Der Rückgabewert von Funktionen

Jede Funktion in C++ (außer Konstruktoren und Destruktoren siehe später) **MUSS** einen Rückgabewert-Typ vereinbaren. Gibt sie nichts zurück, muss der Rückgabetyp `void` verwendet werden (dieser entspricht dem Python-Typ `Nonetype`). **Jede** `return`-Anweisung **sollte** einen geeigneten Rückgabewert definieren (außer der Rückgabetyp war `void`). Entspricht er nicht dem Typ der Funktion, **wird er in diesen umgewandelt** (bzw. es ist ein Fehler, wenn das nicht möglich ist!).

```
double f()
{
     return 1.0;
}

double f()
{
     return 1;        // Achtung, es wird 1.0 zurückgegeben
}
```

`constexpr` Funktionen

Eine als `constexpr` definierte Funktion darf (soll) der Compiler am besten schon bei der Kompilation aufrufen, wenn die Argumente Konstanten sind. Es ist dies nur eine Empfehlung für den Compiler, er muss es nicht machen (obwohl die neueren das garantiert machen):

```
constexpr int factorial(int n)
{    return n <= 1 ?  1 : n * factorial(n-1);  }

auto x{factorial(5)};      // darf ersetzt werden durch: auto x{120};
```

Funktionsattribute

Seit C++17 gibt es in C++ Funktionsattribute, mit denen man dem Compiler Informationen über die Funktion geben kann. Beispiele sind:

`[[noreturn]]` diese Funktion kehrt nicht zurück (sie beendet das Programm).
`[[nodiscard]]` der Aufrufer **muss die Rückgabe** verwenden.

```
[[nodiscard]] bool do_something() {…}

do_something();              // Fehler: Der Rückgabewert wurde nicht beachtet
```

```
if (do_something())          // ok: Der Rückgabewert wurde verwendet
```

`auto` als Rückgabetyp

Die Typangabe `auto` **als Rückgabetyp ist seit C++11 in Funktionsdefinitionen erlaubt**. Allerdings musste man damals den Rückgabewert-Typ trotzdem explizit angeben, nur eben an einer anderen Stelle (trailing return type). Seit C++14 ist die Typangabe `auto` auch ohne explizite Typangabe in **Funktionsdefinitionen** (nicht in Deklarationen) erlaubt, C++ ermittelt den Typ aus der (den) `return`-Anweisung(en). Damit das funktioniert, müssen alle `return`-Anweisungen denselben Typ zurückgeben!

```
auto f() -> double  // auto in C++11 ist nur in dieser Form erlaubt
                    // der tatsächliche Rückgabetyp steht erst nach dem ->
                    // f() ist also vom Typ double
{
    return 1;       // es wird 1.0 zurückgegeben
}

auto f()            // seit C++14 in dieser Form erlaubt
{
    return 1;       // f() ist dadurch vom Typ int, es wird 1 zurückgegeben
}

auto f();           // Fehler: der Typ kann in einer Deklaration nicht erkannt werden
```

`void` als Rückgabetyp

Eine Funktion mit Rückgabetyp `void` gibt nichts zurück. Keine `return`-Anweisung darf hier einen Rückgabewert vereinbaren! Der Datentyp `void` wurde genau für diese Anwendung erfunden. Insbesondere kann man keine Instanzen mit diesem Typ anlegen:

```
void x;             // Fehler in C++! In Python gibt es die konstante Instanz None
```