Algol Part 1

CS4100

March 2, 2011

Based on slides by Istvan Jonyer

After FORTRAN

- International language is needed
  - 1964: New language is proposed to break away from platform dependence
  - Preliminary spec: NPL (new programming language), then PL/I (programming language 1)
  - PL/I is too big
    - Dijkstra: If Fortran is an infantile disorder, then PL/I is a fatal disease
    - Trying to be everything to everyone backfires

Chapter 3: Generality and Hierarchy: ALGOL-60

- An international language is needed
  - A single, universal language would be valuable
  - International (American and European) committee is set up to make recommendations
  - Algol-58 is created in 8 days in Zurich, as a preliminary report
  - Algol: Algorithmic Language

Implementations

- Because of the hype, many started implementation quickly
  - This resulted in many dialects
  - JOVIAL (Jules' Own Version of the International Algebraic Language)
- Committee meets again in 1960 to incorporate suggestions
  - Algol-60 is born and is very different from the '58 report.
  - Report is 17 pages long: remarkable achievement, mainly due to BNF notation (reports used to stretch to hundreds or thousands of pages)

Algol Report

- 1959 UNESCO Conference on Information Processing
  - Backus presents a description of Algol '58
    - Uses formal syntax he developed
  - Naur is editor of Algol Bulletin
    - Disagrees with some of Backus' interpretation
    - Need for more precise description
    - Develops a variant of Backus' formal syntax

Backus-Naur Form, aka BNF used for 1960 Algol Report

5

Algol's Objectives

- The language should be very close to mathematical notation
- Should be useful in publications to describe algorithms
- Mechanically translatable to machine code

6

Structural Organization

- Hierarchically structured language
  - Nesting is introduced (Fortran did not use nesting)
  - Control structures can also be nested
    - One can be made the body of the other

```
if N > 0 then
  for i := 1 step 1 until N do
      sum := sum + Data[i]
```

    - Advantage: decreases the number of GOTOs required
- Reserved words

7

Constructs

- Declarative or Imperative
  - (like in FORTRAN)
- Variable declarations
  - Types: integer, real, Boolean

```
integer i, j, k
```

  - Lower bounds of arrays need not be 1

```
real array Data[-50:50]
```

  - Switch, like FORTRAN's computed GOTO
- Subprogram declarations
  - Keyword: **procedure** and
  - Procedures can be *typed* (functions) and *untyped*

```
real procedure dist(x1, y1, x2, y2);
  real x1, y1, x2, y2;
  dist = sqrt((x1 - x2)^2 + (y1 - y2)^2)
```

8