```
int Funktion();
int Funktion(double);          // ok.
int Funktion(double,char);     // ok.
char Funktion(double,char);    // Fehler, da nur Rückgabewert
                               // unterschiedlich
```

Überladen von Operatoren

- Operatoren in C++ sind als Funktionen mit fester Anzahl von Argumenten definiert.
  - `a + b` ≡ `operator+(a,b)`
  - Operatoren verhalten sich wie Funktionen und können überladen werden, sofern deren Argumente *keine* elementaren Typen, sondern *Klassen* sind.

Binäre Operatoren, Operatoren mit genau *zwei* Argumenten.

- +, -, *, /, aber auch <<, >>
  - Die Anzahl der Argumente eines Operators kann *nicht* verändert werden. Es können auch keine neuen Operatoren definiert, sondern nur bereits existierende überladen werden.

Funktionsaufruruf-Operator `()`, einziger Operator mit nicht vorgegebener Argumentenzahl (wird bei Definition bestimmt). Ermöglicht die Verwendung einer Klasseninstanz in Analogie zu einer Funktion.

```
class Klasse
{
   float operator()(...Argumente...)  ... ;
}
main()
{
   Klasse Instanz;
   float y = Instanz(1,2,3,...);
}
```

Klassen

Klasse, Datenobjekt in C++. Besteht aus Datenstrukturen (**Member**, Variablen, Felder, `struct`'s) und Methoden (**Member-Funktionen**) zur Manipulation der Daten.

- Klassen bilden das Rückgrat eines C++-Programms.
- Member und Member-Funktionen können **private** (nur lokal von Member-Funktionen ansprechbar), **protected** (auch von Member-Funktionen von abgeleiteten Klassen ansprechbar) oder **public** (global im gesamten Programm aufrufbar) deklariert werden.
  - Die Schlüsselworte `private`, `protected` und `public` können beliebig oft in beliebiger Reihenfolge verwendet werden.

```
class Beispiel              // Klassenname ist Beispiel
{
   private:
   int i,j;                 // Private Integer-Variablen
   public:
   void Setzen(int,int);    // Funktion setzt i und j mit
                            // den zu übergebenden Werten.
   int Quotient();          // Funktion dividiert i durch j
}
```