Einführung in die Programmierung für Physiker

Andere Programmiersprachen

Marc Wagner

Institut für theoretische Physik
Johann Wolfgang Goethe-Universität Frankfurt am Main

WS 2017/18

Liste von Programmiersprachen

- Es existieren hunderte von Programmiersprachen.

Fortran

- **Fortran** ist eine imperative Programmiersprache, die sich insbesondere für numerische Berechnungen eignet; sie ist damit im wissenschaftlichen Bereich ebenfalls recht verbreitet.
- Starke Ähnlichkeiten zu **C**.
- In ihrer neuesten Version ist auch objektorientierte Programmierung möglich.

```
 1.         PROGRAM factorial
 2.         IMPLICIT NONE
 3.
 4.         INTEGER i1, n, result
 5.
 6. C   Berechne n! .
 7.         n = 4
 8.
 9.         result = 1
10.
11.         DO i1 = 1, n
12.           result = result * i1
13.         END DO
14.
15. C   "6" entspricht Ausgabe am Bildschirm, "'(I10)'" entspricht Ausgabe eines Integers mit 10 Zeichen.
16.         WRITE(6, '(I10)') result
17.
18.         END
```

```
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 4
-rw-rw-r-- 1 mwagner mwagner 365 Jan 23 12:44 factorial.f
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ gfortran -o factorial factorial.f
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ls -l
insgesamt 16
-rwxrwxr-x 1 mwagner mwagner 9077 Jan 23 12:45 factorial
-rw-rw-r-- 1 mwagner mwagner  365 Jan 23 12:44 factorial.f
mwagner@laptop-tigger:~/lecture_ProgPhys/slides/tmp$ ./factorial
        24
```

Lisp

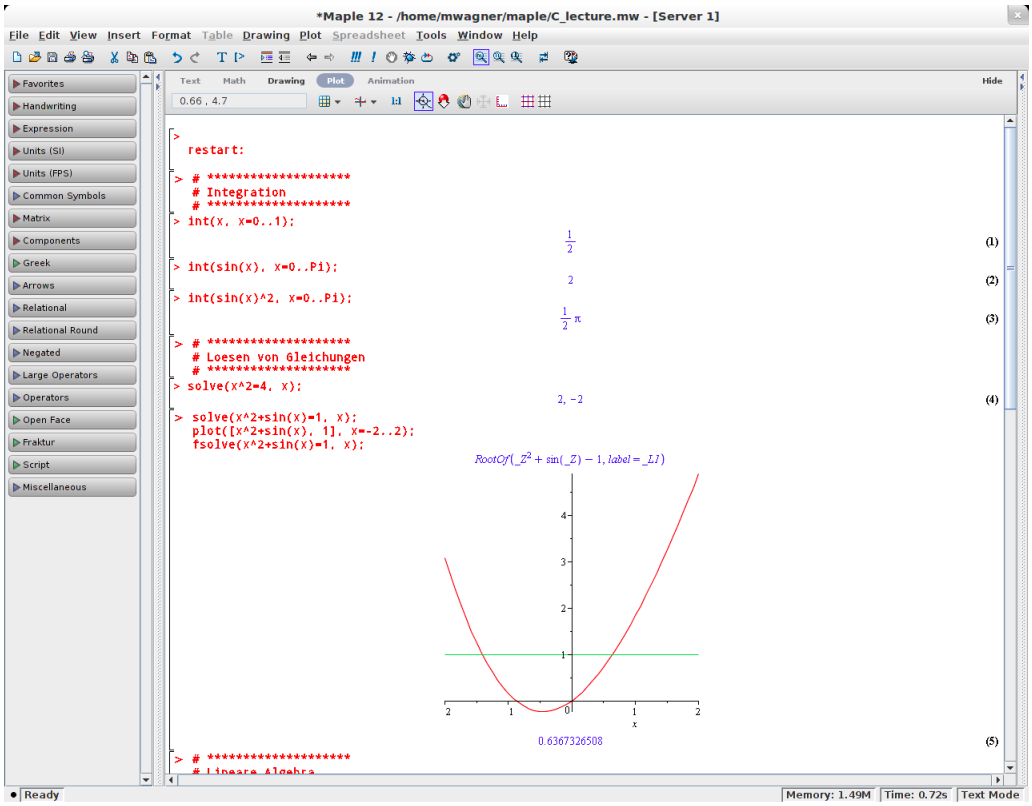
- **Lisp** ist eine Familie von **funktionalen Programmiersprachen** (d.h. Programme bestehen ausschließlich aus Funktionen); eines ihr Haupteinsatzgebiete ist die künstliche Intelligenz; in der Numerik spielt **Lisp** kaum eine Rolle.
- Geringe Ähnlichkeit zu **C**.

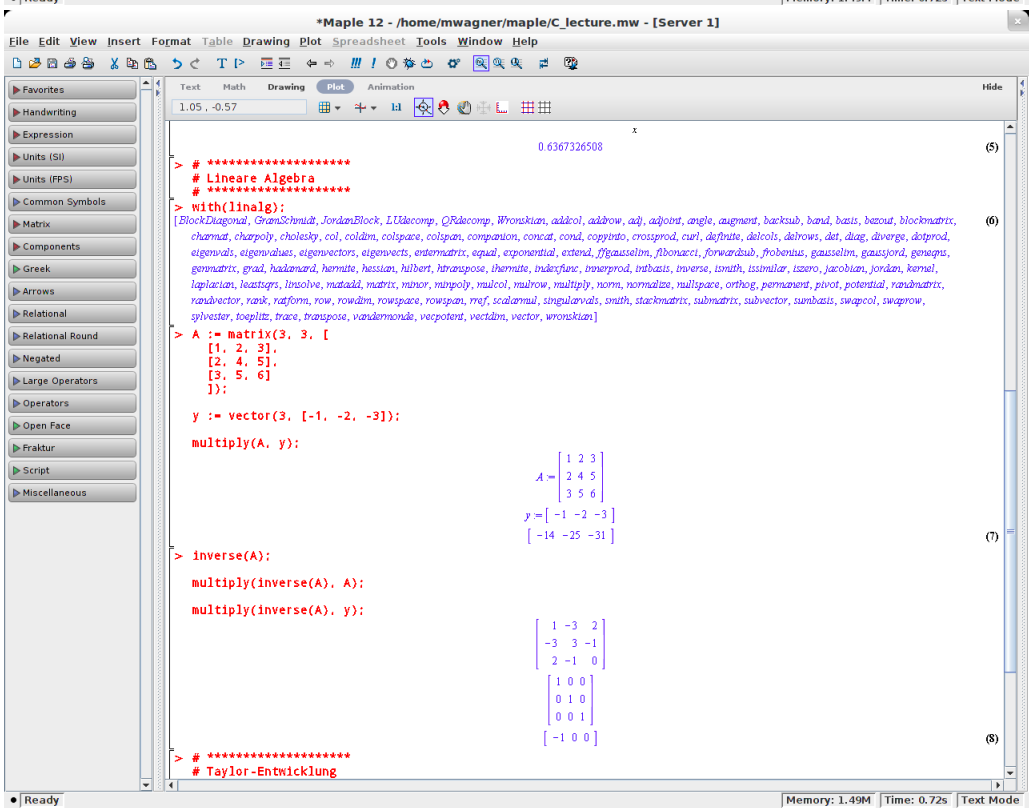
```
1. (defun factorial (n)
2.   (if (> n 1)
3.     (* n (factorial (- n 1)))
4.     1))
```

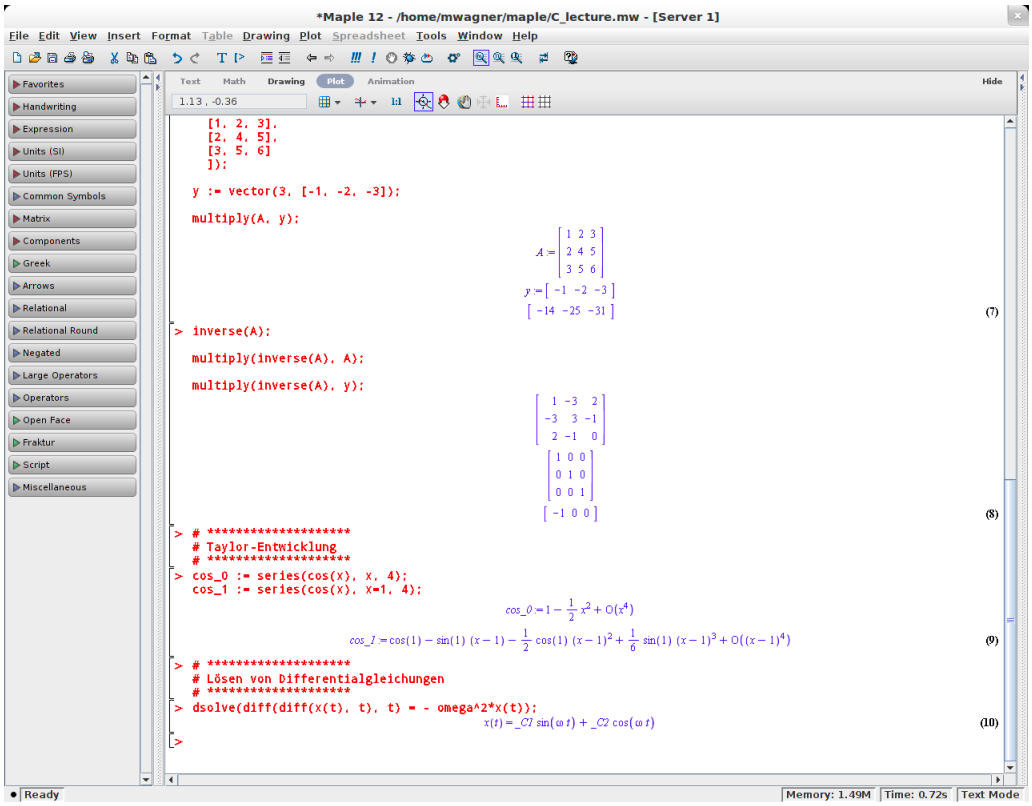
```
mwagner@teefax:~$ sbcl
This is SBCL 1.0.55.0.debian, an implementation of ANSI Common Lisp.
More information about SBCL is available at .

SBCL is free software, provided as is, with absolutely no warranty.
It is mostly in the public domain; some portions are provided under
BSD-style licenses.  See the CREDITS and COPYING files in the
distribution for more information.
* (load "factorial.lsp")

T
* (factorial 4)

24
* (quit)
mwagner@teefax:~$
```

Maple

- **Maple** ist ein Computeralgebrasystem, das vor allem als Unterstützung bei aufwändigen analytischen Rechnungen hilfreich ist.






*Maple 12 - /home/mwagner/maple/C_lecture.mw - [Server 1]
File Edit View Insert Format Table Drawing Plot Spreadsheet Tools Window Help
Favorites
Handwriting
Expression
Units (SI)
Units (FPS)
Common Symbols
Matrix
Components
Greek
Arrows
Relational
Relational Round
Negated
Large Operators
Operators
Open Face
Fraktur
Script
Miscellaneous
Text Math Drawing Plot Animation Hide
1.13 , -0.36
[1, 2, 3],
[2, 4, 5],
[3, 5, 6]
]);
y := vector(3, [-1, -2, -3]);
multiply(A, y);
A := [1 2 3; 2 4 5; 3 5 6]
y := [-1 -2 -3]
[-14 -25 -31] (7)
> inverse(A);
multiply(inverse(A), A);
multiply(inverse(A), y);
[1 -3 2; -3 3 -1; 2 -1 0]
[1 0 0; 0 1 0; 0 0 1]
[-1 0 0] (8)
> # ********************
Taylor-Entwicklung
****
> cos_0 := series(cos(x), x, 4);
cos_1 := series(cos(x), x=1, 4);
cos_0 := 1 - 1/2 x^2 + O(x^4)
cos_1 := cos(1) - sin(1) (x - 1) - 1/2 cos(1) (x - 1)^2 + 1/6 sin(1) (x - 1)^3 + O((x - 1)^4) (9)
> # ********************
Lösen von Differentialgleichungen
****
> dsolve(diff(diff(x(t), t), t) = - omega^2*x(t));
x(t) = _C1 sin(ω t) + _C2 cos(ω t) (10)
>
Ready
Memory: 1.49M Time: 0.72s Text Mode