anschließend. (2 Punkte)

## Lösung 4

a) Die unten gebene Abfolge mit 7 Fahrten ist optimal.




Eine Abfolge von 8 Zuständen mit 7 Fahrten.

b) Die Biber fahren die obersten $n-1$ Baumstämme ins Zwischenlager. Anschließend fahren sie den verbleibenden Baumstamm ins Versteck. Als letztes fahren sie die $n-1$ Baumstämme aus dem Lager ins Versteck.

c) Dem Algorithmus liegt die folgende Idee zu Grunde:
Wir können den Algorithmus aus b) nutzen und anstatt $n-1$ Baumstämme auf einmal zu transportieren, nutzen wir einen rekursiven Aufruf des Algorithmus mit

$n-1$ Baumstämmen. Hierbei werden ggf. die Rollen der Lager vertauscht. Im Basisfall fahren die Biber den einzigen Stamm direkt ins Versteck.

```
1: holsHolz(S : Lagerort, Z : Lagerort, H : Lagerort, n : ℕ+)
2:     if n = 1 then
3:         fahren(S, Z)
4:     else
5:         holsHolz(S, H, Z, n − 1)
6:         fahren(S, Z)
7:         holsHolz(H, Z, S, n − 1)
8:     end
```

Der Aufruf holsHolz(W, V, L, $n$) löst dann das Problem.

d) Im Fall von $n = 1$ wird offensichtlich nie ein minderwertiger Baumstamm auf einen höherwertigen gestapelt.

Wir betrachten den Fall $n > 1$ und nehmen an, dass der rekursive Aufruf korrekt ist, also nie ungültige Stapel erstellt. Dann ist auch der Aufruf von fahren in Zeile 6 korrekt. Daher kann ein Fehler, wenn überhaupt, nur beim zweiten rekursiven Aufruf von holsHolz auftreten. Dies ist aber nicht der Fall, da bis auf den qualitativ schlechtesten Stamm, auf dem alles gestapelt werden kann, die verbleibenden Lager leer sind. Formell kann die Behauptung mit Induktion gezeigt werden.

e) Die Rekurrenzgleichung für die Anzahl der Fahrten $F(n)$ lautet:

$$F(n) = \begin{cases} 1 & \mid n = 1 \\ 2 \cdot F(n-1) + 1 & \mid \text{sonst} \end{cases}$$

Da wir in jedem Schritt die Zahl der Fahrten verdopplen, liegt ein exponentiellen Zusammenhang vor. Es gilt $2 \cdot (2^n - 1) + 1 = 2^{n+1} - 2 + 1 = 2^{n+1} - 1$. Somit benötigt der Algorithmus $2^n - 1$ Fahrten.

## Aufgabe 5 - Zum Knobeln (0+1 Punkte)

Zeige, die zu Aufgabe 3 c) gehörende untere Schranke.

Zeige also $\sum\limits_{i=0}^{\sqrt{n}} \frac{i^2}{2^8} \in \Omega(n \cdot \sqrt{n})$. (1 Punkt)

## Lösung 5

Wir finden eine untere Schranke. Dafür schätzen wir die ersten $\frac{\sqrt{n}}{2}$ Terme nach unten durch 0 und die verbleibenden durch $\left(\frac{\sqrt{n}}{2}\right)^2$ nach unten ab. Dies ist möglich, da die Summanden mit wachsendem i immer größer werden.

Somit folgt:

$$
\begin{aligned}
\sum_{i=0}^{\sqrt{n}} i^2 &\geq \sum_{i=0}^{\frac{\sqrt{n}}{2}-1} 0 + \sum_{i=\frac{\sqrt{n}}{2}}^{\sqrt{n}} \left(\frac{\sqrt{n}}{2}\right)^2 \\
&= 0 + \sum_{i=\frac{\sqrt{n}}{2}}^{\sqrt{n}} \frac{n}{4} \\
&\geq \frac{\sqrt{n}}{2} \cdot \frac{n}{4} \\
&\in \Omega(n \cdot \sqrt{n})
\end{aligned}
$$