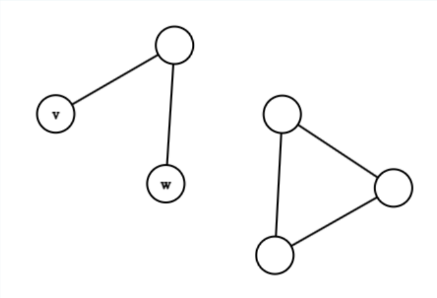
QUIZ-NACHBESPRECHUNG



Wahr oder falsch: Im Graph oben sind die Knoten mit den Bezeichnungen v und w benachbart.

Bitte wählen Sie eine Antwort:

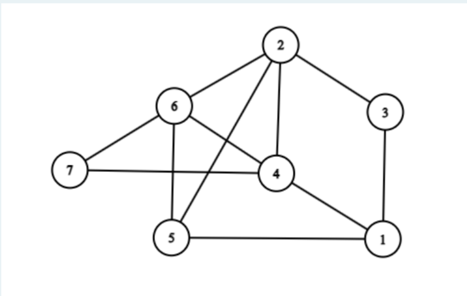
- ○ Wahr
- ☒ Falsch

Sei $G = (V, E)$ ein Graph mit 4 Knoten v_1, v_2, v_3, v_4. Nehme an, dass

$$\deg(v_1) = 1,\ \deg(v_2) = 2,\ \deg(v_3) = 3,\ \deg(v_4) = 2.$$

Wie viele Kanten hat G? *(Die Antwort sollte aus einer einzelnen ganzen Zahl bestehen.)*

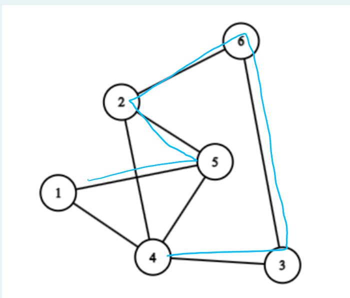
Antwort: 4



Wahr oder falsch: Der Graph oben hat einen Eulerzyklus.

Bitte wählen Sie eine Antwort:

- ○ Wahr
- ☒ Falsch



Wahr oder falsch: Der Graph oben hat einen Hamiltonpfad.

Bitte wählen Sie eine Antwort:

- ☒ Wahr
- ○ Falsch

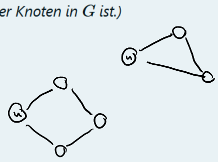
Sei $G = (V, E)$ und nehmen Sie an, dass *alle Knoten von G einen geraden Knotengrad haben*. Erinnern Sie sich an den Algorithmus **walk** aus der Vorlesung:

> $\mathbf{walk}(u)$:
>   **falls** eine Kante $\{u, v\}$ existiert, die nicht markiert ist:
>   markiere die Kante $\{u, v\}$
> $\mathbf{walk}(v)$

Sei $u \in V$ ein Knoten von G. Welche der folgenden Aussagen muss nach der Ausführung von $\mathbf{walk}(u)$ wahr sein?

(Folgend wird eine Kante als inzident zu u *bezeichnet, wenn sie die Form $\{u, v\}$ hat, wobei v ein anderer Knoten in G ist.)*

Wählen Sie eine oder mehrere Antworten:



- ☐ a. Die gesamte Anzahl markierter Kanten in G ist gerade.
- ☐ b. Die gesamte Anzahl markierter Kanten in G ist ungerade.
- ☒ c. Die gesamte Anzahl der nicht markierten Kanten, die *inzident zu u* sind, ist gerade.
- ☐ d. Die gesamte Anzahl der nicht markierten Kanten, die *inzident zu u* sind, ist ungerade.

SERIE 6 - COMMON MISTAKES

- zu viele (unnötige) Base Cases

- keine Nutzung des memo-Arrays

- Überprüfung, ob ein Wert in memo schon berechnet worden ist
  - → Initialisierung mit -1 (oder einem anderen nicht möglichem Resultat)
  - → check: if (memo[n] != -1) ...

- maximum almost subset sum: "possibley skip"
  → ein Eintrag musste nicht übersprungen werden
  → machte nur Sinn, falls Eintrag negativ

NACHBESPRECHUNG SERIE 7

Exercise 7.2 *Road trip.*

You are planning a road trip for your summer holidays. You want to start from city C_0, and follow the only road that goes to city C_n from there. On this road from C_0 to C_n, there are $n-1$ other cities $C_1, \ldots, C_{n-1}$ that you would be interested in visiting (all cities $C_1, \ldots, C_{n-1}$ are on the road from C_0 to C_n). For each $0 \leq i \leq n$, the city C_i is at kilometer k_i of the road for some given $0 = k_0 < k_1 < \ldots < k_{n-1} < k_n$.

You want to decide in which cities among $C_1, \ldots, C_{n-1}$ you will make an additional stop (you will stop in C_0 and C_n anyway). However, you do not want to drive more than d kilometers without making a stop in some city, for some given value $d > 0$ (we assume that $k_i < k_{i-1} + d$ for all $i \in [n]$ so that this is satisfiable), and you also don't want to travel backwards (so from some city C_i you can only go forward to cities C_j with $j > i$).

(a) Provide a *dynamic programming* algorithm that computes the number of possible routes from C_0 to C_n that satisfy these conditions, i.e., the number of allowed subsets of stop-cities. Your algorithm should have $O(n^2)$ runtime.

In your solution, address the following aspects:

1) Dimensionen der DP-Tabelle: $1 \times (n+1)$

2) Teilproblem: DP[i] = # mögliche Wege von C_0 bis C_i, die in C_i stoppen

3) Rekursion: DP[0] = 1 (B.C.)

$$DP[i] = \sum_{\substack{0 \leq j < i \\ k_i - k_j \leq d}} DP[j]$$

4) Berechnungsreihenfolge: mit steigendem i

5) Auslesen der Lösung: DP[n]

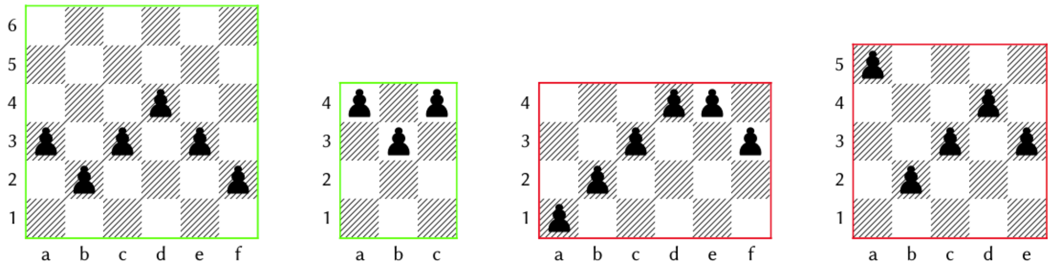
6) Laufzeit: $O(n^2)$

(b) If you know that $k_i > k_{i-1} + d/10$ for every $i \in [n]$, how can you turn the above algorithm into a linear time algorithm (i.e., an algorithm that has $O(n)$ runtime) ?

Exercise 7.3 *Safe pawn lines.*

On an $N \times M$ chessboard (N being the number of rows and M the number of columns), a *safe pawn line* is a set of M pawns with exactly one pawn per column of the chessboard, and such that every two pawns from adjacent columns are located diagonally to each other. When a pawn line is not safe, it is called *unsafe*.

The first two chessboards below show safe pawn lines, the latter two unsafe ones. The line on the third chessboard is unsafe because pawns d4 and e4 are located on the same row (rather than diagonally); the line on the fourth chessboard is unsafe because pawn a5 has no diagonal neighbor at all.



Describe a DP algorithm that, given $N, M > 0$, counts the number of safe pawn lines on an $N \times M$ chessboard. Your solution should have complexity at most $O(NM)$.

1) Dimensionen der DP-Tabelle: $N \times M$

2) Teilproblem: $DP[i,j]$ = #safe pawn lines auf $N \times j$ Schachbrett mit dem letzten Bauern in Zeile i

3) Rekursion: $DP[i,j] = DP[i-1,j-1] + DP[i+1,j-1]$

$DP[1,j] = DP[2,j-1]$

$DP[N,j] = DP[N-1,j-1]$

$DP[i,1] = 1$ (Base Case)

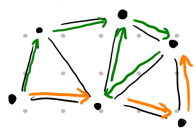
4) Berechnungsreihenfolge: steigendes j

5) Auslesen der Lösung: $\sum_{i=1}^{N} DP[i,M]$

6) Laufzeit: $O(MN)$

EINFÜHRUNG GRAPHENTHEORIE

- Ein **Graph** $G=(V,E)$ ist definiert als Tupel, wobei
  - $V=\{v_1,\dots,v_n\}$ eine Menge von Knoten
  - $E \subseteq V \times V$ eine Menge von Kanten
- $u,v \in V$, u und v **benachbart** (adjacent), falls es eine Kante zwischen u und v gibt
- **Weg**: Folge von benachbarten Knoten, Länge = #Kanten
- **Pfad**: Folge von benachbarten Knoten ohne wiederholte Knoten



Weg der Länge 5

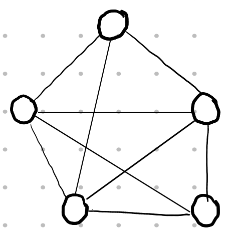
Pfad der Länge 3

- **Zyklus**: Weg mit $v_0 = v_\ell$, $\ell \geq 2$
- $u,v \in V$, u **erreicht** v $\Leftrightarrow$ $\exists$ Weg zwischen u und v (Äquivalenzrelation)
- **Zusammenhangskomponente (ZHK)**: Äquivalenzklassen
  G zusammenhängend $\Leftrightarrow$ #ZHK = 1
- **Eulerweg**: Weg, der jede Kante einmal durchläuft
- **Eulerzyklus**: Zyklus, der jede Kante einmal durchläuft

check in $O(m+n)$

- **Hamiltonpfad**: Pfad, der jeden Knoten einmal besucht
- **Hamiltonkreis**: Hamiltonpfad, aber Endknoten = Startknoten

NP-schwer

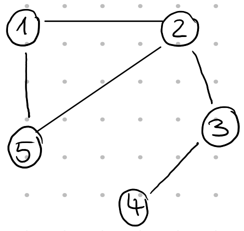


Eulerpfad?
Eulerkreis?
Hamiltonpfad?

G enthält **Eulerweg** $\Leftrightarrow$ alle außer max. zwei Knoten besitzen einen geraden Knotengrad und G ist zusammenhängend

G enthält **Eulerkreis** $\Leftrightarrow$ alle Knoten haben geraden Grad und G ist zusammenhängend

Handschlaglemma: im ungerichteten Graphen $G=(V,E)$ gilt: $\sum_{v \in V} \deg(v) = 2|E|$



Adjazenzmatrix

	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	0	1
3	0	1	0	1	0
4	0	0	1	0	0
5	1	1	0	0	0

bei ungerichteten Graphen
immer symmetrisch

Adjazenzliste

1	2	3	4	5
2	1	2	3	1
5	3	4		2
	5			

Beweis-/Widerlegungsaufgaben (Bonus HS23)

Exercise 8.5 *Short questions about graphs* **(2 points).**

In the following, let $G = (V, E)$ be a graph, $n = |V|$ and $m = |E|$.

(a) Let $v \neq w \in V$. Prove that if there is a walk with endpoints v and w, then there is a path with endpoints v and w.

- Weg W: $v = v_0, v_1, \dots, v_n = w$

  Case ①: W ist Pfad

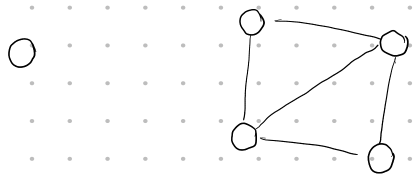
  Case ②: W ist kein Pfad

  W: $v_0, v_1, \dots, v_i, \underbrace{\dots, v_j}, \dots v_n$ $\quad v_i$

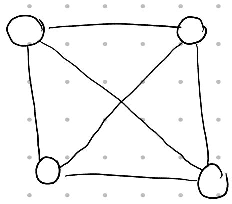
  O.B.d.A. $v_i = v_j$

For each of the following statements, decide whether the statement is true or false. If the statement is true, provide a proof; if it is false, provide a counterexample.

(b) Every graph with $m \geq n$ is connected.



(c) If G contains a Hamiltonian path, then G contains a Eulerian walk.



(d) If every vertex of a non-empty graph G has degree at least 2, then G contains a cycle.

Annahme: G hat keinen Kreis $\rightarrow$ alle ZHK's von G sind Bäume

Bäume haben $n-1$ Kanten

jeder Knoten Grad ≥ 2 $\Rightarrow$ Handschlaglemma mind. n Kanten ↯

alte Prüfungsaufgabe (FS23)

/ 3 P

c) *Graph quiz:* For each of the following claims, state whether it is true or false. You get 1P for a correct answer, -1P for a wrong answer, 0P for a missing answer. You get at least 0 points in total.

As a reminder, here are a few definitions:

A *walk* is a sequence of vertices $v_1, \ldots, v_k$ such that for every two consecutive vertices v_i, v_{i+1} there exists an edge from v_i to v_{i+1}.

A *path* $v_1, \ldots, v_k$ is a walk with the additional property that $v_i \neq v_j$ whenever $i \neq j$ (i.e., all vertices are distinct).

A length-$(k-1)$ *simple cycle* $v_1, \ldots, v_k$ is a walk where additionally $v_1 = v_k$, $v_i \neq v_j$ whenever $i < j < k$ (i.e., all vertices except the endpoints are distinct), and $k \geq 4$ (so $v_1 \to v_2 \to v_1$ is not allowed).

An *Eulerian* tour is a walk in a graph that visits every edge exactly once and returns to the starting vertex.

A graph is *Eulerian* if it contains an Eulerian tour.

Claim	**true**	**false**
Every vertex in a connected Eulerian graph $G = (V, E)$ with $\|V\| \geq 2$ has at least 2 neighbors.	☒	☐
In a graph suppose there exists a path with endpoints a and b, and a path with endpoints b and c(a, b, c are distinct vertices). Then there exists a path with endpoints a,c.	☒	☐
For any tree $T = (V, E)$ with $\|V\| \geq 10$, we can always add a single edge $e \notin E$ between two vertices in V such that the resulting graph contains a simple cycle of length 4 as a subgraph. (The vertex set is not allowed to change.)	☐	☒