# ALGOL 60: The Death of a Programming Language and the Birth of a Science

Huub de Beer

Eindhoven School of Education

Eindhoven, February 12, 2010

# Outline

1. Introduction
2. 1950s: The Era of the Prototype
3. 1955-1965: The Era of the Algorithmic Language
4. The Birth of a Science
5. 1960s: Onwards to General Purpose Programming Languages
6. Conclusion

# The Birth of a Science

## Mahoney (†2008)

- (1955–1975) Computer science established as an independent science
- Science: (research) community with its own *agenda*:
  - Problems
  - Knowledge
  - Tools
  - Techniques

# The Birth of a Science

## Mahoney (†2008)

- (1955–1975) Computer science established as an independent science
- Science: (research) community with its own *agenda*:
  - Problems
  - Knowledge
  - Tools
  - Techniques

## Thesis

ALGOL 60 was a *catalyst* in the transformation of the field of computing into an independent science

# Contents

1. Introduction
2. 1950s: The Era of the Prototype
3. 1955-1965: The Era of the Algorithmic Language
4. The Birth of a Science
5. 1960s: Onwards to General Purpose Programming Languages
6. Conclusion

# Outline

1. Introduction
2. 1950s: The Era of the Prototype
3. 1955-1965: The Era of the Algorithmic Language
4. The Birth of a Science
5. 1960s: Onwards to General Purpose Programming Languages
6. Conclusion

# 1950s: The Era of the Prototype

## Early Computers

- ENIAC, ARC, Manchester Baby. . .
- (1949) EDSAC, Cambridge (Wilkes)

  → first *working* Von Neumann stored-program computer
- Ferranti Mark I (1951), UNIVAC I (1951), IBM 650 (1954)

# 1950s: The Era of the Prototype

## Early Computers

- ENIAC, ARC, Manchester Baby...
- (1949) EDSAC, Cambridge (Wilkes)

→ first *working* Von Neumann stored-program computer

- Ferranti Mark I (1951), UNIVAC I (1951), IBM 650 (1954)

## Example: The Mathematical Center, Amsterdam

- (1946) Foundation: Mathematics useful to society
- (1947) Van Wijngaarden head of computing department

→ international ambitions; dreaming of the AERA

# 1950s: The Era of the Prototype

## Early Computers

- ENIAC, ARC, Manchester Baby. . .
- (1949) EDSAC, Cambridge (Wilkes)

→ first *working* Von Neumann stored-program computer

- Ferranti Mark I (1951), UNIVAC I (1951), IBM 650 (1954)

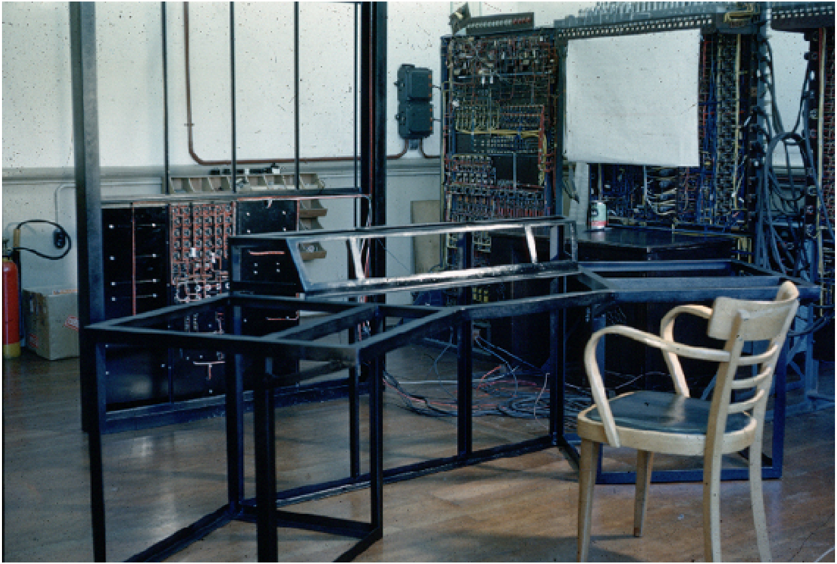
## Example: The Mathematical Center, Amsterdam

- (1946) Foundation: Mathematics useful to society
- (1947) Van Wijngaarden head of computing department

→ international ambitions; dreaming of the AERA

→ This is an agenda of Mathematics

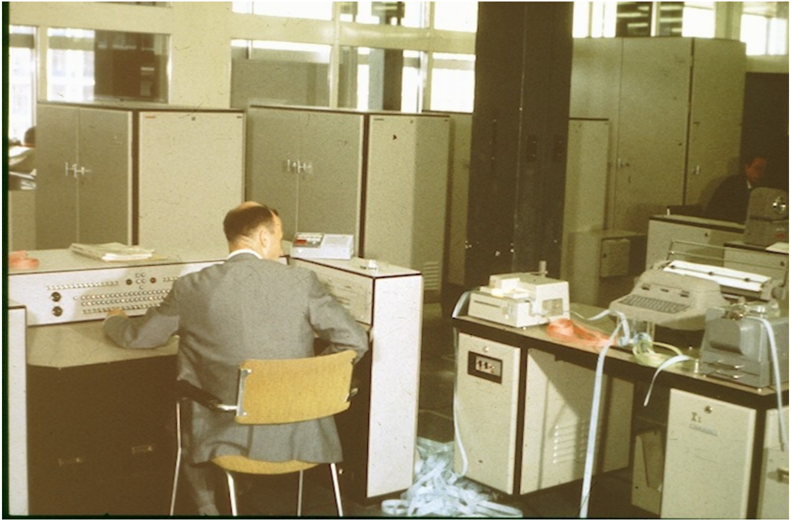
# 1953: Rebuilding the ARRA Computer



# Computer Use at the Mathematical Center

| year | A | MC | ARRA I | ARRA II | ARMAC | X1 | % |
|---|---|---|---|---|---|---|---|
| 1946 | | | | | | | – |
| 1947 | | | | | | | – |
| 1948 | | | | | | | – |
| 1949 | 39 | | | | | | 0.0% |
| 1950 | 52 | 27 | 2 | | | | 3.8% |
| 1951 | 59 | 21 | 1 | | | | 1.6% |
| 1952 | 48 | 17 | 1 | | | | 2.0% |
| 1953 | 52 | 13 | | | | | 0.0% |
| 1954 | 59 | 8 | | 8 | | | 13.6% |
| 1955 | 53 | 13 | | 20 | | | 37.7% |
| 1956 | 60 | 9 | | 5 | 13 | | 30.0% |
| 1957 | 73 | 9 | | | 38 | | 52.1% |
| 1958 | 57 | 5 | | | 28 | | 49.1% |
| 1959 | 55 | 6 | | | 17 | 2 | 34.5% |
| 1960 | 69 | 7 | | | 1 | 42 | 62.3% |
| 1961 | 122 | 11 | | | | 122 | 100.0% |
| 1962 | 179 | | | | | 179 | 100.0% |

# The Electrologica X1 (1958): 2° generation computer



Transistorized, core memory, interrupt, and I/O:
Reliable and fast

# Problematic Machines are No More

- For ten years, computing machines were a problem (of research)
- Around **1958**:
  - Fast and reliable computers (second generation) were available
  - For a reasonable price
  - Hence, more computer installations
  - With more (uninitiated) users
  - And a lot of scientific computational problems

# Problematic Machines are No More

- For ten years, computing machines were a problem (of research)
- Around **1958**:
  - Fast and reliable computers (second generation) were available
  - For a reasonable price
  - Hence, more computer installations
  - With more (uninitiated) users
  - And a lot of scientific computational problems
- Programming becomes a problem (of research)

→ wouldn't it be nice if one could speak mathematics to a computer?

# Problematic Machines are No More

- For ten years, computing machines were a problem (of research)
- Around **1958**:
  - Fast and reliable computers (second generation) were available
  - For a reasonable price
  - Hence, more computer installations
  - With more (uninitiated) users
  - And a lot of scientific computational problems
- Programming becomes a problem (of research)

→ wouldn't it be nice if one could speak mathematics to a computer?

→ Still on the agenda of Mathematics

# Outline

1 Introduction

2 1950s: The Era of the Prototype

3 1955-1965: The Era of the Algorithmic Language

4 The Birth of a Science

5 1960s: Onwards to General Purpose Programming Languages

6 Conclusion

# Early Algorithmic Language Efforts

## Europe: theoretical

- (1946) Zuse's Plankalkül
- (1951) Rutishauser's language

# Early Algorithmic Language Efforts

## Europe: theoretical

- (1946) Zuse's Plankalkül
- (1951) Rutishauser's language

## USA: experimental and practical

- (1953) Backus's FORTRAN (IBM 704)
- (1956) Perlis and Smith's Internal Translator (Datatron; IBM 650)
- (1957) Katz's MATH-MATIC (UNIVAC I)
- (1958) FORTRAN II (range of IBM machines)

# The Start of the ALGOL Effort

## USA

- Different efforts to create an algebraic language
- USE, SHARE, and DUO call for unification of efforts
- (1957) ACM subcommittee on a universal algebraic language

# The Start of the ALGOL Effort

## USA

- Different efforts to create an algebraic language
- USE, SHARE, and DUO call for unification of efforts
- (1957) ACM subcommittee on a universal algebraic language

## Central Europe

- Bauer and Samelson: Interested in formula translation
- (1955) Darmstadt symposium → GAMM subcommittee for programming languages
- (1957) GAMM subcommittee almost finished: ‘make an effort to worldwide unification’

# The International Algebraic Language

## (1958) Joint meeting at Zürich

Based on two proposals; aiming at:

- Close to mathematical notation (*writable*)
- Publication language (*readable*)
- Machine translatable
- Machine independent

# The International Algebraic Language

## (1958) Joint meeting at Zürich

Based on two proposals; aiming at:

- Close to mathematical notation (*writable*)
- Publication language (*readable*)
- Machine translatable
- Machine independent

## Preliminary Report: International Algebraic Language

- Yet another algebraic language; no I/O
- Some nice features and firsts:

  → compound statement, boolean type, and procedure
- Generated interest from all over (Western) Europe

# Developing ALGOL 60: A truly international effort

## Separate discussions

- (USA) Practical: more data types, I/O, sugar
- (Europe) Theoretical: problematic procedure

People from around Europe participate

# Developing ALGOL 60: A truly international effort

## Separate discussions

- (USA) Practical: more data types, I/O, sugar
- (Europe) Theoretical: problematic procedure

  People from around Europe participate

## UNESCO conference on Information Processing (Paris, 1959)

- Buzz about IAL
- Backus's notation: trying to define IAL's syntax formally
  - "*Heretofore there has existed no formal description of a machine-independent language.*"
  - Based on Post's production system
  - Unable to completely and satisfactory define IAL's syntax

# Developing ALGOL 60: A truly international effort

## Separate discussions

- (USA) Practical: more data types, I/O, sugar
- (Europe) Theoretical: problematic procedure

People from around Europe participate

## UNESCO conference on Information Processing (Paris, 1959)

- Buzz about IAL
- Backus's notation: trying to define IAL's syntax formally
  - "*Heretofore there has existed no formal description of a machine-independent language.*"
  - Based on Post's production system
  - Unable to completely and satisfactory define IAL's syntax

→ ALGOL an important part of an agenda

# (1960) ALGOL 60 Meeting in Paris

## Naur’s preparation

- Use of BNF to define large parts of the language
- Draft was highly structured
- Basis of the meeting → Naur becomes editor

# (1960) ALGOL 60 Meeting in Paris

## Naur's preparation

- Use of BNF to define large parts of the language
- Draft was highly structured
- Basis of the meeting → Naur becomes editor

## An impression (Perlis, 1978)

"*The meetings were exhausting, interminable, and exhilarating. (...) diligence persisted during the entire period, the chemistry of the 13 was excellent. (...) Progress was steady and the output, Algol 60, was more racehorse than camel.*"

# Procedure Concept: IAL and ALGOL 60

## IAL

$$
\begin{aligned}
\langle\text{oe}\rangle &:\equiv \langle\text{left element}\rangle\\
\langle\text{out list}\rangle &:\equiv \langle\text{oe}\rangle \textit{ or } \langle\text{outlist}\rangle, \langle\text{oe}\rangle\\
\langle\text{suc}\rangle &:\equiv \langle\text{label}\rangle \textit{ or } \langle\text{id}\rangle \text{ [ } \langle\text{exp}\rangle \text{ ]}\\
\langle\text{succr list}\rangle &:\equiv \langle\text{suc}\rangle \textit{ or } \langle\text{succr list}\rangle,\\
&\quad \langle\text{suc}\rangle\\
\langle\text{A}\rangle &:\equiv =:(\langle\text{out list}\rangle) \textit{ or } \langle\text{blank}\rangle\\
\langle\text{B}\rangle &:\equiv :(\langle\text{succr list}\rangle) \textit{ or } \langle\text{blank}\rangle\\
\langle\text{proc stmt}\rangle &:\equiv \langle\text{function}\rangle\ \langle\text{A}\rangle\ \langle\text{B}\rangle \textit{ or}\\
&\quad \langle\text{id}\rangle =:(\langle\text{outlist}\rangle)\ \langle\text{B}\rangle \textit{ or}\\
&\quad \langle\text{id}\rangle:(\langle\text{succr list}\rangle)\\
\langle\text{ppol}\rangle &:\equiv \langle\text{blank}\rangle \textit{ or } \langle\text{ppol}\rangle\ \langle\text{oe}\rangle,\\
\langle\text{pol}\rangle &:\equiv \langle\text{ppol}\rangle \textit{ or } \langle\text{pol}\rangle, \textit{ or}\\
&\quad \langle\text{pol}\rangle, \langle\text{oe}\rangle\\
\langle\text{A'}\rangle &:\equiv =:(\langle\text{pol}\rangle)\\
\langle\text{ppsl}\rangle &:\equiv \langle\text{blank}\rangle \textit{ or } \langle\text{ppsl}\rangle\ \langle\text{suc}\rangle,\\
\langle\text{psl}\rangle &:\equiv \langle\text{ppsl}\rangle \textit{ or } \langle\text{psl}\rangle, \textit{ or } \langle\text{psl}\rangle,\\
&\quad \langle\text{suc}\rangle\\
\langle\text{B'}\rangle &:\equiv :(\langle\text{psl}\rangle)
\end{aligned}
$$

# Procedure Concept: IAL and ALGOL 60

## IAL

⟨oe⟩ :≡ ⟨left element⟩

⟨out list⟩ :≡ ⟨oe⟩ *or* ⟨outlist⟩, ⟨oe⟩

⟨suc⟩ :≡ ⟨label⟩ *or* ⟨id⟩ [ ⟨exp⟩ ]

⟨succr list⟩ :≡ ⟨suc⟩ *or* ⟨succr list⟩, ⟨suc⟩

⟨A⟩ :≡ =:(⟨out list⟩) *or* ⟨blank⟩

⟨B⟩ :≡ :(⟨succr list⟩) *or* ⟨blank⟩

⟨proc stmt⟩ :≡ ⟨function⟩ ⟨A⟩ ⟨B⟩ *or* ⟨id⟩ =:(⟨outlist⟩) ⟨B⟩ *or* ⟨id⟩:(⟨succr list⟩)

⟨ppol⟩ :≡ ⟨blank⟩ *or* ⟨ppol⟩ ⟨oe⟩,

⟨pol⟩ :≡ ⟨ppol⟩ *or* ⟨pol⟩, *or* ⟨pol⟩, ⟨oe⟩

⟨A’⟩ :≡ =:(⟨pol⟩)

⟨ppsl⟩ :≡ ⟨blank⟩ *or* ⟨ppsl⟩ ⟨suc⟩,

⟨psl⟩ :≡ ⟨ppsl⟩ *or* ⟨psl⟩, *or* ⟨psl⟩, ⟨suc⟩

⟨B’⟩ :≡ :(⟨psl⟩)

## ALGOL 60

<actual parameter> ::= <string> | <expressions> | <array identifier> | <switch identifier> | <procedure identifier>

<letter string> ::= <letter> | <letter string><letter>

<parameter delimeter> ::= , | )<letter string>:(

<actual parameter list> ::= <actual parameter> | <actual parameter list><parameter delimeter><actual parameter>

<actual parameter part> ::= <empty> | (<actual parameter list>)

<procedure statement> ::= <procedure identifier><actual parameter part>

# The ALGOL 60 Report

- Highly structured
- Definition of syntax using BNF
- Recursion: BNF, definition in BNF and the controversial recursive procedures
- Some nice features: block, if-statement, procedure, multiple assignment, . . .
- Set a standard for subsequent language reports

# (Early 1960s) Use and Maintaining ALGOL 60

## Implementation and use

- (August 1960) Dijkstra-Zonneveld compiler; first complete ALGOL 60 compiler
- Many follow all around the world
- Publication language: *Communications of the ACM*, *Numerische Mathematik*, *Computer Journal*, . . .

## Maintenance

- Discussions in the ALGOL Bulletin (European)
- Remove ambiguities, solve problems
- (1962) Revised ALGOL 60 report
- (1962) Under IFIP flag: ALGOL was now institutionalised
- Working Group 2.1: defined a subset of ALGOL and I/O procedures

# Outline

1. Introduction
2. 1950s: The Era of the Prototype
3. 1955-1965: The Era of the Algorithmic Language
4. The Birth of a Science
5. 1960s: Onwards to General Purpose Programming Languages
6. Conclusion

# ALGOL on the Mathematics Agenda

- Create automatic calculators for numerical calculations

# ALGOL on the Mathematics Agenda

- Create automatic calculators for numerical calculations
- Create a programming language (ALGOL) to make programming numerical algorithms easier

# ALGOL on the Mathematics Agenda

- Create automatic calculators for numerical calculations
- Create a programming language (ALGOL) to make programming numerical algorithms easier
- Create a programming language (ALGOL) to communicate numerical algorithms with other mathematics practitioners

# ALGOL on the Mathematics Agenda

- Create automatic calculators for numerical calculations
- Create a programming language (ALGOL) to make programming numerical algorithms easier
- Create a programming language (ALGOL) to communicate numerical algorithms with other mathematics practitioners
- Create a translator (for ALGOL) for my automatic calculator

# ALGOL on the Mathematics Agenda

- Create automatic calculators for numerical calculations
- Create a programming language (ALGOL) to make programming numerical algorithms easier
- Create a programming language (ALGOL) to communicate numerical algorithms with other mathematics practitioners
- Create a translator (for ALGOL) for my automatic calculator
- Create a body wherein this language (ALGOL) is maintained

# ALGOL on the Mathematics Agenda

- Create automatic calculators for numerical calculations
- Create a programming language (ALGOL) to make programming numerical algorithms easier
- Create a programming language (ALGOL) to communicate numerical algorithms with other mathematics practitioners
- Create a translator (for ALGOL) for my automatic calculator
- Create a body wherein this language (ALGOL) is maintained

**So, the agenda has been completed!?**

# A New Agenda: ALGOL and its implications

- Writing a translator for ALGOL → systems software

# A New Agenda: ALGOL and its implications

- Writing a translator for ALGOL → systems software
- Writing a translator for ALGOL → General problem of writing translators for ALGOL-like languages

# A New Agenda: ALGOL and its implications

- Writing a translator for ALGOL → systems software
- Writing a translator for ALGOL → General problem of writing translators for ALGOL-like languages
- A notation (BNF) with some far reaching implications:
- (Ginsburg & Rice, 1962) Connection with linguistics: ALGOL-like language are context-free languages → Formal languages
- Structure of ALGOL-like languages → Generate translators for ALGOL-like languages

# A New Agenda: ALGOL and its implications

- Writing a translator for ALGOL → systems software
- Writing a translator for ALGOL → General problem of writing translators for ALGOL-like languages
- A notation (BNF) with some far reaching implications:
- (Ginsburg & Rice, 1962) Connection with linguistics: ALGOL-like language are context-free languages → Formal languages
- Structure of ALGOL-like languages → Generate translators for ALGOL-like languages
- IFIP WG 2.1 was a discussion forum on programming languages
- ALGOL used for more than just numerical algorithms: systems programming, symbol manipulation, text processing, data processing

# A New Agenda: ALGOL and its implications

- Writing a translator for ALGOL → systems software
- Writing a translator for ALGOL → General problem of writing translators for ALGOL-like languages
- A notation (BNF) with some far reaching implications:
- (Ginsburg & Rice, 1962) Connection with linguistics: ALGOL-like language are context-free languages → Formal languages
- Structure of ALGOL-like languages → Generate translators for ALGOL-like languages
- IFIP WG 2.1 was a discussion forum on programming languages
- ALGOL used for more than just numerical algorithms: systems programming, symbol manipulation, text processing, data processing

**ALGOL had become the typical example or vessel for a whole new set of problems → a new *agenda*: a science is born**

# Beyond Implementing ALGOL: Exploiting its Structure

## Onward to recursive descent parsing

- Grau (1961) *Recursive Processes and ALGOL Translation*:

  A ALGOL translator should be recursive to recursively translate ALGOL programs
- Lucas (1961) *The Structure of Formula-Translators*

# Beyond Implementing ALGOL: Exploiting its Structure

## Onward to recursive descent parsing

- Grau (1961) *Recursive Processes and ALGOL Translation*:

  A ALGOL translator should be recursive to recursively translate ALGOL programs
- Lucas (1961) *The Structure of Formula-Translators*

## Onward to compiler generators

- Irons (1961) *A Syntax Directed Compiler for ALGOL 60*
- Ledley and Wilson (1962) *Automatic-Programming Language Translation Through Syntactical Analysis*
- Irons (1963) *The Structure and Use of the Syntax Directed Compiler*:

  Separate the definition of a language and the translation of a language: meta language and a general translation program

# Outline

1 Introduction

2 1950s: The Era of the Prototype

3 1955-1965: The Era of the Algorithmic Language

4 The Birth of a Science

5 1960s: Onwards to General Purpose Programming Languages

6 Conclusion

# (±1960) The Fab Four

### FORTRAN II (1958)

Language for numerical computations; Aim: as fast as hand-coded programs

### LISP (1958–1962)

Symbol manipulation; AI

### COBOL (1959)

Language for data processing: Intended for business users; Context of large scale punch card data processing

### ALGOL 60 (1960)

Algorithmic language: Numerical computation; Publication language

**If ALGOL was so important, why is ALGOL the one that died?**

# (±1964) ALGOL Diagnosed with Lack of Features

- Once people started programming in ALGOL, soon they broke out of the small field of numerical computation:
  - Information processing: Data structures; Searching, sorting
  - Symbol manipulation
  - Text processing
  - Systems programming (even an ALGOL compiler in ALGOL)

# (±1964) ALGOL Diagnosed with Lack of Features

- Once people started programming in ALGOL, soon they broke out of the small field of numerical computation:
  - Information processing: Data structures; Searching, sorting
  - Symbol manipulation
  - Text processing
  - Systems programming (even an ALGOL compiler in ALGOL)
- ALGOL became a hammer, and a bad one at that: a new ALGOL was needed

# Solution: A General Purpose Programming Language

## PL/I (1963–1964)

For business data processing and numerical computations: a combination of FORTRAN, COBOL and ALGOL 60 with a lot of features.

## IFIP Working Group 2.1 (1964): The next ALGOL

- (1964) Start working on ALGOL X and ALGOL Y

  Duncan (revived ALGOL Bulletin, 1964):
  "*there was a considerable body of opinion in favour of developing a so-called 'ALGOL X' by building extensions on to ALGOL 60. This extended language would provide both a long overdue short-term solution to existing difficulties and a useful tool in the development of the radically reconstructed future ALGOL (the so-called 'ALGOL Y')*"

# Extending ALGOL: SIMULA

- (1962) Kristen Nygaard and Ole-Johan Dahl start with the development of SIMULA
- SIMULA is a discrete event simulation language
- (1962-1963) Preprocessor for ALGOL 60 with a large library

# Extending ALGOL: SIMULA

- (1962) Kristen Nygaard and Ole-Johan Dahl start with the development of SIMULA
- SIMULA is a discrete event simulation language
- (1962-1963) Preprocessor for ALGOL 60 with a large library
- (1963-1964) Adapting ALGOL 60 compiler: SIMULA is ready for use

  (Problematic ALGOL 60 implementation on the UNIVAC)

# Extending ALGOL: SIMULA

- (1962) Kristen Nygaard and Ole-Johan Dahl start with the development of SIMULA
- SIMULA is a discrete event simulation language
- (1962-1963) Preprocessor for ALGOL 60 with a large library
- (1963-1964) Adapting ALGOL 60 compiler: SIMULA is ready for use

  (Problematic ALGOL 60 implementation on the UNIVAC)
- (1965) New SIMULA as a general purpose language: SIMULA 67
- (1968) SIMULA 67 Common Base Language set
- (1969) First compiler ready

# ALGOL X: Wishes and Proposals

## Wishes after two years of using ALGOL

- I/O facilities
- Symbol manipulation
- A better for statement
- Double precision numbers
- More standard types
- User-defined types
- . . .

# ALGOL X: Wishes and Proposals

### Wishes after two years of using ALGOL

- I/O facilities
- Symbol manipulation
- A better for statement
- Double precision numbers
- More standard types
- User-defined types
- . . .

### Proposals for ALGOL X

- Case expression (replacing the switch)
- Naur's Environment Enquiry (using machine information)
- All-statement (sort of an foreach?)
- reference type
- (C.A.R. Hoare, 1965) Record type
- . . .

# The End of The ALGOL Effort: Kootwijk, 1966

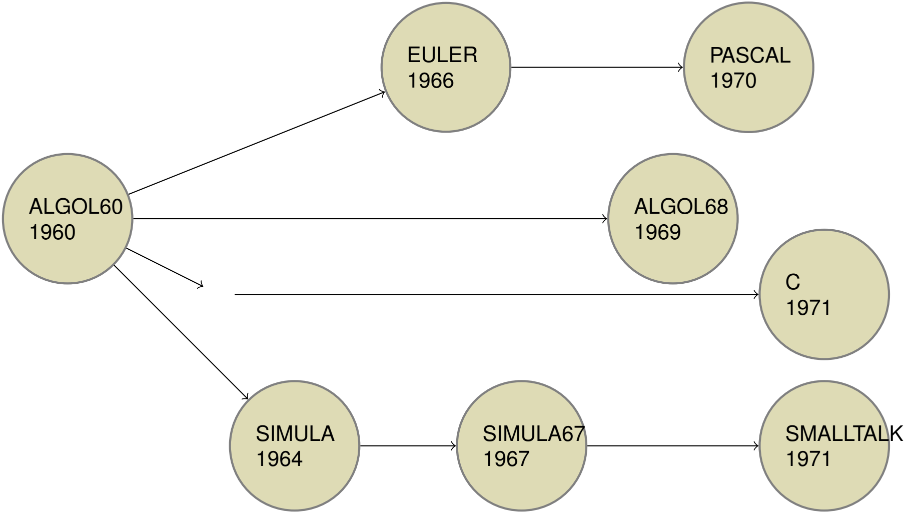
## Orthogonality

- Headed by Van Wijngaarden
- Create the conceptual best programming language
- Enormous
- (1969) ALGOL 68

## Pragmatism

- Headed by Wirth, Hoare
- Create an ALGOL 66: ready for use
- (1968) Minority report
- Wirth’s languages: Euler, ALGOL W, and PASCAL

# ALGOL 60's Ancestral Tree of Languages



# Why Did ALGOL Die?

- In 1960 ALGOL 60 was on the agenda of mathematics
- Soon it became a vessel for a new agenda: a new science
- For computer scientists, ALGOL 60 was not particular interesting
- Aim: Create a general purpose programming language
- All modern languages inherit from ALGOL 60 and the languages produced by the ALGOL effort

# Outline

1. Introduction
2. 1950s: The Era of the Prototype
3. 1955-1965: The Era of the Algorithmic Language
4. The Birth of a Science
5. 1960s: Onwards to General Purpose Programming Languages
6. Conclusion

Questions, Discussion, or Remarks?