

Funktionen

Funktion	Selbstständiger Codeabschnitt
<p>Wichtige Befehle:</p> <p>Definition: <code>int my_fun (bool arg1, float arg2) {...}</code> Rückgabe: <code>return my_val;</code> Aufruf: <code>my_fun(true, 3.75f)</code></p> <p>Der Rückgabewert wird immer zum Rückgabetyp konvertiert.</p> <p>Jede Funktion, die nicht den Rückgabetyp <code>void</code> hat, muss ein return haben.</p>	
<pre>unsigned int bin_digits (unsigned int n) { if (n == 0) return 1; // stops function and returns 1 unsigned int count = 0; do { n /= 2; ++count; } while (n > 0); return count; } int main () { std::cout << bin_digits(3) << "\n"; // Output: 2 std::cout << bin_digits(8) << "\n"; // Output: 4 return 0; }</pre>	

// PRE: ... // POST: ...	Funktionsbeschreibung
<p>PRE-/POST-Conditions gehören vor jede Funktionsdefinition ausser der main-Funktion. (In diesen Programmier-Befehlszusammenfassungen werden sie aber manchmal aus Platzgründen weggelassen.)</p> <p>Man kann beispielsweise assert verwenden, um das Programm abubrechen, falls die Funktion doch mal mit Argumenten aufgerufen wird, welche die PRE-Condition verletzen.</p>	

(...)

Programmier-Befehle - Woche 5

(...)

```
// POST: return value is a^4
int power_4 (unsigned int a) {
    return a*a*a*a;
}

// PRE: width >= 0 and height >= 0
// POST: returns the rectangle area given by width and height
double area (double width, double height) {
    assert(width >= 0 && height >= 0);
    return width * height;
}
```

Datentypen

<code>void</code>	Datentyp für Funktion ohne Rückgabe .
void-Funktionen haben keinen Rückgabewert, aber sinnvollerweise einen Effekt (z.B. Textausgabe im Beispiel unten).	
<pre>void print_account (double assets, double interest) { std::cout << "Your assets: " << assets << "\n" << "Your interest: " << interest << "\n"; }</pre>	