5.9 Einen Automaten programmieren

Die Wache in einem Schleichspiel patrouilliert, bis sie Sie sieht. Dann verfolgt sie Sie; verliert sie Sie aus den Augen, sucht sie eine Weile und kehrt schließlich zur Patrouille zurück. Spieleentwickler programmieren solches Verhalten sehr oft als endlichen Automaten: eine Variable für den Zustand, eine Tabelle für die Übergänge. Wie wenig Code das ist, zeigt dieser Abschnitt.

> In der Webfassung steht hier ein Bedienfeld zum Ausprobieren: **Zustandsmaschine**.

Die Übergangstabelle lässt sich unmittelbar in Code übersetzen. Das Programm hält nur eine einzige Variable fest: den aktuellen Zustand.

Python

```python
# Automat: akzeptiert alle Woerter ueber {a, b}, die "aa" enthalten
UEBERGAENGE = {
    ("z0", "a"): "z1", ("z0", "b"): "z0",
    ("z1", "a"): "z2", ("z1", "b"): "z0",
    ("z2", "a"): "z2", ("z2", "b"): "z2",
}
START = "z0"
ENDZUSTAENDE = {"z2"}


def akzeptiert(wort):
    zustand = START
    for zeichen in wort:
        if (zustand, zeichen) not in UEBERGAENGE:
            return False                    # Zeichen nicht im Alphabet
        zustand = UEBERGAENGE[(zustand, zeichen)]
    return zustand in ENDZUSTAENDE


for wort in ["abab", "abba", "baab", "aa", ""]:
    print(wort, akzeptiert(wort))
```

Object Pascal

Gymnasium Ottweiler

```
program Automat;

type
  TZustand = (z0, z1, z2);

function Uebergang(zustand: TZustand; zeichen: Char): TZustand;
begin
  Result := zustand;
  case zustand of
    z0: if zeichen = 'a' then Result := z1 else Result := z0;
    z1: if zeichen = 'a' then Result := z2 else Result := z0;
    z2: Result := z2;                        { Endzustand bleibt erhalten }
  end;
end;

function Akzeptiert(const wort: string): Boolean;
var
  zustand: TZustand;
  i: Integer;
begin
  zustand := z0;
  for i := 1 to Length(wort) do
    zustand := Uebergang(zustand, wort[i]);
  Result := zustand = z2;
end;

begin
  WriteLn(Akzeptiert('abab'));               { False }
  WriteLn(Akzeptiert('baab'));               { True  }
end.
```

Beachten Sie den Unterschied im Aufbau: Python hält die Tabelle als Datenstruktur, Pascal als `case`-Verzweigung. Die Datenstruktur ist der bessere Weg — man kann einen anderen Automaten einsetzen, ohne den Code zu ändern.

Arbeitsauftrag 9

36. Erweitern Sie das Python-Programm so, dass es den durchlaufenen Zustandsweg ausgibt.
37. Ersetzen Sie die Tabelle durch die Ihres Automaten für „endet auf abb” und testen Sie.
38. Schreiben Sie die Pascal-Fassung so um, dass sie eine Übergangstabelle als Array `array[TZustand, 'a'..'b'] of TZustand` benutzt statt der `case`-Anweisung.
39. Schreiben Sie ein Programm, das einen Text zeichenweise liest und prüft, ob er ein gültiger Bezeichner ist.
40. **Projekt:** Ein Programm liest eine Übergangstabelle aus einer Textdatei und prüft dann beliebige Wörter. Damit haben Sie einen Automaten-Simulator geschrieben.