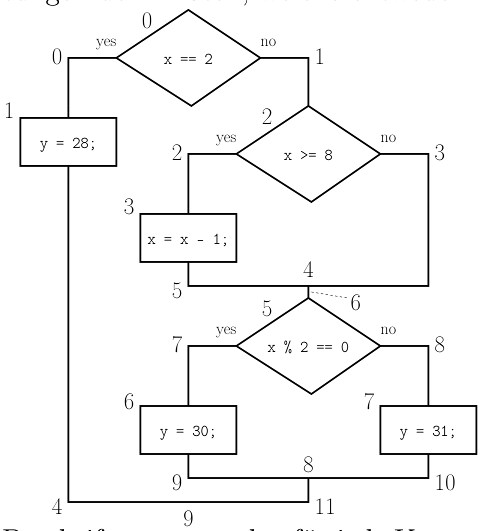
1) a) Betrachtet man Monate als Zahlen, mit Januar= 1, ..., Dezember= 12, und gibt man einen Monat als Eingabe, dann ist die Ausgabe die Anzahl der Tage des Monats (eines normalen Jahres).

b) Es gibt 10 Knoten und 12 Kanten. Wir nehmen ein beliebiges Set an Knoten $\{0, \dots, 9\}$ und dazugehörige Kanten $\{0, \dots, 11\}$, wie links im Flussdiagramm dargestellt. Daraus resultieren die Tabellen auf der rechten Seite. Die linke Tabelle enthält die Kanten und ihre Beschriftungen, welche yes, no oder nichts sind. Die andere Tabelle enthält die Beschriftungen der Knoten, welche entweder Tests, Statements, oder nichts sind.

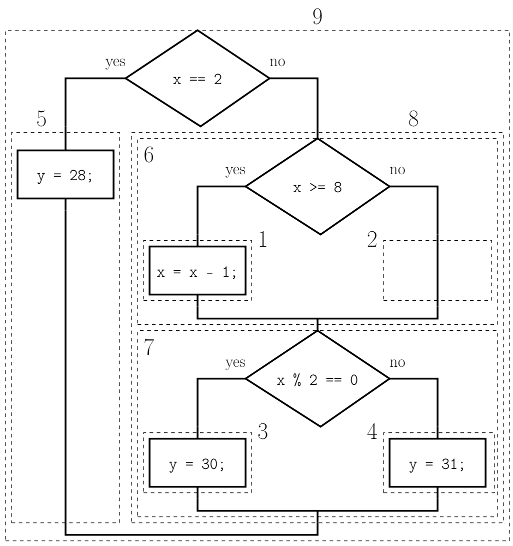


k	b(k)	q(k)	z(k)
0	yes	0	1
1	no	0	2
2	yes	2	3
3	no	2	4
4		1	9
5		3	4
6		4	5
7	yes	5	6
8	no	5	7
9		6	8
10		7	8
11		8	9

e	a(e)
0	`x == 2`
1	`y = 28;`
2	`x >= 8`
3	`x = x - 1;`
4	
5	`x % 2 == 0`
6	`y = 30;`
7	`y = 31;`
8	
9	

Beschriftungen werden für jede Kante von einem Test-Knoten benötigt, um unterscheiden zu können ob sie einer yes oder no Kante entspricht.

c) Wie der Flussgraph induktiv erstellt werden kann wird durch die nummerierten gestrichelten Boxen in



gezeigt. Boxen 1, 3, 4 und 5 sind Ausdrucks-Basisfälle, Box 2 ist ein leerer Basisfall. Wir haben dann Auswahlzusammensetzungsboxen 6 (aus 1 und 2), sowie 7 (aus 3 und 4), welche zu einer sequentiellen Zusammensetzungsbox 8 (aus 6 und 7) kombiniert werden. Schlussendlich werden 5 und 8 zu einer Auswahlzusammensetzungsbox 9 kombiniert, welche das gesamte Flussdiagramm darstellt.

Wir verwenden rekursive Inferenz (siehe Einführung in die Theoretische Informatik) um das Programm abzuleiten:

	Wort	Variable	Regel	Rekursion
1	`x == 2`	T	Test	
2	`y = 28`	S	Ausdruck	
3	`y = 28;`	P	$P \to S;$	2
4	`x >= 8`	T	Test	
5	`x = x -1`	S	Ausdruck	
6	`x = x -1;`	P	$P \to S;$	5
7	`;`	P	$P \to ;$	
8	`x % 2 == 0`	T	Test	
9	`y = 30`	S	Ausdruck	
10	`y = 30;`	P	$P \to S;$	9
11	`y = 31`	S	Ausdruck	
12	`y = 31;`	P	$P \to S;$	11
13	`if (x >= 8) {x = x - 1;} else {;}`	P	$P \to \texttt{if}\,(T)\,\{P\}\,\texttt{else}\,\{P\}$	$4, 6, 7$
14	`if (x % 2 == 0) {y=30;} else {y=31;}`	P	$P \to \texttt{if}\,(T)\,\{P\}\,\texttt{else}\,\{P\}$	$8, 10, 12$
15	`komplettes Programm aus Angabe`	P	$P \to \texttt{if}\,(T)\,\{P\}\,\texttt{else}\,\{P\}$	$1, 3, 14$

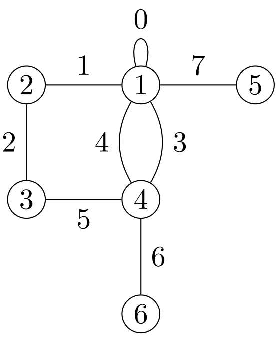
d)

$$
\begin{aligned}
f() &= 1\\
f(S;) &= 1\\
f(P_1\,P_2) &= f(P_1)\cdot f(P_2)\\
f(\texttt{if}\,(T)\,\{P_1\}\,\texttt{else}\,\{P_2\}) &= f(P_1)+f(P_2)
\end{aligned}
$$

Für das Programm P berechnen wir $5 = 1 + (1+1)\cdot(1+1)$.

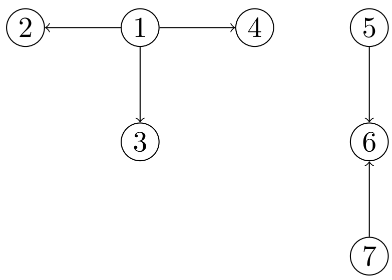
2) *Lösung.* Nach Definition ist ein Wurzelbaum $B = (E, K, q, z)$ ein gerichteter Graph, sodass eine ausgezeichnete Ecke $w \in E$ existiert und jede Ecke $e \in E$ mit einem eindeutigen Weg von w aus erreichbar ist. Angenommen, es exisitiert ein Zykel in B. Dann enthält dieser Zykel zumindest eine Ecke e. Nun ist e von w aus, auf unendlich vielen Wegen erreichbar. Das ist ein Widerspruch zur Definition. Im Besonderen gilt die Behauptung. $\square$

4) *Lösung.* G_1:



G_2:

G_3:



Es gibt unendlich viele Wege in G_1 von Ecke 1 nach Ecke 3, da man z.B. die Kante 0 bei Knoten 1 beliebig oft nehmen kann, bevor man zu Ecke 3 geht. Die einfachen Wege sind: $(1, 2), (3, 5), (4, 5)$

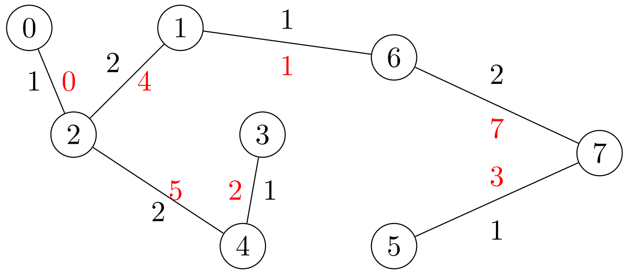
G_1 hat die einfachen Zykel $(0), (3, 4), (4, 3)$, die 8 möglichen Zykel die durch die Kanten $(1, 2, 5, 3)$ entstehen (je 4 verschiedene Startknoten, und 2 mögliche Richtungen) sowie die 8 Zykel die man durch Vertauschen von 3 mit 4 erhält. Insgesamt hat G_1 also 19 *einfache* Zykel.

G_2 und G_3 haben keine Zykel.

G_1 und G_2 sind zusammenhängend.

G_1 ist nicht zyklenfrei also weder ein Baum noch ein Wald. G_2 ist ein Baum. G_3 ist kein ungerichteter Multigraph, also ist die Definition nicht direkt anwendbar. Der Graph, der entsteht, wenn man den gerichteten Graphen in einen ungerichteten überführt, indem man die Kantenorientierung weglässt, wäre ein Wald aber kein Baum. □

5) *Lösung.* Wir entfernen parallele Kanten (bis auf jene mit minimaler Bewertung) und Schleifen und sortieren die verbleibenden Kanten nach aufsteigender Bewertung: 0, 2, 10, 11, 3, 4, 5, 13, 7, 8, 9, 1. Der folgende Graph veranschaulicht das Ergebnis, das der Algorithmus von Kruskal liefert, wobei die rote Zahl den Schritt i angibt, in welchem die Kante hinzugefügt wurde. Die schwarze Zahl gibt die Kantenbewertung an.



□