ulm university universität
uulm

Institut für Numerische Mathematik

Dr. Andreas F. Borchert

20. Mai 2016
Blatt 5

Digitale Typografie (SS 2016)

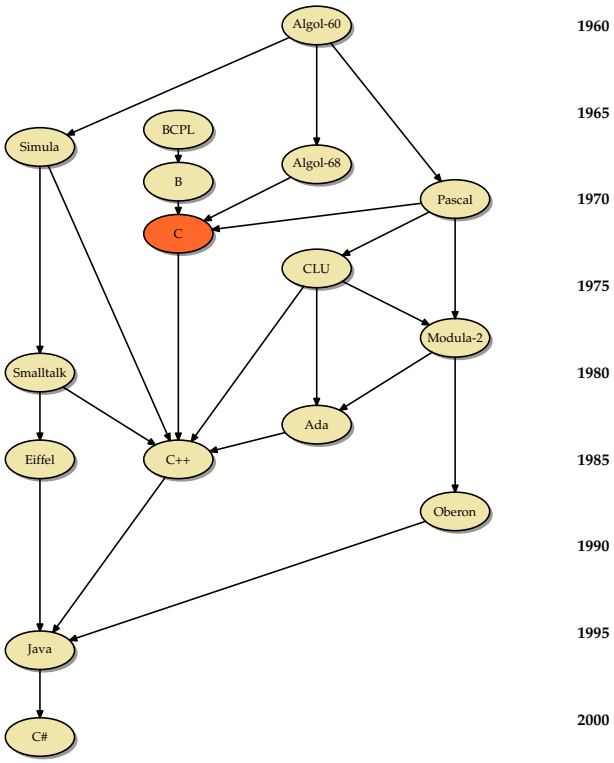
Abgabe bis zum 3. Juni 2016, 10:00 Uhr

Lernziele:

- Definition von Makros für Elemente eines komplexen Diagramms

Aufgabe 7: Programmiersprachen

Das folgende Diagramm zeigt einige der Entwicklungslinien rund um die Programmiersprache C:



Jede Sprache wird hier einer senkrecht verlaufenden Entwicklungslinie und einem Jahr zugeordnet. Ferner werden mit Pfeilen die direkten Beziehungen dargestellt. Im Rahmen

dieser Übungsaufgabe soll diese Art von Diagrammen mit Hilfe von Makros beschrieben und dann mit Hilfe der Makros automatisiert erzeugt werden. Konkret wurden in diesem Beispiel die einzelnen Entwicklungslinien und Programmiersprachen auf folgende Weise beschrieben:

```
DevelopmentLines(OOLine, CStyle, AlgolLine, PascalLine);

Language(Algol, AlgolLine, btex Algol-60 etex, 1960);

Language(BCPL, CStyle, btex BCPL etex, 1966);
Language(B, CStyle, btex B etex, 1969);
ThisLanguage(C, CStyle, btex C etex, 1972);
Language(Cpp, CStyle, btex C++ etex, 1985);

Language(Simula, OOLine, btex Simula etex, 1967);
Language(Smalltalk, OOLine, btex Smalltalk etex, 1980);
Language(Eiffel, OOLine, btex Eiffel etex, 1985);
Language(Java, OOLine, btex Java etex, 1996);
Language(Csharp, OOLine, btex C\# etex, 2001);

Language(AlgolSixtyEight, AlgolLine, btex Algol-68 etex, 1968);
Language(Pascal, PascalLine, btex Pascal etex, 1970);
Language(Modula, PascalLine, btex Modula-2 etex, 1978);
Language(CLU, AlgolLine, btex CLU etex, 1974);
Language(Ada, AlgolLine, btex Ada etex, 1983);
Language(Oberon, PascalLine, btex Oberon etex, 1988);
```

Danach erfolgt direkt oder implizit eine Positionierung des gesamten Diagramms einschließlich der Jahre, wobei der Jahresbereich automatisiert zu bestimmen ist. Es darf aber gerne festgelegt werden, dass die Jahre in 5er-Schritten rechts eingezeichnet werden. Wenn dies alles feststeht, können die einzelnen Pfeile spezifiziert werden. Das Arrow-Makro kann dann davon ausgehen, dass die Pfade für den Start- und den Zielknoten bereits existieren, so dass dann der *intersectionpoint*-Operator problemlos verwendet werden kann.

```
Arrow(Algol, AlgolSixtyEight); Arrow(Algol, Simula);
Arrow(Algol, Pascal); Arrow(Pascal, Modula);
Arrow(Modula, Oberon);

Arrow(BCPL, B); Arrow(B, C);
Arrow(AlgolSixtyEight, C); Arrow(Pascal, C);

Arrow(Simula, Smalltalk); Arrow(Simula, Cpp);
Arrow(C, Cpp); Arrow(Smalltalk, Eiffel);
Arrow(Eiffel, Java); Arrow(Smalltalk, Cpp);
Arrow(Pascal, CLU); Arrow(Modula, Ada);
Arrow(CLU, Modula); Arrow(CLU, Cpp);
Arrow(CLU, Ada); Arrow(Ada, Cpp);
```

```
Arrow(Cpp, Java); Arrow(Oberon, Java);
Arrow(Java, Csharp);
```

Für Ihre Lösung können Sie gerne auch eine andere Entwicklungslinienthematik verwenden. Auch steht es Ihnen frei, die Gestaltung zu verändern – etwa unter der Verwendung von Boxen an Stelle von Ellipsen. Wenn Sie Ellipsen verwenden möchten, empfiehlt es sich, Kreise mit dem *xscaled*-Operator zu geeigneten Ellipsen zu verwandeln. In jedem Falle wäre es angemessen, sämtliche Knoten gleich groß zu gestalten und die Wahl der Größe automatisiert festzulegen aufgrund der ausgemessenen Texte für die einzelnen Knoten.

Wenn Sie in *btex* ... *etex* beliebige TEX-Konstruktionen unterbringen können möchten, empfiehlt es sich, mit folgendem durch *verbatimtex* ... *etex* eingeklammerten Prolog die benötigten LATEX-Pakete einzubinden:

```
verbatimtex
%&latex
\documentclass[12pt]{article}
\usepackage{palatino}
\usepackage{mathpple}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\begin{document}
etex
```

Wenn Sie mit Ihrer Lösung fertig sind, sollten Sie die die METAPOST-Datei `diagramm.mp` nennen und diese mit dem Kommando

```
submit typo 7 diagramm.mp
```

auf der Thales einzureichen.

Viel Erfolg!