Programmiersprachen Pascal, Modula-2, Oberon

Charlotte Prieß

Überblick

- Leben
- Historische Einordung
- Pascal
- Modula-2
- Oberon
- Vorteile von Pascal gegenüber C
- Warum sich C durchgesetzt ist?

Leben

- 1934 geboren in Winterthur, Schweiz
- 1954-1958 Studium der Elektrotechnik an der Eidgenössischen Technischen Hochschule (ETH) Zürich
- 1959: Dipl.-Ing., Heirat
- 1959-1960: Studium an der Université Laval, Quebec, Canada
- 1960-1963: Studium an der University of California, Berkeley, mit einem Stipendium der Ford Foundation
- 1963: Promotion mit einer Dissertation über die Generalisierung der Programmiersprache Algol 60
- 1963-1967: Assistant Professor für Computer Science an der Stanford University
- 1967-1968: Assistenzprofessur für Computerwissenschaften an der Universität Zürich
- 1968-1999: Professor für Computerwissenschaften an der ETH Zürich
- 1984: Auszeichnung mit dem Turing-Award

Historische Einordnung

- vor 1950:

  ~1837: Analyctical Engine Order Code

  1943-1945: Plankalkül (Konzept)

  1943-1946: ENIAC Coding System
- 1950er:

  1954-1955: FORTRAN “0” (Konzept)

  1956-1958: LISP (Konzept)

  1958: ALGOL 58

- 1960er:

  1960: ALGOL 60

  1962: Simula (Konzept)

  1964: BASIC

  1966: ALGOL W (Vorläufer von Pascal)
- 1970er

  1970: Pascal

  1972: Prolog

  1972: C

  1973: ML

  1979: Modula-2

Historische Einordnung

1980er:

- 1983: Ada (Erweiterung von Pascal)
- 1983: C++
- 1987: Perl
- 1987: Oberon

- 1990er:
- 1991: Python
- 1994: PHP
- 1995: Java

- 2000er:
- 2000: D
- 2000: C#

Pascal

Weiterentwicklung von Algol W mit folgenden Zielen:

- hoher Grad an Systematik
- Verwendung mathematischer Notationen
- einfache und intuitiv verständliche Regeln
- hohe Effizienz
- Sowohl für Wissenschaft als auch für Industrie geeignet

"...convinced that an elegant style and an effective implementation were not mutually exclusive..."

Pascal

- Festgelegte Struktur: Vereinbarungsteil, Anwenungsteil
- Starke Typisierung:
- Datentypen: int, char, real, boolean, Teilbereichstypen, Aufzählungstypen, Arrays, Records, Dateitypen, Zeigertypen
- Gültigkeitsbereiche: Trennung globaler und lokaler Variablen

Pascal

- Kontrollstrukturen:
- If then else, while do, repeat until, case, for
- Unterscheidung zwischen Prozeduren (“void”) und Funktionen
- “nested procedures” möglich

Pascal

- Inoffizieller Standard (“Report”)
- 1983 ISO 7185 (“unextended pascal”)
- 1990 ISO 10206 (“extended pascal”)
- sehr viele verschiedene Dialekte, die meisten mit den Standards inkompatibel
- Durchsetzen konnten sich v.a. die Versionen von Borland (Turbo Pascal, Delphi (=objektorientiertes Pascal)

Modula-2

- Sprache für Workstation Lilith
- Neuerungen:
- Multiprogramming
- Modularisierung und getrennte Kompilierung von Programmen
- Sprache soll ganze Systeme beschreiben können, daher u.a. Module zur Low-Level-Programmierung und Grafikprogrammierung
- Möglichkeit, bereits vorhandene Funktionen aus Bibliothken einzubinden

Oberon

- Neuerungen:
- Objektorientierung:
- Klassenkonzept
- Polymorphie, Late Binding
- Vererbung (keine Mehrfachvererbung)
- Kapselung (+ Schreibrecht, - Leserecht)
- Methoden und Attribute (Erweiterung des RECORD-Konzepts)

Vorteile von Pascal gegenüber C

- Besser lesbarer Code:
- In Pascal:

```
Function StringToUppercase(Zeile: String): String;

type Kleinbuchstaben = ['a'..'z'];

var temp: String;
    i    : Byte;

begin
  temp:=zeile;
  for i:=1 to Length(zeile) do
    if zeile[i] in Kleinbuchstaben  then temp[i]:=chr(ord(temp[i])-32);
  StringToUppercase:=temp;
end;
```

Vorteile von Pascal gegenüber C

- In C:

```
char * StringToUpper(char * zeile)
{
  char * temp;
  int i;
  int p = strlen(zeile);
  temp = (char *) malloc(p);
  for (i=0; zeile[i]!=0 && i<256; i++)
    if (zeile[i]>='a' && zeile[i]<='z')
      temp[i]=(char) zeile[i]-32; else
temp[i]=zeile[i];
  temp[i]=0;
  return temp;
}
```

Vorteile von Pascal gegenüber C

- Schwache Typisierung, keinerlei Überprüfungen zur Laufzeit:

```
#include "stdio.h"
void myprint(int *i)
{
  printf("i: %d\n",*i);
}
void main ()
{
  int i=10;
  myprint(&i);
  myprint(i);

}
```

Vorteile von Pascal gegenüber C

- Häufiger Fehler: Buffer Overflow

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int main(int argc, char *argv[])

{
	char mybuffer[5];

	if (argc < 2)

	{
		printf("strcpy() NOT executed....\n");
		printf("Syntax: %s <characters>\n", argv[0]);
		exit(0);
	}

	strcpy(mybuffer, argv[1]);
	printf("mybuffer content= %s\n", mybuffer);
}
```

Warum konnte sich C dennoch durchsetzen?

Mögliche Gründe:

- Mängel der ersten Pascal-Version: monolithische Programme, kein mächtiges I/O, keine hardwarenahe Programmierung usw.
- C hat sich mit Unix verbreitet
  - Programmieren unter Unix mit C einfacher als mit anderen Sprachen
  - C-Compiler bei Unix immer dabei
  - Studenten lernten C, weil sie so gleichzeitig Unix verstehen konnten
- auch viele andere wichtige OS in C implementiert
- C lässt dem Programmierer mehr Freiheiten

Quellen

- en.wikipedia.org, de.wikipedia.org
- http://www.bernd-leitenberger.de/pascal-und-c.shtml
- http://www.computerwoche.de/heftarchiv/1989/48/1153166/
- http://www.moorecad.com/standardpascal/
- http://www.eptacom.net/pubblicazioni/pub_eng/wirth.html
- http://www.swissdelphicenter.ch/en/niklauswirth.php
- Modula-2 Programmierhandbuch
- Pascal – User Manual and Report
- The Art of Simplicity
- Pioniere der Informatik
- Arbeitshefte Informatik – Objektorientierte Programmierung
- http://www.tenouk.com/Bufferoverflowc/Bufferoverflow1.html