Die Wert-Übergabe (by copy, by value)

```
int f(int x)          // x wird als Wert (Kopie) des Aufrufer-Arguments erstellt
{
     return 2*x + 1;
}
```

Bei der Wertübergabe steht beim Parameter nur der Typ und der Bezeichner ohne Ampersand `&`. Wird vom Aufrufer eine **Variable** per Wert übergeben, **besitzt diese nach dem Funktionsaufruf immer noch den alten Wert**, **sie kann also durch den Funktionsaufruf nie verändert werden** (die Funktion erhält in Wahrheit nur eine Kopie des Originalarguments)**:**

```
void f(int x)
{
     x = 5;           // ändert den Wert  von  x  (der Kopie!)
}

...
x = 1;
f(x);
cout << x;            // druckt  1 (NICHT 5).
```

Was darf man per Wert übergeben (am obigen Beispiel : `int f(int);`)

Konstante:	`f(1);`
Variable	`int i;  f(i);`
umwandelbare Variable	`char i;  f(i);`
konstante Variable	`const int i; f(i);`
Rechenergebnisse	`f(i+3); f(i++);`

Was darf man nicht per Wert übergeben:

- nicht umwandelbare Ausdrücke `string i;  f(i);`
- Objektinstanzen, die nicht kopiert werden können (z.B. `cin, cout`)