

Explizit gegebene Folgen

Folgen sind (unendliche viele) Zahlen $a_0, a_1, a_2, \dots$ (manchmal auch ab Index > 0). Das mathematische Modell ist eine Funktion: $a: \mathbf{N} \rightarrow \mathbf{Z}, \mathbf{R}, \mathbf{C}, \dots$ ($\mathbf{Z}$ = ganze Zahlen, $\mathbf{R}$ = reelle Zahlen, $\mathbf{C}$ = komplexe Zahlen).

Eine explizit gegebene Folge wird durch einen Funktionsterm definiert, mit dem man das n . Element direkt (explizit) berechnen kann, ohne andere Folgenglieder dafür zu benötigen:

$$a_n = n^2 - n + 3$$

Als Funktion (Lambda):

Solche Folgen lassen sich durch eine C++-Funktion mit ganzzahligen oder Gleitkomma-Werten beschreiben, z.B.:

```
double a(int n)
{    return n*n - n + 3; }

double a(int n)
{    double x = n; return x*x - x + 3; } // rechne mit double
```

Die 2. Variante ist hier besser, weil sie keinen Überlauf produzieren kann. Auch werden Divisionen nicht versehentlich mit Ganzzahl-Arithmetik berechnet. Man kann die Umwandlung `int -> double` auch schon bei der Argumentübergabe erledigen (das gefällt mir persönlich nicht so gut!):

```
double a(double x)                // schon hier double
{    return x*x - x + 3; }        // rechne mit double
```

Natürlich kann man auch Lambdas verwenden (besser ohne `auto` beim Argument):

```
auto a = [](double x){    return x*x - x + 3; };
```

Als Container:

Ebenso kann man alle benötigten Werte vorher berechnen und in einem Container abspeichern:

```
std::vector<double> a{3., 3., 7., 9.};
```

Der Zugriff erfolgt dann mit dem Indexoperator: `a[3]`