

Programmier-Befehle - Woche 2

Datentypen

<code>unsigned int</code>	Datentyp für natürliche Zahlen (inklusive 0)
Literal: <code>...u</code>	
<pre>unsigned int a = 4; // Conversion int --> unsigned int unsigned int b = 4u; // No conversion std::cout << a - 5 << "\n"; // too small (underflow)</pre>	

Operatoren

<code>/</code>	Division
Präzedenz: 14 und Assoziativität: links	
Falls ints oder unsigned ints dividiert werden, so rundet der Operator automatisch zu 0 hin .	
<pre>unsigned int a = 9 / 3; // Result: 3 unsigned int b = 5 / 3; // Result: 1 int c = -3 / 2; // Result: -1</pre>	

<code>%</code>	Modulo . Rest der <i>Ganzzahl</i> division
Präzedenz: 14 und Assoziativität: links	
% gibt es <i>nur</i> für int und unsigned int. Bei negativen Zahlen übernimmt % das Vorzeichen des linken Operanden.	
<pre>int a = 5; int division = a / 3; // Result: 1 int rest = a % 3; // Result: 2 int negative = -5 % -3; // Result: -2</pre>	

Programmier-Befehle - Woche 2

<code>++...</code>	Prä-Inkrement. Erhöht den Wert der Variablen und gibt den <i>neuen</i> Wert zurück.
Präzedenz: 16 und Assoziativität: <i>rechts</i>	
Sonst gibt es noch: <code>--...</code> Prä-Dekrement	
<pre>int a = 0; int b = ++a; // b gets value 1, // a gets value 1</pre>	

<code>...++</code>	Post-Inkrement. Erhöht den Wert der Variablen und gibt den <i>alten</i> Wert zurück.
Präzedenz: 17 und Assoziativität: <i>links</i>	
Sonst gibt es noch: <code>...--</code> Post-Dekrement	
<pre>int a = 0; int b = a++; // b gets value 0, // a gets value 1</pre>	

<code>+=</code>	Addiert den rechten Operanden zum linken Operanden.
Präzedenz: 4 und Assoziativität: <i>rechts</i>	
Sonst gibt es noch: <code>--...</code> für Subtraktion <code>*=...</code> für Multiplikation <code>/=...</code> für Division <code>%=...</code> für Modulo	
<pre>int a = 4; a += 5; // a gets value 9</pre>	

Generell

<code>std::numeric_limits<T>::min()</code>	Ermittelt kleinsten zulässigen Wert des Datentyps <code>T</code> .
Erfordert: <code>#include<limits></code>	
Sonst gibt es noch: <code>std::numeric_limits<T>::max()</code>	
<pre>int lower_bdd = std::numeric_limits<int>::min(); std::cout << "Enter a number larger than " << lower_bdd << ": "; int input; std::cin >> input; // User knows the smallest valid number.</pre>	