

Vergleich einiger Programmiersprachen unter Linux

Johann Mitlöhner

10. Januar 2007

Bash, Perl, Sed, und Awk sind auf praktisch allen Unix-System verfügbar. Einen C-Compiler gibt es meist auch, außer es wurden bei der Installation die Development-Komponenten ausgelassen.

Python ist ebenfalls fast immer installiert, nicht unbedingt aber alle libraries.

Java ist meist installiert, jedoch gibt es unterschiedliche JREs (Java Runtime Environment) mit unterschiedlichen Eigenschaften, insb. bezüglich Performance.

1 Bash

Bourne again shell. Die interaktive Shell `/bin/bash` ist vor allem für wiederholte Arbeitsabläufe geeignet. Für numerische Aufgaben ist die bash eher nicht gedacht. Details siehe frühere Einheit.

2 Sed

Streaming editor. Besonders interessant in Zusammenhang mit Editor-Makros.

```
sed '1,$s//\&ouml;/g' < infile
```

3 Awk

Benannt nach Autoren Aho, Weinberger, Kernighan. Für kurze Einzeiler gut, durch BEGIN und END Block.

```
awk '{ s += $2 } END { print s }' < infile
```

4 Perl

Practical Extraction and Reporting Language. Für umfangreiche Aufgaben und auch als eigenständige allgemeine Entwicklungssprache geeignet. Syntax verleitet zu schwer verständlichem Programmierstil.

```
#!/usr/bin/perl

# print column averages for combinations of 1st and 2nd col
# 1st col: a, 2nd col: i, rest: values to be averaged
```

```

while (<>) {
  if (/^(\\d\\S*) (\\S+) (\\S+)/) {
    # line with values
    @w = split /\s+/;
    $a = sprintf("%06d",$w[0]);
    $i = sprintf("%06d",$w[1]);
    $ai{"$a,$i"} = 1;
    for ($j = 2; $j < @w; $j++) {
      $sum{"$a,$i,$j"} += $w[$j];
      $cnt{"$a,$i,$j"}++;
    }
  } elsif ($rec++ == 0) {
    # header line
    print;
  }
}
foreach $k (sort keys %ai) {
  ($a,$i) = split /,/ , $k;
  print "$a $i";
  for ($j = 2; $j < @w; $j++) {
    $avg = $sum{"$k,$j"} / $cnt{"$k,$j"};
    print " ", sprintf("%.4f", $avg);
  }
  print "\n";
}

```

5 Python

Besticht durch klare Syntax, daher auch “executable pseudo-code” genannt. Besonderheit: Blockstruktur wird durch Einrücken markiert. Interpreter unterstützt interaktive Entwicklung.

```

# factorial
def fact(n):
    f=1
    while n>1:
        f=f*n
        n=n-1
    return f

```

6 C

ISO Standard, Low-level, aber trotzdem maschinenunabhängig und beste Performance (außer Assembler). Vorsicht mit Pointern. Der C-Compiler (cc oder gcc) erzeugt native code, der ohne Interpreter auf jedem Rechner mit passender Architektur läuft.

C-Code ist typischerweise deutlich länger als vergleichbare Programme in Perl oder Python. C ist die klassische Entwicklungssprache in Unix; viele Werkzeuge und praktisch alle Teile des Betriebssystems sind in C geschrieben.

```

#include <stdlib.h>
#include <stdio.h>
#include <assert.h>

/* integer vector */
int* ivec(int* x, int m) {
    int i;
    if (x==NULL) x=(int*)malloc(m*sizeof(int)); /* allocate if null */
    assert(x!=NULL);
    for (i=0;i<m;i++) x[i]=0; /* zero */
    return x;
}

int** mat(int r, int c) {
    int **x, i;
    x=(int**)malloc(r*sizeof(int*));
    assert(x!=NULL);
    for (i=0;i<r;i++) x[i]=ivec(NULL,c);
    return x;
}

void printvec(int* x, int n) {
    int i;
    printf("(");
    for (i=0;i<n;i++) {
        printf("%d",x[i]);
        if (i<n-1) printf(",");
    }
    printf(")");
}

void printmat(int** p, int n, int m) {
    int i,j;
    printf("(");
    for (i=0;i<n;i++) {
        printvec(p[i],m);
        if (i<n-1) printf(",");
    }
    printf(")");
}

int vecsum(int* x, int n) {
    int i,sum=0;
    for (i=0;i<n;i++) sum+=x[i];
    return sum;
}

```

7 Java

Objektorientiert, wenn in der JRE auch JIT (Just in Time) Compiler verfügbar ist, dann ist die Performance annähernd jener von C. JRE am Zielrechner erforderlich; verschiedene Versionen der JDK und JRE können Probleme bereiten.

```
class Primes {
    public static void main(String[] args) {
        int i,n=100000,cnt=0;
        for (i=2;i<n;i++) {
            if (prime(i)==1) {
                cnt++;
            }
        }
        System.out.println(cnt);
    }

    public static int prime(int x) {
        int i;
        for (i=2;i<x-1;i++) {
            if (x%i==0) return 0;
        }
        return 1;
    }
}
```

The Unix Programming Environment, by Brian W. Kernighan and Rob Pike. Prentice Hall, 1984.



A reader from down under. Picture by Phil McCrea