```
================================================================================
Indentation (Sourcecodeformatierung) mit vim
================================================================================

Vim unterstützt automatische Einrückung. Dafür empfehlen sich folgende
Einstellungen (einfach in die Datei~/.vimrc einsetzen):

set sts=4 ts=8 sw=4

Im Einzelnen

ts=8        stellt die Breite eines Tabs ein (nichts anderes als 8 einstellen)
sts=4       stellt ein, wieviele Spaces vim automatisch vorrückt, wenn man auf
            die Tabtaste drückt
sw=4        stellt für vims eingebauten Indentationalgorithmus ein, wieviele
            innerhalb eines Blocks (also { ... }) eingerückt werden soll

Um Teile des Sourcecodes umzuindenten empfiehlt sich

v           fängt mit einem neuen visuellen Block an
bewegen     bis zum Ende des Blocks der neu eingerückt werden soll
=           eigentliches Einrücken

Insbesondere praktisch ist es, wenn man auf einer öffnenden oder schliessenden
geschweiften Klammer steht ("{" oder "}"), die Taste % zu verwenden, um zur
passenden Klammer zu gehen. Die gesamte Einrückung kann dann mit 3 Tasten
vorgenommen werden:

v%=         rückt Block zwischen passenden Klammern neu ein

Diese spezielle Operation geht sogar noch kürzer mit

=%          rückt Block zwischen passenden Klammern neu ein

Man kann vims Einrückung sehr viel genauer abstimmen, es gibt viele andere
Optionen.

:help C-Indenting

Zwei weitere Befehle sind nach-rechts Einrücken und nach-links Einrücken:

v           fängt mit einem neuen visuellen Block an
bewegen     bis zum Ende des Blocks der neu eingerückt werden soll
< >         eigentliches Einrücken

die sich wieder analog verkürzen lassen zu

v%>         erhöht Indentationlevel zwischen passenden Klammern
v%<         erniedrigt Indentationlevel zwischen passenden Klammern
>%          erhöht Indentationlevel zwischen passenden Klammern
<%          erniedrigt Indentationlevel zwischen passenden Klammern
```



```
================================================================================
Divide and Conquer Algorithmen
================================================================================
Divide and Conquer Algorithmen sind solche, die das Problem in einfachere
Teilprobleme zerlegen, und diese Teilprobleme isoliert lösen, und dann die
Lösungen zusammfügen. Es ergibt sich also das Schema für einen Schritt:

              Gesamtproblem
                    |
                    V
                 Zerlegen

           /        |        \
                    V
  Teilproblem  Teilproblem  Teilproblem
      |             |            |
      V             V            V
  Teillösung   Teillösung    Teillösung

           \        |        /
                    V
              Zusammenfügen
                    |
                    V
               Gesamtlösung

Dies lohnt sich natürlich nur, wenn das Zerlegen, das Lösen der Teilprobleme
und das Zusammenfügen der Lösungen insgesamt weniger Aufwand bedeutet als das
Lösen des Gesamtproblems.

Oft lässt sich der Zerlegungsschritt auch auf die einzelnen Teilprobleme wieder
anwenden, sodass man das Problem solange immer weiter zerlegt, bis die Lösung
sehr einfach wird oder sich das Problem nicht weiter zerlegen lässt.
```

```
================================================================================
Divide and Conquer für die Berechnung von Basis hoch Exponent
================================================================================

Wie lässt sich die Formel Basis hoch Exponent in einfachere Teile zerlegen?

|| Betrachten wir nun zunächst den Fall von Exponent = 6 (gerade):

    ergebnis = b * b * b * b * b * b            (5 Multiplikationen)
               \-------/   \-------/
                   x           x

Das Problem laesst sich also zerlegen in:

    x = b * b * b                               (2 Multiplikationen)
    ergebnis = x * x                            (1 Multiplikation)

Wir sparen damit 2 Multiplikationen.

|| Nun der Fall Exponent = 7 (ungerade):

    ergebnis = b * b * b * b * b * b * b
               \-------/   \-------/
                   x           x

Das Problem laesst sich also zerlegen in:

    x = b * b * b                               (2 Multiplikationen)
    ergebnis = x * x * b                        (2 Multiplikation)

|| Schliesslich der Fall Exponent = 8 (gerade):

    ergebnis = b * b * b * b * b * b * b * b    (7 Multiplikationen)
               \-----------/   \-----------/
                     x             x

Das Problem laesst sich wieder zerlegen in:

    x = b * b * b * b                           (3 Multiplikationen)
    ergebnis = x * x                            (1 Multiplikation)

Hier kann man aber x noch weiter zerlegen (es handelt sich ja auch um
die Berechnung von einer hoch-funktionen, mit geradem Exponenten,
und solche lassen sich wie vorhin gesehen zerlegen):

    y = b * b                                   (1 Multiplikation)
    x = y * y                                   (1 Multiplikation)
    ergebnis = x * x                            (1 Multiplikation)
```