ETH zürich

Übung 11 – Dynamische Programmierung

Informatik II

29. / 30. April 2025

Heutiges Programm

- Mars-Mission
- Der Weg des Königs
- Der Weg des gierigen Königs

Letzte Woche: Dynamische Programmierung

Gibt es Fragen?

- Tribonacci
- Catalan-Zahlen
- Blöcke
- Aufgabeplanung v2.0

1. Mars-Mission

Problem Beschreibung

Die Aufgabe betrifft einen Rover namens Perseverance, der auf dem Mars an einer mit S (Start) bezeichneten Position gelandet ist.

S	9	2	5	11	8
17	21	32	5	15	3
2	2	3	8	1	5
8	2	8	11	15	9
0	5	3	10	4	Z

Problem Beschreibung

Der Rover will eine Zielposition Z (Ende) erreichen

S	9	2	5	11	8
17	21	32	5	15	3
2	2	3	8	1	5
8	2	8	11	15	9
0	5	3	10	4	Z

Problem Beschreibung

Der Rover kann sich nur in Richtung Osten (rechts) oder Süden (unten) bewegen.

S	9	2	5	11	8
17	21	32	5	15	3
2	2	3	8	1	5
8	2	8	11	15	9
0	5	3	10	4	Z

Problem Beschreibung

Jede Zelle des Gitters enthält einen Wert, der eine Gesteinsprobe darstellt, die vom Rover gesammelt werden kann, wenn er diese Zelle passiert.

S	9	2	5	11	8
17	21	32	5	15	3
2	2	3	8	1	5
8	2	8	11	15	9
0	5	3	10	4	Z

Problem Beschreibung

Ziel ist es, einen Weg von S nach Z zu finden, der den Gesamtwert der gesammelten Gesteinsproben maximiert.

S	9	2	5	11	8
17	21	32	5	15	3
2	2	3	8	1	5
8	2	8	11	15	9
0	5	3	10	4	Z

Problem Beschreibung

- **Eingabe:** Eine $m \times n$ Matrix A

Problem Beschreibung

- **Eingabe:** Eine $m \times n$ Matrix A
- **Ausgabe 1:** Maximal möglicher Wert, der an Steinen gesammelt werden kann, die auf einem Süd-Ost-Pfad von $(0, 0)$ nach $(m-1, n-1)$ liegen.

Problem Beschreibung

- **Eingabe:** Eine $m \times n$ Matrix A
- **Ausgabe 1:** Maximal möglicher Wert, der an Steinen gesammelt werden kann, die auf einem Süd-Ost-Pfad von $(0, 0)$ nach $(m-1, n-1)$ liegen.
- **Ausgabe 2:** Der Pfad, der diesem Maximalwert entspricht.

Schritt 1: Teilprobleme identifizieren

Was sind die Teilprobleme?

Schritt 2: Rekursion identifizieren

Welche Optionen haben Sie, wenn Sie sich an der Position (i, j) befinden?

Schritt 2: Rekursion identifizieren

Definieren Sie eine Rekursion mittels der zuvor definierten Handlungsoptionen.

Schritt 3: Implementierung

Datenstruktur. Welche Datenstruktur benötigen Sie, um die Lösungen für die Teilprobleme zu speichern?

Schritt 3: Implementierung

Abhängigkeiten. Was benötigen Sie, um $S(i, j)$ zu berechnen?

Schritt 3: Implementierung

Berechnungsreihenfolge. Wie sollten Sie S ausfüllen?

Schritt 3: Implementierung

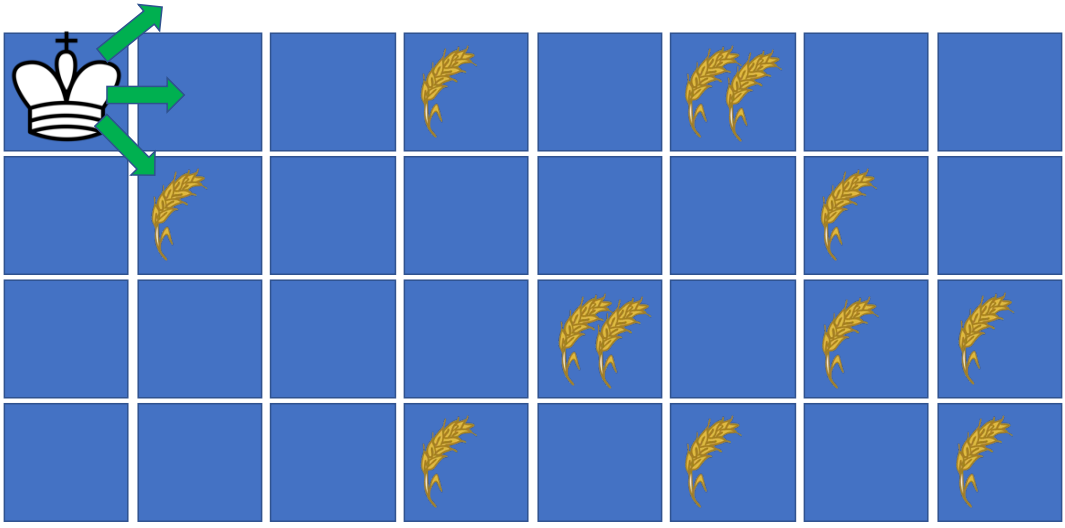
Extraktion der Lösung.

Schritt 3: Implementierung

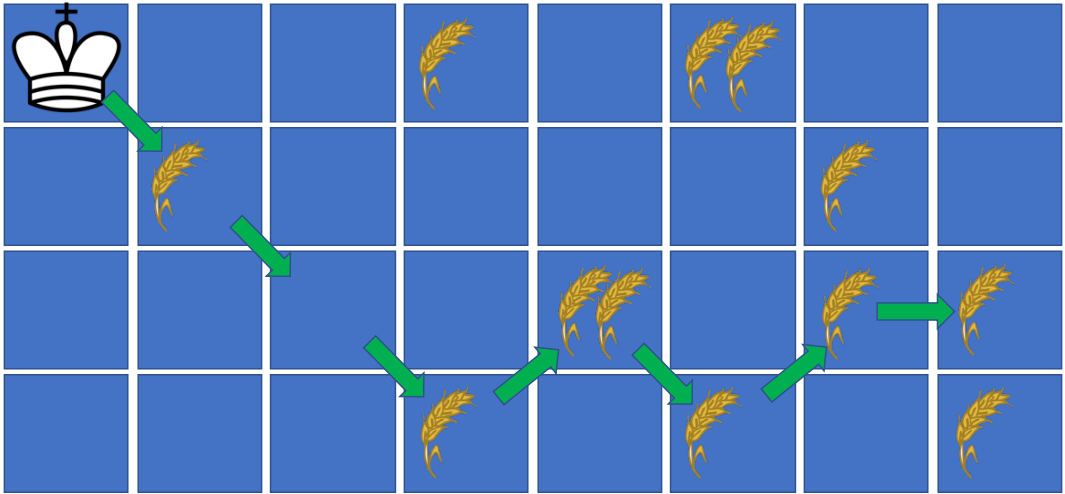
Wie lange ist die Laufzeit?

2. Königsweg

Problem Beschreibung



Problem Beschreibung



Problem Beschreibung

- Ein König befindet sich auf der Zelle $(0, 0)$ eines $m \times n$ Schachbretts.

Problem Beschreibung

- Ein König befindet sich auf der Zelle $(0, 0)$ eines $m \times n$ Schachbretts.
- Der König kann sich nur nordostwärts, ostwärts oder südostwärts bewegen. Das bedeutet, wenn der König sich in der Position (i, j) befindet, kann er nur zu $(i - 1, j + 1)$, $(i, j + 1)$, oder $(i + 1, j + 1)$ ziehen, solange er nicht das Spielfeld verlässt.

Problem Beschreibung

- Ein König befindet sich auf der Zelle $(0, 0)$ eines $m \times n$ Schachbretts.
- Der König kann sich nur nordostwärts, ostwärts oder südostwärts bewegen. Das bedeutet, wenn der König sich in der Position (i, j) befindet, kann er nur zu $(i - 1, j + 1)$, $(i, j + 1)$, oder $(i + 1, j + 1)$ ziehen, solange er nicht das Spielfeld verlässt.
- Jede Zelle ist mit einer Belohnung markiert.

Problem Beschreibung

- Ein König befindet sich auf der Zelle $(0, 0)$ eines $m \times n$ Schachbretts.
- Der König kann sich nur nordostwärts, ostwärts oder südostwärts bewegen. Das bedeutet, wenn der König sich in der Position (i, j) befindet, kann er nur zu $(i - 1, j + 1)$, $(i, j + 1)$, oder $(i + 1, j + 1)$ ziehen, solange er nicht das Spielfeld verlässt.
- Jede Zelle ist mit einer Belohnung markiert.
- Welches ist der beste Weg, den der König gehen kann, um die maximale Belohnung zu erhalten?

Problem Beschreibung

- Ein König befindet sich auf der Zelle $(0, 0)$ eines $m \times n$ Schachbretts.
- Der König kann sich nur nordostwärts, ostwärts oder südostwärts bewegen. Das bedeutet, wenn der König sich in der Position (i, j) befindet, kann er nur zu $(i - 1, j + 1)$, $(i, j + 1)$, oder $(i + 1, j + 1)$ ziehen, solange er nicht das Spielfeld verlässt.
- Jede Zelle ist mit einer Belohnung markiert.
- Welches ist der beste Weg, den der König gehen kann, um die maximale Belohnung zu erhalten?
- Belohnungen sind nicht negativ.

Problem Beschreibung

- **Eingabe:** Eine 2D-Liste `a` der Grösse $m \times n$ von nicht negativen Ganzzahlen.

Problem Beschreibung

- **Eingabe:** Eine 2D-Liste `a` der Grösse $m \times n$ von nicht negativen Ganzzahlen.
- **Ausgabe:** Der maximale Wert, den der König erreichen kann, und die Sequenz der Züge, um ihn zu erreichen.

Schritt 1: Teilprobleme identifizieren

Was sind die Teilprobleme?

Schritt 2: Rekursion identifizieren

Was sind meine Optionen?

Schritt 2: Rekursion identifizieren

Verwenden Sie die Optionen, um eine Rekursion zu definieren.

Schritt 3: Implementierung

Datenstruktur. Welche Datenstruktur benötigen Sie, um die Lösungen für die Teilprobleme zu speichern?

Schritt 3: Implementierung

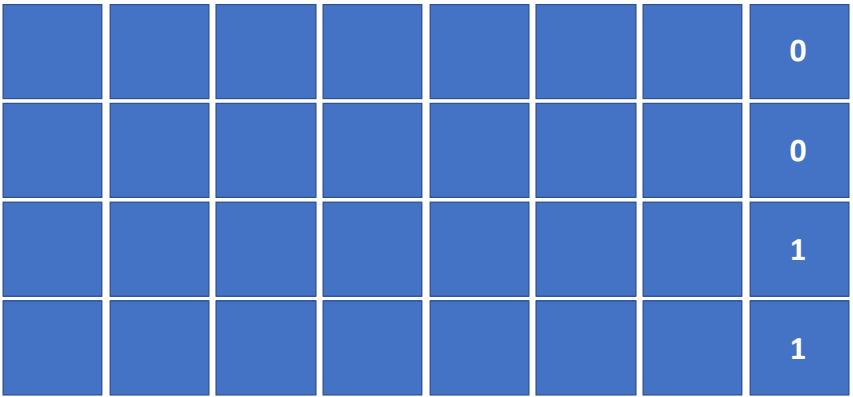
Abhängigkeiten. Was benötigen Sie, um $S(i, j)$ zu berechnen?

Schritt 3: Implementierung

Berechnungsreihenfolge. Wie sollten Sie `s` ausfüllen?

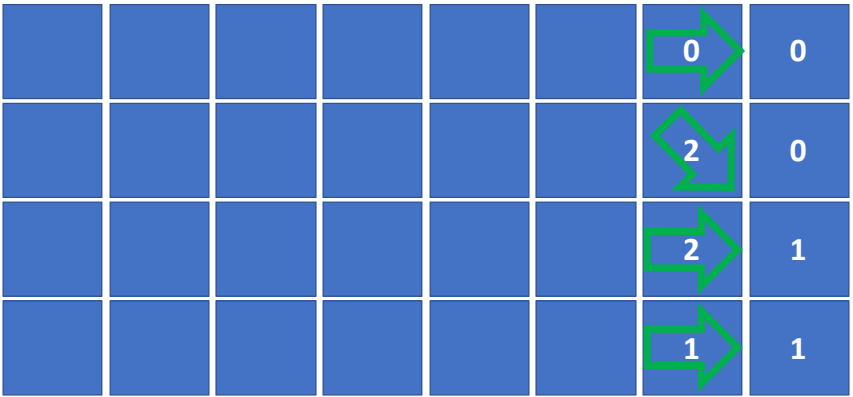
Schritt 3: Implementierung

Füllen Sie die Matrix `s` aus. Markieren Sie für jedes Teilproblem die optimale Entscheidung.



Schritt 3: Implementierung

Füllen Sie die Matrix `s` aus. Markieren Sie für jedes Teilproblem die optimale Entscheidung.



Schritt 3: Implementierung

Python-Implementierung im Code-Expert.

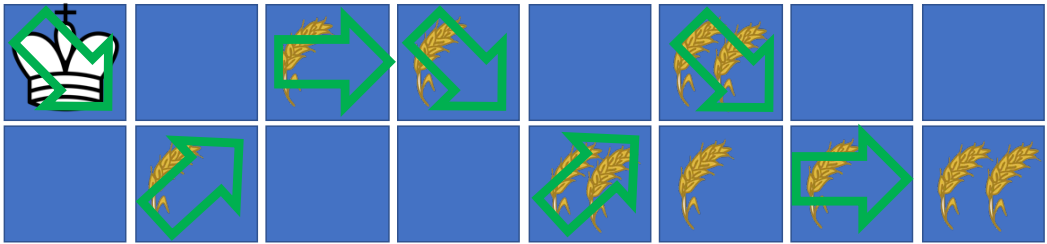
3. Gieriger Königsweg

Problem Beschreibung

Was ist, wenn Sie nur 2 Zeilen haben?

Problem Beschreibung

Brauchen Sie wirklich DP?



Gierige Lösung

Wenn der König sich in Position (i, j) befindet, ...

Gierige Lösung

Strategie: Verwenden Sie eine for-Schleife, um den König zur nächsten Zelle mit dem höchsten Wert zu bewegen.

Was ist die Invariante für diese gierige Lösung?

Implementierung

Python-Implementierung im Code-Expert.

4. Homework

Übung 9: Dynamische Programmierung

On https://expert.ethz.ch/enrolled/SS25/mavt2/exercises

- Mission Mars with lava
- Binomial coefficients
- Triangle
- Longest common substring

Abgabedatum: Montag 28.04.2025, 20:00 MEZ **NO HARDCODING**