

## Generell

<code>namespace</code>	Katalogisierung von Befehlen
<p>Mit <code>namespaces</code> kann man Funktionen, Typen, etc. katalogisieren (z.B. bezüglich Projekten). Beispielsweise können Funktionen im <code>namespace ifee</code> zusammengefasst werden. Damit kann u.A. Problemen mit Namenskonflikten (z.B. unterschiedliche Funktionen mit identischem Namen) bei grösseren Projekten vorgebeugt werden.</p>	
<pre>namespace ifee { // namespace called ifee      // POST: "Hi" was written to the terminal     void output_func () { // this function is in namespace ifee         std::cout &lt;&lt; "Hi";     } }  int main () {     ifee::output_func(); // use output_func from namespace ifee     return 0; }</pre>	

## Standardbibliothek

<code>std::min</code>	Minimum zweier Argumente
<p>Erfordert: <code>#include &lt;algorithm&gt;</code></p> <p>Sonst gibt es noch: <code>std::max</code>     <b>Maximum</b> zweier Argumente</p> <p>Wichtig ist, dass beide Argumente <b>vom selben Typ</b> sind. Sonst geht es <b>nicht</b>.</p>	
<pre>double z; std::cin &gt;&gt; z; std::cout &lt;&lt; std::min(z, 1.0); // min of z and 1  std::cout &lt;&lt; std::min(z, 1); // Error: z is double, 1 is int</pre>	

## Datentypen

Referenzen	Alias für bestehende Variable
<p>Referenzen können <b>nur Variablen</b> ihres zugrundeliegenden <b>Typs</b> referenzieren. Sonst gibt es einen Fehler.</p> <p>Ausserdem können Referenzen <b>nur mit l-Werten initialisiert</b> werden (also Werten mit einer Adresse im Speicher).</p>	
<pre data-bbox="335 772 1287 1198">// Usage int a = 3; int&amp; b = a; // reference to a std::cout &lt;&lt; b &lt;&lt; "\n"; // Output: 3 a = 18; std::cout &lt;&lt; b &lt;&lt; "\n"; // Output: 18 b = 25; std::cout &lt;&lt; a &lt;&lt; "\n"; // Output: 25  // Issues int&amp; c = 3; // Error: 3 has no address in memory bool d = false; int&amp; e = d; // Error: d is bool, e wants to reference an int</pre>	