

Referenzen

Java:

```
class Stud ...
{
    Stud s2;
    {
        Stud s1 = new Stud();
        s2 = s1;
    } // ~ s1
} // ~ s2
```

s1 ist eine Referenz auf Stud

Stud

```
Stud.vn = ""
Stud.nn = ""
Stud.m = 0
```

C++:

```
Stud s1; // neue Variable = Instanz
Stud &s2 = s1; //Referenz      Stud *p_s2 = &s1; ← Zeiger
s1.print();                    s1.print();
s2.print();                    (*p_s2).print();
                                p_s2->print();

s2 = s1; // !                  p_s2 = &s; // setzt Zeiger neu
s1 = s; // kopiert
```

Eine Referenz wird einmal gesetzt, ein Zeiger kann auch im weiteren Verlauf umgesetzt werden.

<pre>{ Stud &s2; { Stud s1; s2 = s1; } s2.print(); }</pre>	<pre>{ Stud * p_s2 = NULL; { Stud s1; p_s2 = &s1; ... } // hier endet die Lebensdauer von s1, deshalb p_s2->printf(); }</pre>
---	--

<pre>void up (int &i) { i++; } ... { int wert = 5 up (wert); //call by value printf("%d\n", wert); }</pre>	<pre>void up (int *p_i) { (* p_i)++; } ... { int wert = 5; up (&wert); printf("%d\n", wert); }</pre>
--	--

↳ 5
↳ 6

↳ 6

```

class Stud {...}

void zeigeStud (Stud &s) {...}

void zeigeStud (const Stud &s)
{
    printf(...);
}
...
const Stud s1("...", ...);
zeigeStud(s1); // call by reference
Stud s2(...);
zeigeStud(s2);

```

```

template <typename T>
void tausche (int &i, int &j);
{
    T tmp = i;
    i=j; j=tmp;
}
...
{
    int a=10;
    int b=15;
    tausche (a, b);

    double x=3.1;
    double y=4.7;
    tausche (x, y);
}

```

} nur mit Template möglich

```

#include <string>
class Stud
{
public:
    Stud (const std::String &vn)
    :vn(vn)
    {
    }
private:
    std::string &vn;
    std::string nn;
}
...
{
    std::string vn="Otto"; // oder std::string vn("Otto");
    Stud s(vn);
    vn="Karl"; /ändert nicht doch s.vn
}

```

Default-Constructor	Parameterloser Constructor
	Constructor mit Parameter
	Copy-Constructor

```

Stud s1; // default, o.) parameterlos
Stud s2("Otto", "Waalkes", 37); // Constructor mit Parameter
Stud s3(); // Deklaration einer Funktion, die
            // - keine Parameter
            // - s3 heißt
            // - Stud liefert
Stud s4(s1); // Copy-Constructor
Stud s5=s1; // Copy-Constructor

s4 = s1; // Zuweisung
s5 = s1; // Zuweisung

```