4.15 Satz
Zu $n \in \mathbb{N}$ gibt es n^{n-2} verschiedene Bäume mit Knotenmenge $\{1, \ldots, n\}$.

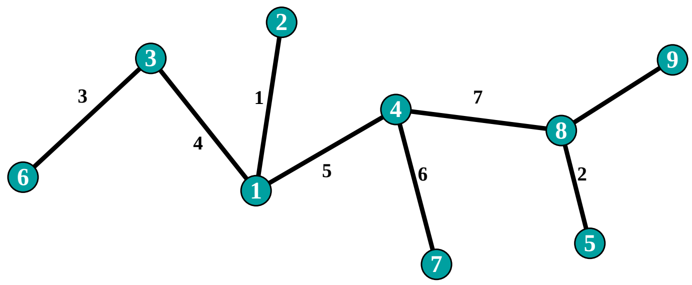
■ **Beweis:** Da die Aussage für $n = 1$ offensichtlich stimmt, können wir $n \geq 2$ annehmen. Wir wenden das Identitätsprinzip an, indem wir die Bäume bijektiv auf $\{1, \ldots, n\}^{n-2}$ abbilden.

Einem Baum wird ein Tupel $(v_1, \ldots, v_{n-2})$ wie folgt zugewiesen. Ein Baum hat immer einen Knoten vom Grad 1, denn wenn alle Knoten einen größeren Grad hätten, wäre die Summe der Knotengrade größer oder gleich $2n$, sodass aus dem Handschlaglemma 4.4 die Existenz von mindestens $\frac{2n}{2} = n$ Kanten folgte (Widerspruch zu Satz 4.13). Unter den Knoten vom Grad 1 sei v der kleinste. Der eindeutige Nachbar definiert v_1. Werden danach v und die eine zu v inzidente Kante entfernt, erhalten wir wieder einen Baum und wählen erneut den kleinsten Knoten vom Grad 1. Dessen Nachbar definiert v_2, usw. Das so konstruierte Tupel $(v_1, \ldots, v_{n-2})$ heißt *Prüfercode* des Baums. Man beachte, dass zum Schluss eine Kante übrig bleibt, da nur $n - 2$ Kanten entfernt werden.

Wir zeigen, dass die dadurch definierte Abbildung injektiv und surjektiv ist, indem wir zeigen, dass zu jedem $(v_1, \ldots, v_{n-2}) \in \{1, \ldots, n\}^{n-2}$ genau ein Baum mit Prüfercode $(v_1, \ldots, v_{n-2})$ existiert.

Wir stellen zunächst fest, dass ein Knoten, der im Baum den Grad d hat, im Prüfercode genau $d - 1$ mal auftritt (nämlich immer dann, wenn ein Nachbar entfernt wird, der Knoten selbst aber noch einen weiteren Nachbarn im Baum hat). Knoten vom Grad 1 treten im Prüfercode also nicht auf. Der kleinste nicht auftretenden Knoten muss derjenige sein, der als erster entfernt wurde, und sein Nachbar war v_1. Der kleinste von den $n-1$ anderen Knoten, der nicht in $(v_2, \ldots, v_{n-2})$ auftritt, muss Nachbar von v_2 gewesen sein, usw. Nachdem so $n - 2$ mal ein Knoten bestimmt worden ist, sind nur v_2 und ein weiterer Knoten übrig. Die letzte hinzuzufügenden Kante muss dann diese beiden verbinden und der bei der Erzeugung des Prüfercodes übrig gebliebenen entsprechen. □

4.16 Beispiel
Der Prüfercode des folgenden Baums ist $(1, 8, 3, 1, 4, 4, 8)$ *und die Kantenbeschriftungen geben die Reihenfolge an, in der sie entfernt wurden.*



Bei der Rekonstruktion werden die Kanten in derselben Reihenfolge eingefügt, die unbeschriftete zuletzt.

4.2 Durchläufe

In diesem Abschnitt werden wir Graphen *durchlaufen*, um Eigenschaften zu testen oder Teilgraphen zu identifizieren. Durchlaufen bedeutet dabei, an einem Knoten zu starten und jeweils von einem bereits besuchten Knoten über eine Kante den benachbarten Knoten aufzusuchen. Dazu werden zunächst die Definitionen von Wegen und Kreisen verallgemeinert.

4.17 Definition (Graphenhomomorphismus)
Gibt es zu zwei Graphen $G_1 = (V_1, E_1)$ *und* $G_2 = (V_2, E_2)$ *eine Abbildung* $\alpha : V_1 \to V_2$ *mit*

$$\{u, v\} \in E_1 \implies \{\alpha(u), \alpha(v)\} \in E_2 \ ,$$

dann heißt α *(Graphen)homomorphismus und wir nennen den Teilgraphen*

$$\alpha(G_1) = (\alpha(V_1), \{\{\alpha(u), \alpha(v)\} \in E_2 \, : \, \{u, v\} \in E_1\}) \subseteq G_2$$

homomorphes Bild von G_1 *in* G_2.