Quellcode – Sourcecode
am Beispiel

Beispiel eines maschinenlesbaren Programms

Nachfolgend sehen Sie einen Programm-Teil, welcher beim Aufruf aus einer Inventar-Liste die einzelnen Inventarobjekte abruft, deren Abschreibungswürdigkeit prüft, gemäß der gesetzlichen Regelung die Abschreibungsperiode ermittelt und dann den jährlichen Abschreibungswert (wenn dieser größer Null ist) an die Liste der jährlichen Abschreibungen übergibt.

```
~
H‰ÌVMo_E_Õzg–]‘¶ó_Æ †X‰C
àd_Ddƒ Œ¥ ü{^õìšX¾p_eGÚ_ÕtW½ªzõ
z®_Óoä¸Š—Ä!5_’ FÜ[¤‡ >%¥?  q?
Òïîì ›_" ýzí®œ-…_®VÕç”+ Ÿ·ßÁß_~ŸÁ
åûËÕx 4[Ž³ ããai_
îM«‡ùøÁìd_d1 |²ZŽO†n__õà«_÷®Ø§É
_v_ Gýí À_ n"¯Ÿö—ËÕ€p} ÙÇÇÃ|…¬–
~
```

Beispiel funktionierende Programmquelle

Nachfolgend sehen Sie den gleichen Programmteil in der Quellsprache „LISP“ des Programmierers.
Hier muss sich ein Dritter intensiv einlesen und lange suchen, um den Teil zu finden, der geändert werden muss – wenn er überhaupt gefunden wird.

```
~
(defun afa-liniar (obj) (value) (time))
(setq obj (car obj#)) (declare (integer
(setq value (lamda ( ) x)))) (set-pprint-
dispatch `(cons ( obj value)) #`(lamda
(obj list) ( if (and (consp (cdr list)) ( null
cddr list))) (funcall (afa `~obj) time
(cadr list)) (pprint-fill time list)))))
(set-pprint-dispatch `(cons (inventar
value)
(pprint-dispatch `(let) nil))
~
```

Beispiel „Best Practice“ Programmierung

Und nun noch einmal der gleiche Quellcode, welcher in diesem Fall aber kommentiert und strukturiert ist.

```
~
;;; Modul afa-line zur Ermittlung der jährlichen linearen
;;; Abschreibung. Genutzte Variablen:
;;;         obj (Abschreibungsobjekt aus Inventarliste Objekte)
;;;         time (gesetzlich erlaubter Abschreibungszeitraum)
;;;         value (Abschreibungswert – zur Übergabe an Liste)
;;; Änderungen zum Abschreibungszeitraum sind im Modul
;;; law-time zu aktualisieren
(defun afa-liniar (obj) (value) (time))
        (setq obj (car obj#))
        (declare (integer (setq value (lamda ( ) x))))
                (set-pprint-dispatch `(cons ( obj value))
                #`(lamda (obj list) ( if (and (consp (cdr list))
                ( null cddr list)))
        (funcall (afa `~obj) time(cadr list)) (pprint-fill time list)))))
        (set-pprint-dispatch `(cons (inventar value)
(pprint-dispatch `(let) nil))
;;; Sollte der Abschreibungswert 0 sein (nil) wird der Wert
;;; ignoriert, andernfalls wird er an das Modul inventar
;;; übergeben und dort weiter verarbeitet
~
```