Algorithmen und Datenstrukturen **Winter 2023/24**
Prof. Dr. Sándor P. Fekete
Ramin Kosfeld
Chek-Manh Loi

Präsenzblatt 0

Dieses Blatt dient lediglich der persönlichen Vorbereitung. Es wird nicht abgegeben und geht nicht in die Bewertung ein. Die Besprechung der Aufgaben und ihrer Lösungen erfolgt in den kleinen Übungen in der Woche vom 13.11.2023.

Präsenzaufgabe 1 (Begriffe der Graphentheorie):
Bestimme für den in Abbildung 1 dargestellten Graphen $G = (V, E)$ das Folgende:

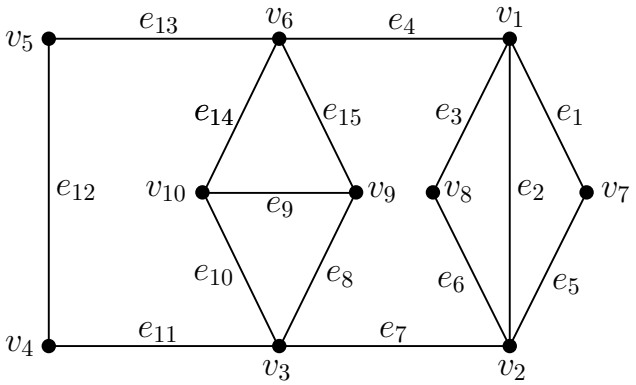


Abbildung 1: Der Graph G

a) Alle zu v_1 adjazenten Knoten,
b) alle zu v_1 inzidenten Kanten,
c) eine Kantenfolge von v_1 nach v_4, die kein Weg ist,
d) einen Weg von v_1 nach v_4, der kein Pfad ist,
e) einen Pfad von v_1 nach v_4,
f) einen Eulerweg (oder begründe, dass dieser nicht existiert),
g) einen Hamiltonpfad (oder begründe, dass dieser nicht existiert),
h) einen Hamiltonkreis (oder begründe, dass dieser nicht existiert).

Präsenzaufgabe 2 (Fibonacci-Zahlen):
Die Fibonacci-Zahlen sind rekursiv definiert. So berechnet sich eine bestimmte Fibonacci-Zahl aus der Summe der beiden vorherigen Fibonacci-Zahlen. Konkret: $F(n) := F(n-1) + F(n-2)$, wobei $F(0) = 0$ und $F(1) = 1$. Wir wollen nun zwei Algorithmen (siehe Algorithmus 1) testen, die $F(n)$ berechnen.

a) Berechne $F(5)$ mit FibonacciRek und gib dabei *alle* Aufrufe der Funktion an. Wie oft wird eine Summe berechnet? Hat FibonacciRek alle Eigenschaften eines Algorithmus?

b) Berechne $F(5)$ mit Fibonacci und gib dabei die Werte aller $f(i)$ aus. Wie viele Summen wurden berechnet? Hat Fibonacci alle Eigenschaften eines Algorithmus?

```
1: function FibonacciRek(n)
2:     if n ≤ 1 then
3:         return n
4:     a := FibonacciRek(n − 1)
5:     b := FibonacciRek(n − 2)
6:     return a + b
```

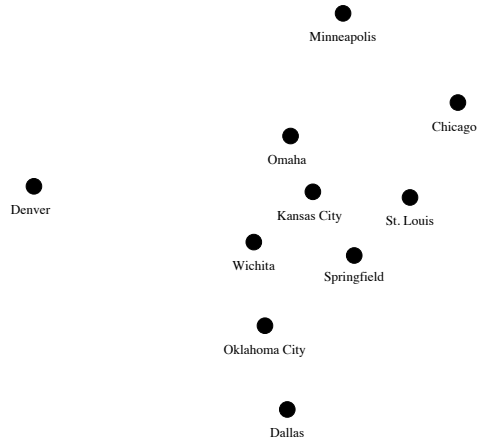
```
1: function Fibonacci(n)
2:     f(0) := 0
3:     f(1) := 1
4:     for i in 2, . . . , n do
5:         f(i) := f(i−1)+f(i−2)
6:     return f(n)
```

Algorithmus 1: Zwei Algorithmen für $F(n)$: Rekursiv (links) und „mit Gedächtnis“(rechts)

Präsenzaufgabe 3 (kürzeste Pfade):
Sei $G = (V, E)$ ein einfacher Graph. Für einen Pfad $P = (v_1, \ldots, v_i)$ ist die Länge von P die Anzahl der Kanten auf diesem Pfad. Sei nun P ein kürzester Pfad von Knoten v_1 nach v_i.

Zeige oder widerlege: Jeder Teilpfad $P' = (v_k, \ldots, v_\ell)$ von P mit $1 \leq k \leq \ell \leq i$ ist ein kürzester Pfad zwischen v_k und v_ℓ.

Präsenzaufgabe 4 (Rundreise):
Abbildung 2 zeigt Städte der USA und eine Distanzmatrix. Die Zahlen geben die Entfernung pro 10 Meilen an. Beispielsweise ist die Distanz von Chicago nach Dallas 920 Meilen. Ein Handelsreisender möchte nun alle Städte ablaufen und dabei möglichst wenig Meilen zurücklegen.



		2	3	4	5	6	7	8	9	10
1	Chicago	92	99	50	41	79	46	29	50	70
2	Dallas		78	49	94	21	64	63	42	37
3	Denver			60	84	61	54	86	76	51
4	Kansas City				45	35	20	26	17	20
5	Minneapolis					80	36	55	59	64
6	Oklahoma City						46	50	29	16
7	Omaha							45	37	30
8	St. Louis								21	45
9	Springfield									25
10	Wichita									

Abbildung 2: Städte der USA (links) und eine Distanzmatrix (rechts, je 10 Meilen)

a) Wie viele Kanten braucht man für eine Tour? (Hinweis: Diese Frage bezieht sich nur auf die Anzahl, sie ist unabhängig von den zurückgelegten Distanzen.)

b) Finde eine möglichst kurze Tour, d.h. eine für die die Summe der Entfernungen der ausgewählten Kanten möglichst klein ist.

c) Wie gut ist Deine Tour? Gib eine Mindestdistanz für *jede mögliche* Städterundreise an und vergleiche diese mit der Länge der von Dir gefundenen Tour.