

Die `break` und `continue` Anweisungen

Diese 2 Anweisungen zählt man zu den erweiterten Sprunganweisungen. **Sie sind nur in Schleifen erlaubt** (`break` zusätzlich noch in `switch case` Blöcken).

`break`: Bricht die innerste (bei mehreren ineinander geschachtelten Schleifen) Schleife ab und verzweigt hinter diese.

`continue`: Bricht den aktuellen Durchlauf der innersten (bei mehreren ineinander geschachtelten Schleifen) Schleife ab und beginnt einen neuen Durchlauf. Hierbei wird nicht die Schleife selbst beendet sondern nur der aktuelle Durchlauf durch diese. Bei einer `while` oder `do while` Schleife springt `continue` auf den Test der *Bedingung*, bei einer `for` Schleife auf *Update* gefolgt von der *Bedingung*.

Beispiel: Wir fordern ganze Zahlen an (bis 0 kommt) und untersuchen nur die geraden positiven

```

// diese Variablen brauchen wir auch nach der Schleife!
int  gelesen{}, positiv{}, gerade{};
for (int x;;)      // x wird nur innerhalb der Schleife gebraucht; Endlosschleife
{
    cout << "int Nr. " << gelesen+1 << "(Ende bei 0): ";
    if (!(cin >> x) || x == 0)
        break;    // Keine Zahl gelesen oder 0 wurde eingegeben: fertig
    ++gelesen;    // wir registrieren die Eingabe immer
    if (x < 0)    // negative Zahlen ignorieren wir
        continue; // lies die nächste Zahl ein
    ++positiv;    // hier muss die Zahl sicher positiv sein
    gerade += (x % 2 == 0); // +1, falls gerade!
}

```

Hier haben wir z.B. keine *Bedingung* im `for`, also handelt es sich im Prinzip um eine Endlosschleife. Wir haben aber innerhalb der Schleife ein ordnungsgemäßes Ende mittels `if` und `break` programmiert. Mit der `continue`-Anweisung halten wir die Anzahl der Einrückungen klein, weil wir sofort zur Eingabe der nächsten Zahl verzweigen, wenn wir die Bearbeitung der aktuellen abgeschlossen haben (keine verschachtelten `if else` Konstruktionen).

Die `break` Anweisung beendet nur die innerste Schleife. Will man gleich mehrere verschachtelte Schleifen auf einmal beenden, verwendet man am besten eine `goto`-Anweisung (selbst wenn diese einen schlechten Ruf besitzt). Auch ein `return` (falls möglich) beendet sämtliche Schleifen auf einmal und verlässt die ganze Funktion.