

# AIK Programmieren 1 Übung: Backtracking

*Klaus Kusche*

## **Das 8-Damen-Problem**

Das 8-Damen-Problem lautet wie folgt: „Wie stelle ich  $n$  Damen auf ein  $n*n$  großes Schachbrett, sodass keine zwei Damen einander bedrohen, d.h. sodass in jeder Zeile und jeder Spalte genau eine Dame und in jeder Diagonale höchstens eine Dame steht?“

Versuche, ein Programm zu schreiben, das alle Lösungen des 8-Damen-Problems für eine beim Programmstart auf der Befehlszeile anzugebende Brettgröße (bis zu einer Maximalgröße von 30) berechnet und ausgibt.

Im Unterschied zu den Folien empfehle ich, das Brett und seine Größe nicht allen Funktionen als Parameter zu übergeben, sondern als globale Variablen zu deklarieren (das macht das Programm übersichtlicher und schneller):

```
// Seitenlänge des Brettes = Anzahl der zu setzenden Damen
int anzahl;
// brett[zeile] speichert die Nummer der Spalte (0...anzahl-1),
// in der die Dame in der jeweiligen Zeile (0...anzahl-1) gerade steht
// Das Brett wird von oben nach unten gefüllt (Zeile 0, Zeile 1, ...)
int brett[MAX_ANZAHL];
```

Weiters brauchst du 4 Funktionen:

1. **main:**

**main** speichert die Anzahl der Damen von der Befehlszeile in der globalen Variablen **anzahl** (und prüft dabei, ob sie zwischen 1 und 30 liegt) und ruft dann die rekursive Funktion für Zeile 0 auf.

2. Die rekursive Funktion zum Setzen der Damen:

Sie funktioniert genau so wie in der Theoriestunde besprochen und auf den Folien angegeben.

3. Die Funktion ok zum Prüfen einer Position:

Wenn diese Funktion aufgerufen wird, um die Position der Dame in Zeile Nummer **zeile** zu prüfen, dann sind die Damen in den kleineren Zeilen schon richtig gesetzt, und in den größeren Zeilen sitzt noch keine Dame.

Man muss daher die angegebene Position in einer Schleife mit allen gespeicherten Positionen in den kleineren Zeilen vergleichen. Wenn man dabei eine Dame findet, die in derselben Spalte oder derselben Diagonale sitzt, liefert die Funktion **false**, hat die Schleife alle Zeilen ohne „Treffer“ geprüft, endet die Funktion mit **true**.

Ob zwei Damen in derselben Diagonale stehen, prüft man, indem man die Differenz der Zeilen und die Differenz der Spalten berechnet: Ist Zeilendifferenz gleich Spaltendifferenz oder Zeilendifferenz gleich minus Spaltendifferenz, so ist die Diagonale dieselbe.

4. Die Ausgabe-Funktion:

Im einfachsten Fall gibt sie in einer Schleife die Spalten aller Damen aus.

## Erweiterung 1: Grafische Anzeige

Wir wollen die Lösung grafisch anzeigen. Dazu sind 2 Umbauten nötig:

- Wir müssen den Header **sdllinterf.h** inkludieren und in **main** vor der rekursiven Funktion **sdllInit()**; und nach der rekursiven Funktion **sdllExit()**; aufrufen.
- Die Ausgabe-Funktion wird komplizierter:
  - Wir berechnen zuerst einmal die Größe (in Pixel) eines Feldes des Schachbretts und die Koordinaten des Mittelpunktes des linken oberen Feldes:  

```
feldgr = ((SDL_X_SIZE >= SDL_Y_SIZE) ? SDL_Y_SIZE : SDL_X_SIZE) - 9;  
feldgr /= anzahl;  
if (feldgr % 2 == 0) --feldgr; // macht feldgr ungerade  
links = (SDL_X_SIZE - feldgr * (anzahl - 1)) / 2;  
oben = (SDL_Y_SIZE - feldgr * (anzahl - 1)) / 2;
```
  - Dann zeichnen wir in einer Schleife für jede einzelne Dame einen Kreis: **sdllDrawCircle(xMittel, yMittel, xRad, yRad, r, g, b)**;  
Den Mittelpunkt berechnet man, indem man zum Mittelpunkt links oben Zeilen mal bzw. Spalten mal die Feldgröße dazurechnet, und der Radius ist in beide Richtungen die halbe Feldgröße minus drei Pixel Rand.
  - Vor der Schleife löscht man die alte Anzeige (**sdllSetBlack()**);, nach der Schleife aktualisiert man die Anzeige (**sdllUpdate()**); und gibt etwas Zeit zum Anschauen (**sdllMilliSleep(ms)**);).

Tipp: Durch Ändern der **#define**'s von **SDL\_X\_SIZE** und **SDL\_Y\_SIZE** in **sdllinterf.h** kannst du die Größe des Grafikfensters quadratisch machen und an deinen Bildschirm anpassen. Du musst aber nach der Änderung sowohl dein Programm als auch **sdllinterf.c** frisch kompilieren (es gibt einen Button „Alles neu bauen“ im DevCpp).

## Erweiterung 2: Life-Anzeige

Wir wollen nicht nur die fertige Lösung anzeigen, sondern jedes Setzen oder Wegnehmen einer Dame, damit man das Suchen der Lösungen verfolgen kann.

Dazu sind folgende Änderungen notwendig:

- **feldgr**, **links** und **oben** werden globale Variablen und am Beginn von **main** berechnet.
- Die Anzeige wird nie ganz gelöscht: **sdllSetBlack()**; fällt ersatzlos weg.
- Das Zeichnen einer Dame (**sdllDrawCircle(...)** mit **sdllUpdate()**; und **sdllMilliSleep(ms)**; danach) wandert in die rekursive Funktion: Sobald eine Dame einen zulässigen Platz hat (nach if (ok(...))) und vor dem rekursiven Aufruf, wird sie gezeichnet.
- Unmittelbar nach dem rekursiven Aufruf muss die Dame wieder weggelöscht werden: Wir machen genau dieselben drei **sdll**-Aufrufe, aber mit Farbe schwarz für den Kreis.
- Die Anzeige-Funktion hat nichts mehr zu tun (außer mit einem längeren **sdllMilliSleep(ms)**; zu warten), denn die Damen sind ja schon alle gezeichnet.