Programmiersprachen & -paradigmen

ein Überblick

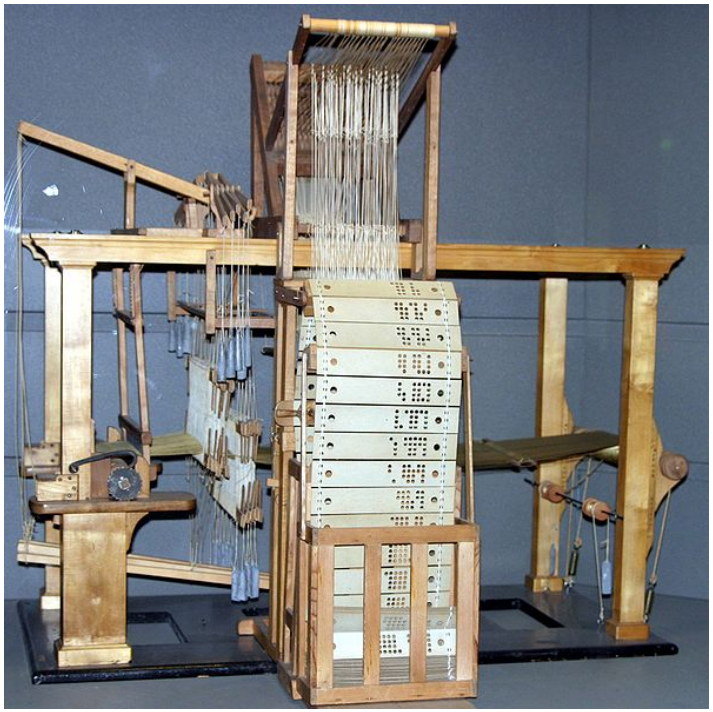
Programmierpraktikum 2013

04.07.2013

Übersicht

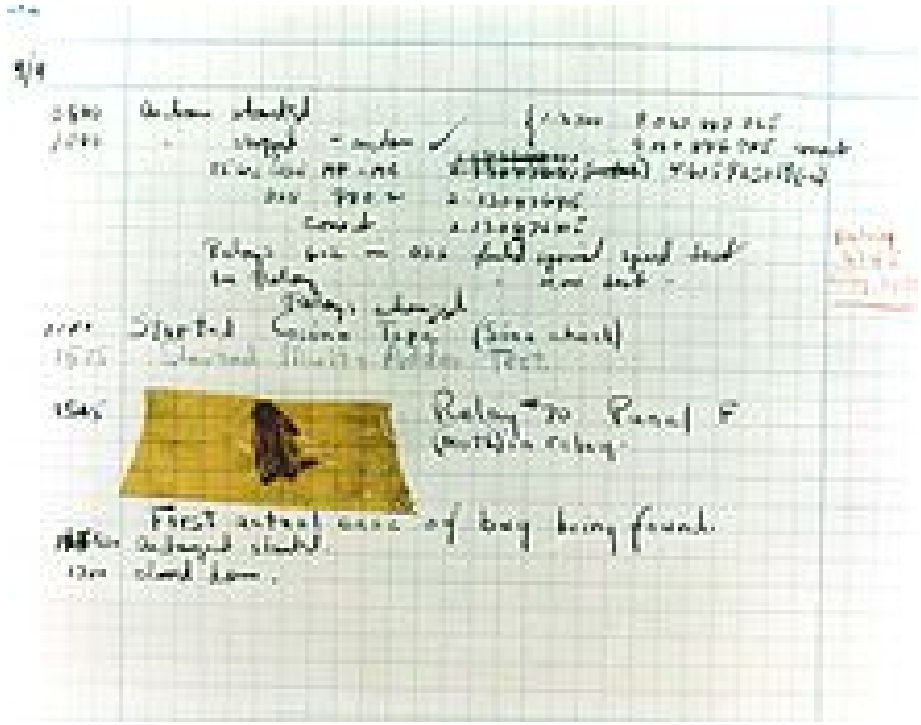
- Eine kurze Geschichte des Programmierens
- Überblick Sprachen und Paradigmen
- Codegolf
- Esoterische Sprachen

Jacquard-Webstuhl



- die erste programmierbare Maschine
- liest das zu webende Muster von Lochkarten
- und das 1805!

Der erste Bug



- 1947 von Grace Hopper aus den Relais gezogen
- "First actual case of bug being found"

Imperative Programmierung - Ausgangspunkt

- Programm ist Folge von Befehlen
- Kontrollfluss durch Sprünge
- Beispiele: frühes Cobol / Algol / Fortran
- Und: Assembler!
- Gegensatz: Deklarative Programmierung - doch dazu später

Strukturierte Programmierung

- Erweiterung der imperativen Programmierung
- "Go To Statement Considered Harmful"
  - Edsger W. Dijkstra, 1968
- Kontrollstrukturen werden eingeführt
- Beispiele: Pascal, frühes Basic

Prozedurale Programmierung

- Erweiterung der strukturierten Programmierung
- Programme werden in Teilaufgaben / Prozeduren zerlegt
- wesentlicher Abstraktionsschritt ...
- ... in Richtung Hochsprache
- Beispiele: C, Pascal, Fortran, Algol, Cobol

Modulare Programmierung

- Erweiterung der prozeduralen Programmierung
- Programmteile werden in Module zusammengefasst
- Module enthalten Methoden und deren Daten
- Und das führt uns zur objektorientierten Programmierung
  - Klassen statt Module
  - Vererbung, Polymorphie, ...
  - Beispiele: C++, Java, Python, und viele weitere

Und dann gab es da noch.....

- komponentenorientiert
- aspektorientiert
- generativ
- generisch
- subjektorientiert
- datenstromorientiert
- konkatenativ
- und weitere, mehr oder weniger gruselige Namen

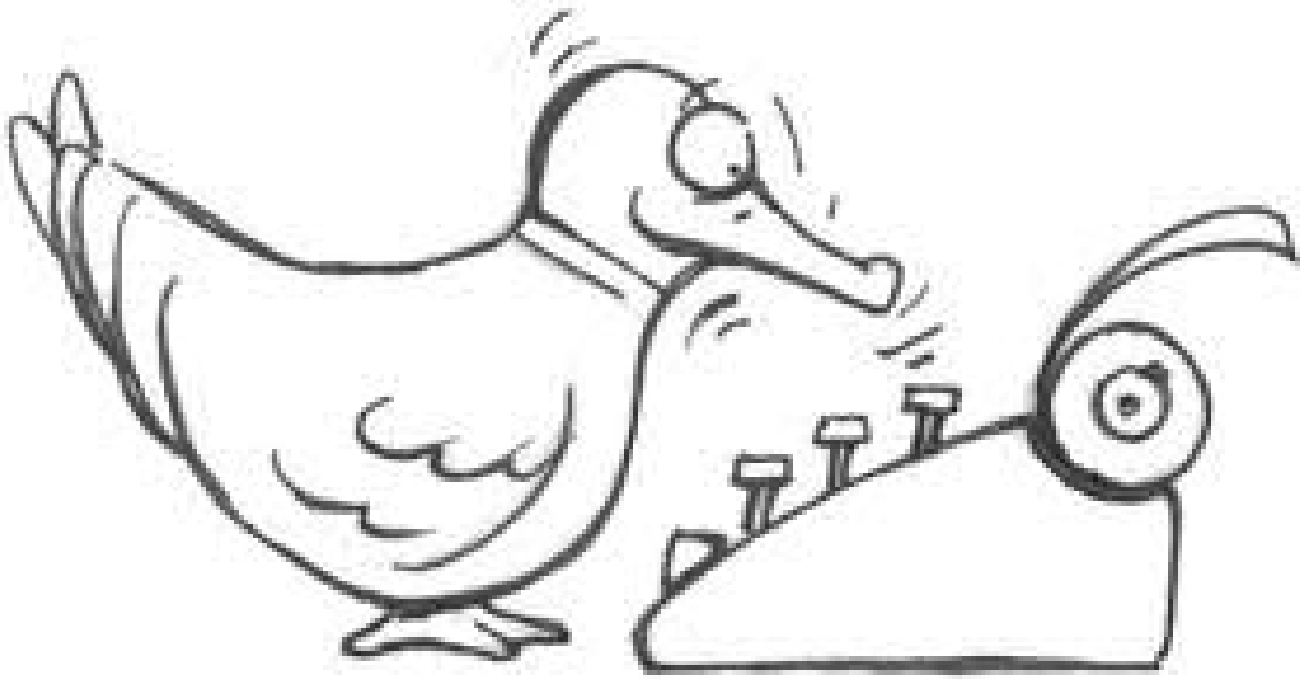
Übersicht

- In diesem Abschnitt “interessantere“ Sprachen
- Und Beispiele dazu
- Wir werden uns ansehen:
  - Python - Dynamische Sprache
  - Haskell - Funktionale Sprache
  - Prolog - Logische Sprache

Dynamisch?

- Typisierung - Duck Typing
- Reflektion / Instrospektion
- late-binding
- häufig interpretiert & mit Garbage Collection

Duck Typing



"When I see a bird that walks like a duck and swims like a duck and quacks like a duck, I call that bird a duck."
– James Whitcomb Riley

Beispiele

Python Beispiele

Einschub - Deklarative Programmierung

- Beschreiben des Problems, nicht des Lösungswegs
- Finden der Lösung der Sprache überlassen
- Programme kürzer und leichter zu verstehen
- Bessere Beweisbarkeit
- in der Regel weniger performant

Deklarativ vs. Imperativ

```
1  procedure quicksort(l,r : integer);
2  var x,i,j,tmp : integer;
3  begin
4    if r>l then
5    begin
6      x:=a[l]; i:=l; j:=r+1;
7      repeat
8        repeat  i:=i+1 until a[i]>=x;
9        repeat  j:=j-1 until a[j]<=x;
10       tmp:=a[j]; a[j]:=a[i]; a[i]:=tmp;
11     until j<=i;
12     a[i]:=a[j]; a[j]:=a[l]; a[l]:=tmp;
13     quicksort(l,j-1);
14     quicksort(j+1,r)
15   end
16 end;
```

```
1  quicksort [] = []
2  quicksort (x:xs) = quicksort [n | n<-xs, n<x] ++ [x] ++ quicksort [n | n<-xs, n
       >=x]
```

Funktional?

- Programme bestehen aus Funktionen
  - hängen nur von ihren Parametern ab
  - haben keine Nebeneffekte
- Funktionen höherer Ordnung
- Rekursive Funktionsaufrufe

Auswertungsstrategie

- strict - Argumente vor Funktionen
- eager - Ausdrücke als Ganzes
  - lazy - Auswertung erst wenn nötig
  - Ermöglicht unendliche Strukturen

Beispiele

Haskell Beispiele

Logisch?

- basierend auf mathematischer Logik
- Programm ist Menge von Axiomen und Folgerungen
- Interpreter versucht Anfrage zu “beweisen“
- Mehrere Lösungen für Anfrage möglich

Was bringt das?

- einfaches schreiben von Parsern und Interpretern
- KI
- IBM Watson system
- Computerlinguistik

Was noch?

- Angehöriger eines Adelshauses?
- Unsicher über den Nachwuchs?
- Frag PROLOG!

Beispiele

Prolog Beispiele

Codegolf - Mit welchem Schläger denn?

- Unübliche Programmieraufgabe
- Unübliche Zielsetzung
- Teils extra Sprachen zum Codegolfen
- Beispiele sagen mehr als tausend Worte...

We're no strangers to code golf, you know the rules, and so do I

- Must output the lyrics exactly as they appear in the above pastebin. Here's the raw dump: http://pastebin.com/raw.php?i=wwvdjvEj
- Cannot rely on any external resources - all lyrics must be generated by / embedded in code.
- No use of existing compression algorithms (e.g. gzip / bzip2) unless you include the full algorithm in your code.
- Use any language, shortest code wins.

Never gonna give up Ruby - 552 bytes

```
1  i=44
2  s="We; n7trangMsL8loT63Ke rules5s8d8I
3  AJull commit4nt'sChatFKink: of6CHldn'tRetKisJrom<ny@Ruy-/A= if?<sk 42DS'tLE 4?;
       Lo8bli=L7ee..
4  O,R1)O,R001)/-.."
5  "
6  I justCannaLE?2Gotta >u=Msta=.|
7  Ng1Nlet? downNrun<rH=5desMt?N>cryNsayRoodbyeNtE< lie5hurt?|
8
9  We'T3n each@Jor s8lSg6r hear9<ch: but6;Lo7hyL7BInsideCe both3Cha9Ro: S
10 We3KeRa45we;QpIB|1)O)NgiT, nPgiT
11 (G|iT? up| howFJeel:
12 [...]
```

vollstädiger Code: http://codegolf.stackexchange.com/a/6166 by Ed. H.

Anagramme

- Eingabe: 2 Strings
- Erkenne ob Anagramm
- Wenigste Zeichen gewinnen

Python - 32 byte

```
1 f=lambda a,b,S=sorted:S(a)==S(b)
```

Camelcase

- Ich mag Tiere...
- ...oder ich bin aus dem Iran (Okay, Rassismuspunkt für mich)
- Ich möchte ein Programm, dass ein Kamel auf der Konsole ausgibt...
- ... dessen Sourcecode wie ein Kamel aussieht

Camelcase

```
#!/usr/bin/perl -w                                          # camel code
use strict;

                                                  $_='ev
                                               al("seek\040D
                ATA,0,                       0;");foreach(1..3)
           {<DATA>;}my                     @camel1hump;my$camel;
   my$Camel   ;while(                    <DATA>){$_=sprintf("%-6
9s",$_);my@dromedary                   1=split(//);if(defined($
_=<DATA>)){@camel1hum                 p=split(//);}while(@dromeda
 ry1){my$camel1hump=0                ;my$CAMEL=3;if(defined($_=shif
        t(@dromedary1                ))&&/\S/){$camel1hump+=1<<$CAMEL;}
       $CAMEL--;if(d               efined($_=shift(@dromedary1))&&/\S/){
      $camel1hump+=1  <<$CAMEL;}$CAMEL--;if(defined($_=shift(
     @camel1hump))&&/\S/){$camel1hump+=1<<$CAMEL;}$CAMEL--;if(
     defined($_=shift(@camel1hump))&&/\S/){$camel1hump+=1<<$CAME
     L;;}$camel.=(split(//,"\040..m`{/J\047\134}L^7FX"))[$camel1h
      ump];}$camel.="\n";}@camel1hump=split(/\n/,$camel);foreach(@
      camel1hump){chomp;$Camel=$_;y/LJF7\173\175`\047/\061\062\063\
      064\065\066\067\070/;y/12345678/JL7F\175\173\047`/;$_=reverse;
       print"$_\040$Camel\n";}foreach(@camel1hump){chomp;$Camel=$_;y
        /LJF7\173\175`\047/12345678/;y/12345678/JL7F\175\173\0 47`/;
         $_=reverse;print"\040$_$Camel\n";}';;s/\s*//g;;eval;   eval
           ("seek\040DATA,0,0;");undef$/;$_=<DATA>;s/\s*//g;(    );;s
             ;^.*_;;;map{eval"print\"$_\"";}/.{4}/g; __DATA__    \124
               \1   50\145\040\165\163\145\040\157\1 46\040\1  41\0
                    40\143\141  \155\145\1 54\040\1   51\155\  141
                    \147\145\0  40\151\156 \040\141    \163\16 3\
```

Esoterische Sprachen - Motivation

- nicht für den praktischen Einsatz gedacht
- ungewöhnliche Konzepte / Syntax
- Spaß
- ungewöhnliche Ziele

Brainfuck

- Ziele:
  - touringmaschinenähnlich
  - möglichst kleiner Compiler

Brainfuck - Befehle

- <Zeiger inkrementieren
- >Zeiger dekrementieren
- + Wert inkrementieren
- - Wert dekrementieren
- . Wert als ASCII Zeichen ausgeben
- , Wert als ASCII Zeichen einlesen
- [Sprung nach vorne hinter passendes]
-]Sprung zurück wenn Wert ungleich Null

Ook!

- Brainfuck Dialekt mit leicht anderer Zielsetzung:
  1. Eine Programmiersprache sollte schreib- und lesbar für Orang-Utans sein
  2. Die Syntax sollte einfach sein, leicht zu merken und das Wort Affe vermeiden
  3. Bananen sind gut!
- Es gibt drei Syntaxelemente, von denen jeweils zwei zu einem Ook-Tupel zusammengefasst werden

Brainfuck - Hello World

```
1  ++++++++++
2  [
3   >+++++++>++++++++++>+++>+<<<<-
4  ]
5  >++.
6  >+.
7  +++++++.
8  .
9  +++.
10 >++.
11 <<++++++++++++++.
12 >.
13 +++.
14 ------.
15 --------.
16 >+.
17 >.
18 +++.
```

Ook - Hello World

```
1  Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
2  Ook. Ook. Ook. Ook. Ook! Ook? Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
3  Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook.
4  Ook! Ook. Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
5  Ook. Ook. Ook! Ook? Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook?
6  Ook! Ook! Ook? Ook! Ook? Ook. Ook. Ook. Ook! Ook. Ook. Ook. Ook. Ook. Ook. Ook.
7  Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook. Ook! Ook. Ook. Ook. Ook. Ook.
8  Ook. Ook. Ook! Ook. Ook. Ook? Ook. Ook? Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook.
9  Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook? Ook. Ook. Ook.
10 Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook. Ook! Ook.
11 Ook. Ook? Ook. Ook? Ook. Ook? Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
12 Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook! Ook? Ook? Ook. Ook. Ook.
13 Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook. Ook.
14 Ook. Ook? Ook! Ook! Ook? Ook! Ook? Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook.
15 Ook? Ook. Ook? Ook. Ook? Ook. Ook? Ook. Ook! Ook. Ook. Ook. Ook. Ook. Ook. Ook.
16 Ook! Ook. Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook.
17 Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook! Ook!
18 Ook! Ook. Ook. Ook? Ook. Ook? Ook. Ook. Ook! Ook.
```

Chef und SPL

- Chef: Programmiersprache soll sich lesen wie ein Kochrezept
- Variablen = Zutaten, Stacks = Rührschüsseln, Befehle = mischen, rühren, backen...
- Shakespeare Programming Language: soll sich lesen wie ein Sonett
- ermöglichen sehr viel Freiheit bei der Ausgestaltung eines Programms

Chef - Hello World

```
1  Hallo-Welt-Soufflee
2
3   Ingredients.
4   72 g haricot beans
5   101 eggs
6   108 g lard
7   111 cups oil
8   32 zucchinis
9   119 ml water
10  114 g red salmon
11  100 g dijon mustard
12  33 potatoes
13
14  Method.
15  Put potatoes into the mixing bowl.
16  Put dijon mustard into the mixing bowl.
17  Put lard into the mixing bowl.
18 [....]
19  Put lard into the mixing bowl.
20  Put lard into the mixing bowl.
21  Put eggs into the mixing bowl.
22  Put haricot beans into the mixing bowl.
23  Liquefy contents of the mixing bowl.
24  Pour contents of the mixing bowl into the baking dish.
25
26  Serves 1.
```

Malbolge

- Ziel: Schlimmstmögliche Programmiersprache
- Verwendet den ASCII Wert eines Zeichens als Befehl
- - Effekt eines Befehls hängt von Speicherstelle ab (modulo 94)
- - und plus einen Zähler (auch auf Datenpointer)
- - Speicherzugriffe nur ternär
- - nur unconditional jumps
- So kompliziert, dass das erste lauffähige Malbolge Programm von einem Suchalgorithmus gefunden wurde.....

Malbolge - Hello World

```
1  (=<':9876Z4321UT.-Q+*)M&%$H" !~}|Bzy?=|{z]KwZY44Eq0/{mlk**
2  hKs_dG5[m_BA{?-Y;;Vb'rR5431M}/.zHGwEDCBA@98\6543W10/.R,+O<
```

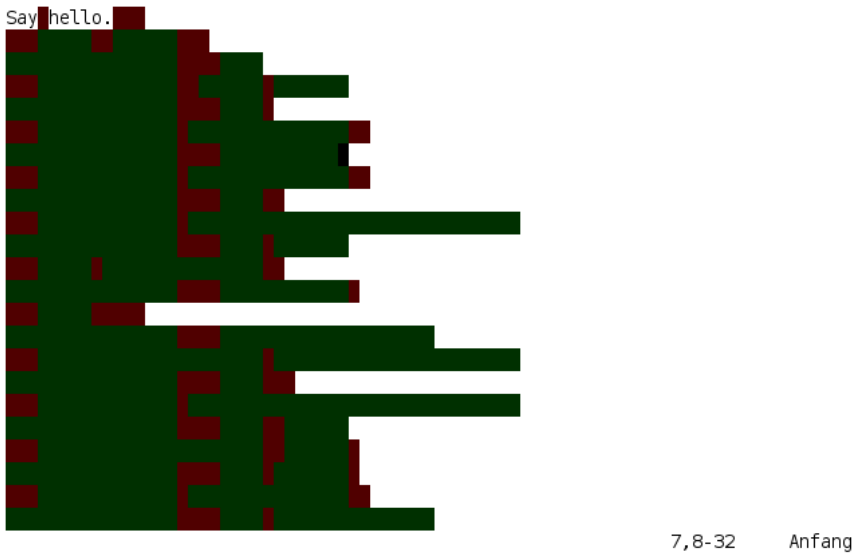
Befunge-93

- stackbasiert
- multidimensional
- Ziel: möglichst hart für Compiler (Befunge Compiler in Befunge: BefBef)

```
1  v   > . v
2
3  4   +   @
4
5  > 3 ^
```

Whitespace

- Besteht nur aus Whitespace, Tab und Linefeed
- Stack- / heapbasierend
- Released am 1. April 2003
- Hello World in Whitespace: (rot = Whitespace, grün = Tab)



Piet

- Source code sind Bitmap Bilder
- Sieht aus wie Pixelart
- Pointer folgt kontinuierlichen Farbabschnitten
- Farben sind besondere Befehle

Noch Fragen?

Gibt es noch Fragen oder Unklarheiten?

Dann bleibt nur noch...

Vielen Dank für Ihre Aufmerksamkeit!