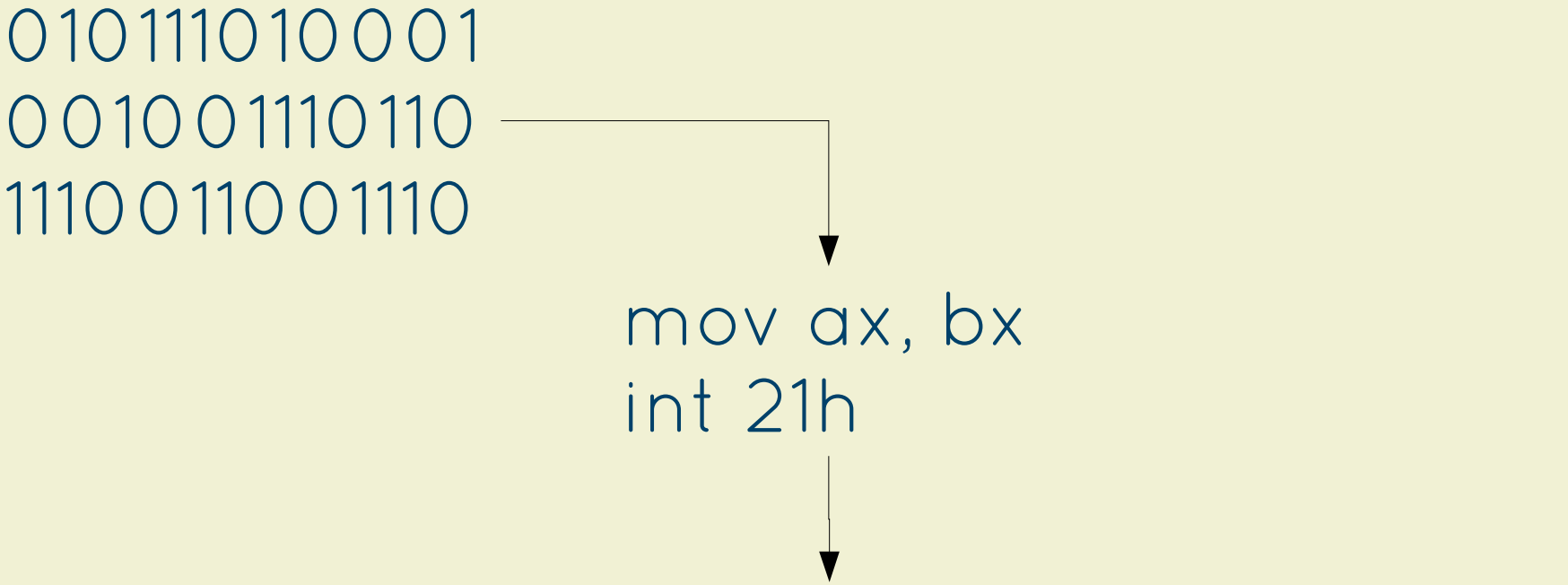
CMSC 124

CONCEPTS OF PROGRAMMING LANGUAGES

CNM PERALTA

A BRIEF HISTORY OF PROGRAMMING LANGUAGES

How did this happen?



We know that we started with **machine language** – 1's and 0's. What then?

The use of **numeric codes** and **absolute addressing** made machine language **error-prone** and **tedious**.

Assembly languages

were developed to address these concerns.

However, most problems required **floating-point arithmetic** and **array indexing**, which were not easily implemented using assembly.

The first high-level programming language to be **designed** was called

Plankalkül.

Designer:

Konrad Zuse

Year Designed:

~1943-1945

Year Implemented:

1998

http://upload.wikimedia.org/wikipedia/commons/d/da/Konrad_Zuse_%281992%29.jpg

Though fairly complete in design, **statements** were too **complicated**.

FOR EXAMPLE...

In C:

```
A[5] = A[4] + 1
```

In Plankalkül:

```
A + 1 => A
4       5
1.n     1.n
```

Then, in **1954**, the **IBM 704 mainframe computer** was released and, along with it,

FORTRAN.

Designer:

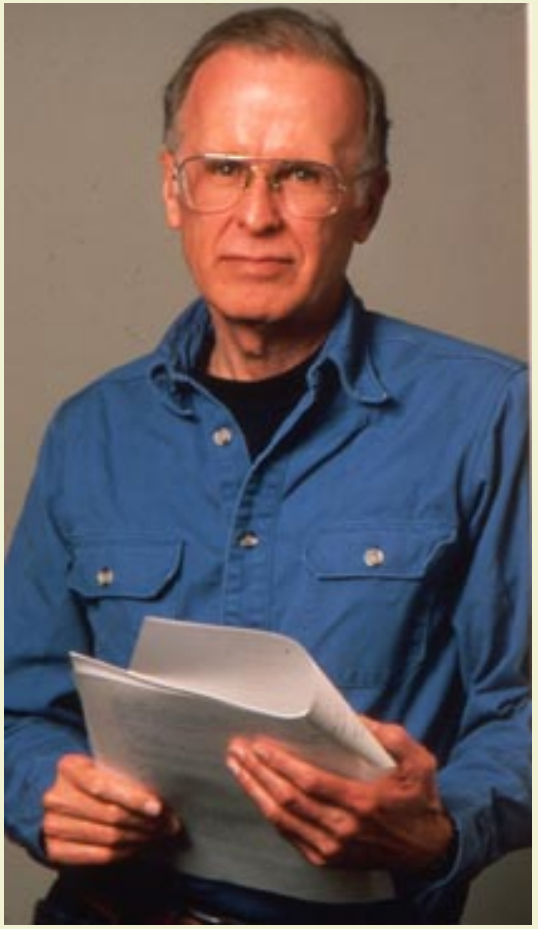
John Backus

Year Designed:

1954

Year Implemented:

1957



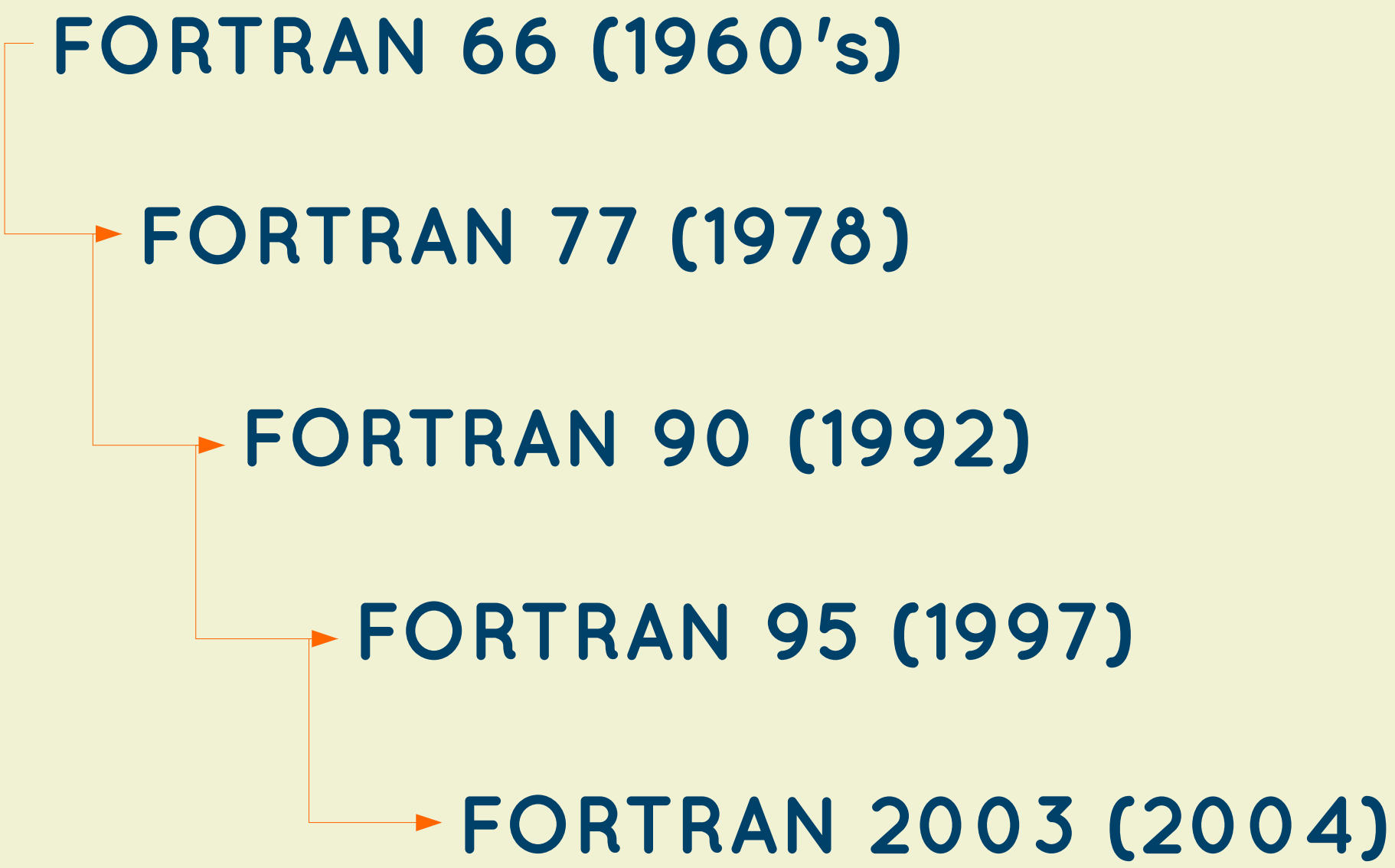
http://informat444.narod.ru/museum/lanr/picture/backus.jpg

http://upload.wikimedia.org/wikipedia/commons/5/51/IBM_logo.svg

Derived from "**for**mula **tran**slating system," FORTRAN was intended for **numeric** and **scientific computing**.

Feature	Assembly	FORTRAN
Floating-point arithmetic	X	√
Array indexing	X	√


FORTRAN I (1957)
FORTRAN II (1958)
FORTRAN IV
aka FORTRAN 66
(1960s)


FORTRAN 66 (1960's)
FORTRAN 77 (1978)
FORTRAN 90 (1992)
FORTRAN 95 (1997)
FORTRAN 2003 (2004)

The latest version of Fortran is **Fortran 2008 (2010)**.

SIGNIFICANCE

FORTRAN IV became one of the **most widely-used** programming languages of its time.

"**Fortran** is the *lingua franca* of the computing world...And it has survived and will survive because it has turned out to be a remarkably useful part of a very vital commerce."

- Alan Perlis, 1978

SIGNIFICANCE

One of the most significant contributions of FORTRAN was the **independent compilation of subroutines**, by **FORTRAN II** in **1958**.

SIGNIFICANCE

Fortran is often credited with being the **first compiled high-level language**.

MEANWHILE...

List processing

was gaining popularity for its usefulness in **artificial intelligence**.

The concept was introduced by **Allen Newell**, **John Clifford Shaw**, and **Herbert Simon** in **1956**.

LISP

was the **first functional programming language**, and was invented for **list processing**.

Among LISP's contributions to PLs are:

- **Tree data structures**
- **Recursion**
- **Dynamic typing**

The first LISP was called **pure LISP**, and was implemented in **1958** by **John McCarthy**.

LISP notation was based on

Lambda calculus.

Designer:

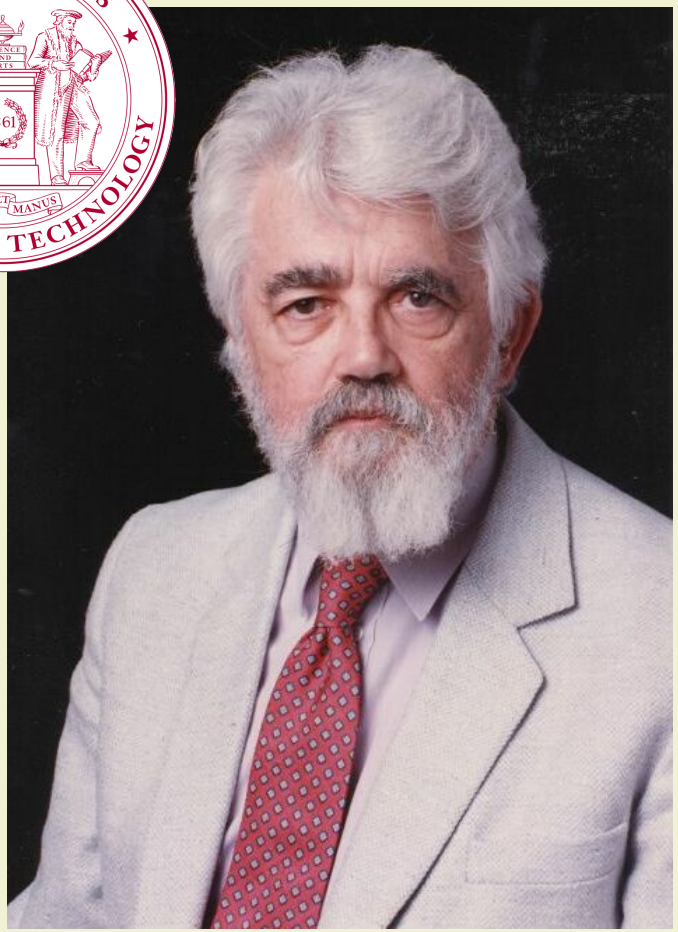
John McCarthy

Year Designed:

1958

Year Implemented:

1960



http://www-formal.stanford.edu/jmc/jmccolor.jpg

http://www.irononstickers.net/images/2012/09/10/Massachusetts%20Institute%20of%20Technology%20Logo.png

LISP spawned many dialects, the most commonly used being:

SCHEME (1970s)

and

Common LISP (1994)

PROBLEM!

Programming languages were **machine-dependent**.

Numerous computing organizations proposed the development of a **machine-independent** programming language.

Zurich, Switzerland

http://upload.wikimedia.org/wikipedia/en/8/8e/Association_for_Computing_Machinery_%28ACM%29_logo.svg

http://4vector.com/i/free-vector-tu-airplane-clip-art_108958_Tu_Airplane_clip_art_hight.png

http://upload.wikimedia.org/wikipedia/commons/0/08/Flag_of_Switzerland_%28Pantone%29.svg

http://upload.wikimedia.org/wikipedia/de/thumb/5/55/Gesellschaft_f%C3%BCr_Angewandte_Mathematik_und_Mechanik_Logo.svg/220px-Gesellschaft_f%C3%BCr_Angewandte_Mathematik_und_Mechanik_Logo.svg.png

1958

GOALS

1.

Readable programs with syntax close to **mathematical notation**

2.

Can **describe** **algorithms** in printed publications

3.

Mechanically-translatable to machine language

Originally called the

International Algorithmic Language (IAL), the result was eventually called

ALGOL.

ALGOL basically **generalized** concepts introduced by **FORTRAN**.

The language designed in the 1958 meeting was called

ALGOL 58.

DESIGNERS

Friedrich
Ludwig Bauer

GAMM

DESIGNERS

Hermann Bottenbruch

GAMM

DESIGNERS

Heinz
Rutishauser

GAMM

DESIGNERS

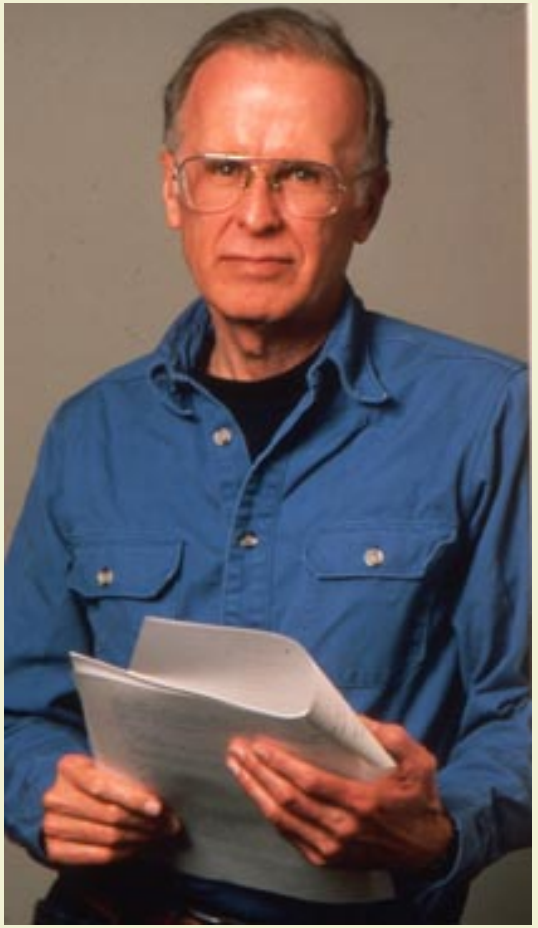
Klaus
Samelson

GAMM

DESIGNERS

John Backus

ACM



DESIGNERS

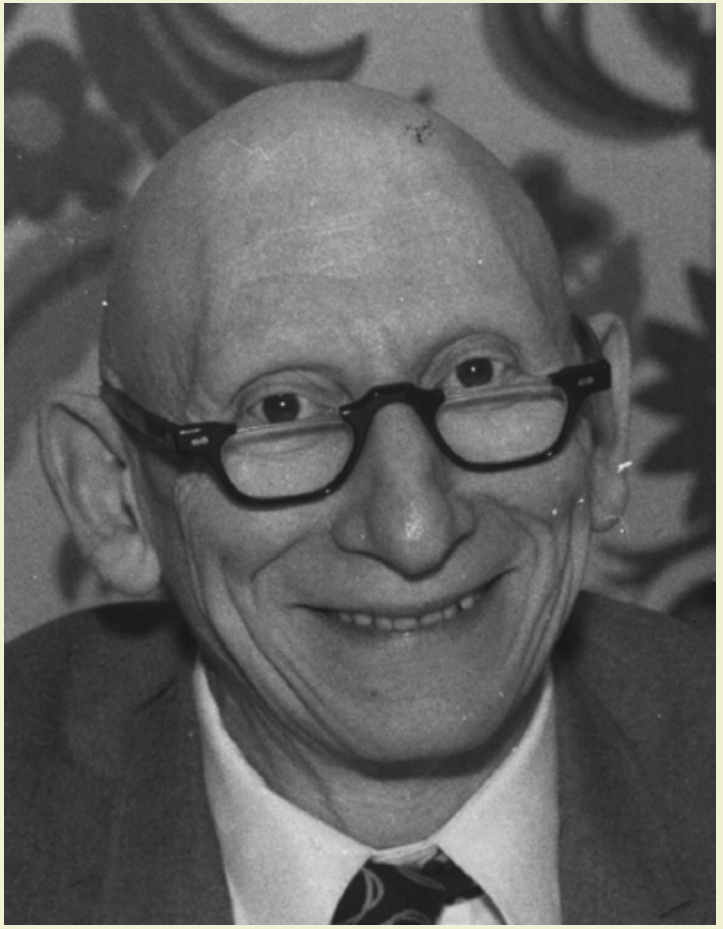
Charles Katz

ACM

DESIGNERS

Alan Perlis

ACM



DESIGNERS

Joseph Henry Wegstein

ACM

SOME CONCEPTS INTRODUCED BY ALGOL 58

1.

Formal **data types**

2.

Compound statements

3.

Nested selection statements

The most notable implementation of the ALGOL 58 design was **JOVIAL**.

JOVIAL

Jules Own Version of the International Algorithmic Language

Designer:

Jules Schwarz

Year Implemented:

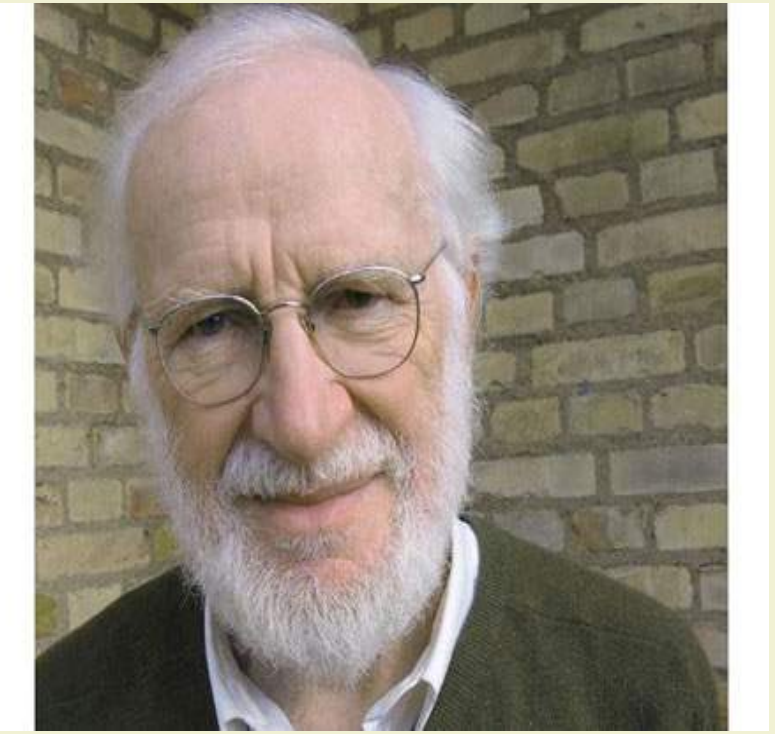
1960

ALGOL was revised in **1960** to produce **ALGOL 60**.

DESIGNERS

Peter Naur

Europe



DESIGNERS

Bernard Vauquois

Europe

DESIGNERS

Adriaan van Wijngaarden

Europe

DESIGNERS

Michael Woodger

Europe

DESIGNERS

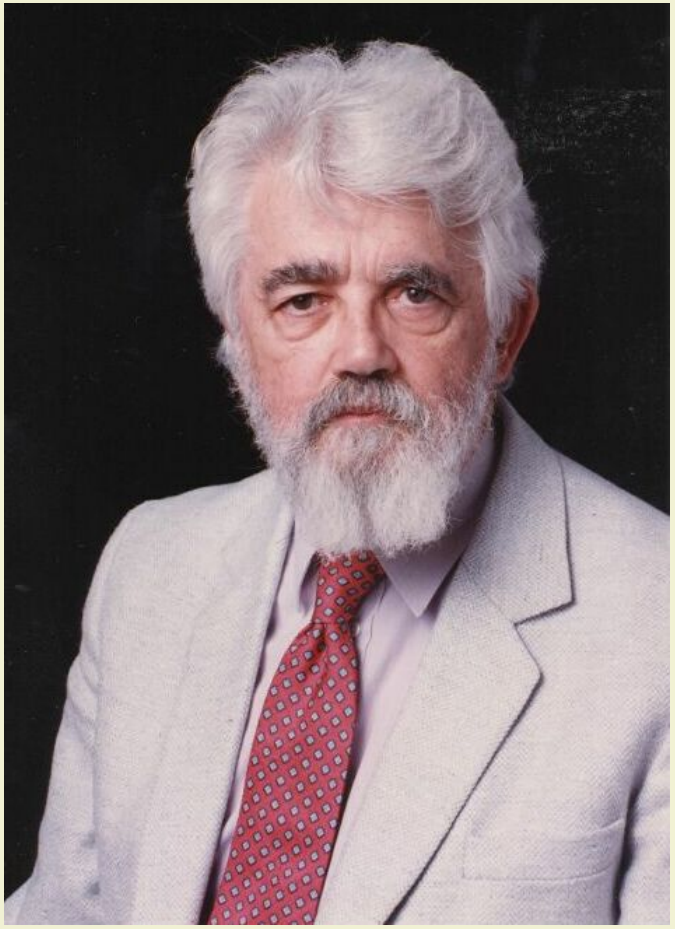
Julien Green

USA

DESIGNERS

John McCarthy

USA



SOME CONCEPTS INTRODUCED BY ALGOL 60

1.

Block structure

```
int main() {
  //This is a block
}
```

in C

in Pascal

```
procedure Print(var j : integer);
begin
…
end
```

2.

Pass-by-value

and

pass-by-name

ALGOL 60 was defined using **Backus' Normal Form (BNF)**

Despite the effort put into designing ALGOL, it **never became popular**.

BUT WHYYYYYY?

1.

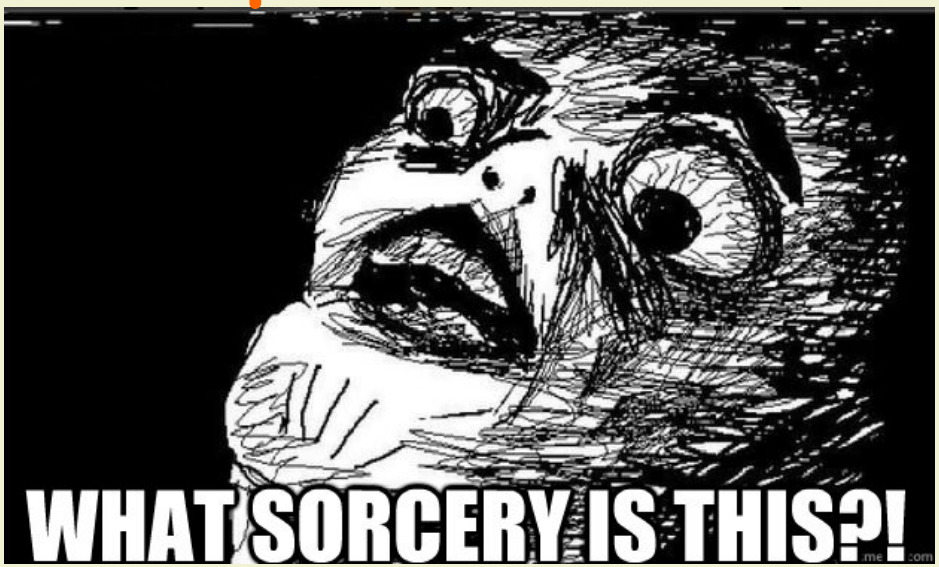
Too flexible

http://www.blogging4jobs.com/wp-content/uploads/2013/09/workplace_flexibility.jpg

2.

```
...
<program> ::= <block> | <compound
statement>
<block> ::= <unlabelled block> | <label>:
<block>
<unlabelled block> ::= <block head> ;
<compound tail>
<block head> ::=
begin
<declaration> | <block head> ;
<declaration>
<compound statement> ::= <unlabelled
compound> | <label>: <compound statement>
<unlabelled compound> ::=
begin
<compound tail>
<compound tail> ::= <statement>
end
| <statement> ; <compound tail>
...
```

BNF was considered **strange** and **complicated**



3.

Entrenchment of **FORTRAN**; lack of support from IBM

MOVING ON...

"**Mathematical programs** should be written in **mathematical notation**, **data processing programs** should be written in **English statements**."

- **Grace Hopper, December 1953**

On **May 28-29, 1959**, the **Department of Defense** held a meeting to create a common PL for **business applications**.

DESIGN CONSIDERATIONS

1.

Use as much **English** as possible

2.

Prioritize ease-of-use over power

The resulting language was called

COBOL (Common Business-Oriented Language) 60

Designer(s):

US DoD

Year Designed:

1959

Year Implemented:

1960

DESIGNERS

Grace Hopper Jean E. Sammet

COBOL became the **first PL** to be **standardized** by the **American National Standards Institute (ANSI)**

Strength: Data Division

COBOL was the first language to implement **hierarchical structures**

```
01 ADDRESS.
  05 ADDRESS-LINE-1 PIC
X(40).
  05 ADDRESS-LINE-2.
    10 CITY PIC X(17).
    10 STATE PIC XX.
    10 FILLER PIC X.
    10 ZIP1 PIC 9(5).
    10 FILLER PIC X VALUE IS
"-".
    10 ZIP2 PIC 9(4).
```

Weakness: Procedure Division

Until the **1974 standard**, there was **no parameter passing** in COBOL.

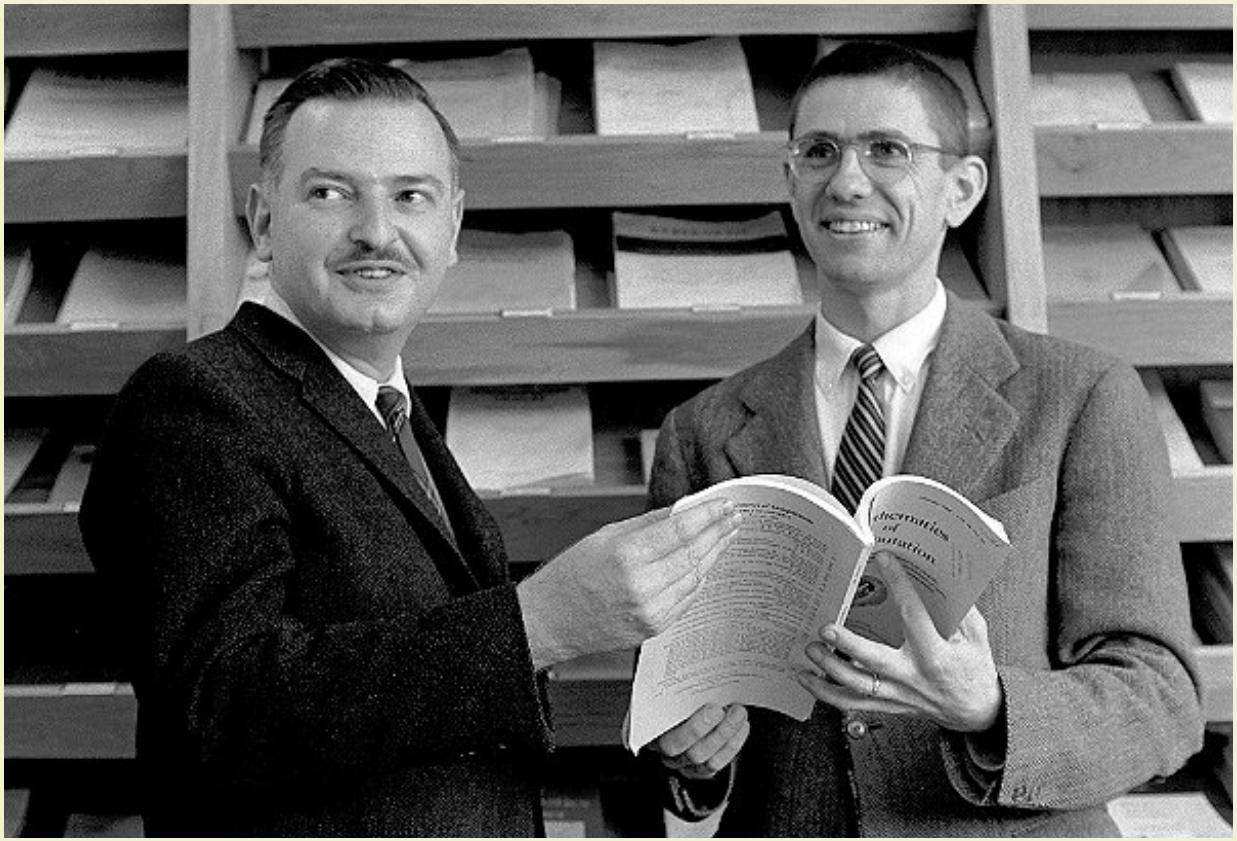
COBOL was widely-used partly because it was **mandated by the US DoD.**

COBOL has evolved as a language, introducing **OOP** in **2002**, and with a **2014 update** coming soon.

On the whole, the languages that we have discussed have been intended for use in a **professional capacity**.

AND THEN...

In **1963**, **John Kemeny** and **Thomas Kurtz** of **Dartmouth College** developed a language for **liberal arts students**.



Designers:

John Kemeny and Thomas Kurtz

Year Designed:

1964

Year Implemented:

1964

Since the language was intended for non-science students, its **main goals** were **ease-of-use** in **usability in doing homework**, even at the expense of computer time.

The language was called **BASIC**, which stands for **B**eginner's **A**ll-purpose **S**ymbolic **I**nstruction **C**ode.

BASIC became the first widely-used language to use

time-sharing.

Time-sharing

allows multiple users to share a computing resource.

Though its **features were limited**, BASIC was very **easy to learn**.

AT AROUND THE SAME TIME...

An **overlap** was developing between **scientific computing** and **business applications**.

Thus, a language that could be used on **more than one application area** was needed.

Developed by IBM (again) and originally called **FORTRAN VI**, it was eventually released as **Programming Language One** or

PL/I.

(Before that, it was NPL – New Programming Language.)

PL/I was influenced by COBOL, FORTRAN and ALGOL.

CONCEPTS INTRODUCED BY PL/I

1.

Concurrently-executing subprograms

2.

Exception detection

and

Run-time error handling

3.

Pointers

MEANWHILE, IN NORWAY...

Kristen Nygaard and **Ole-Johan Dahl** started developing a language to **simulate discrete event systems** from **1962-1964**.

They named it

SIMULA I.

Designers:

Kristen Nygaard and Ole-Johan Dahl

Years Designed:

1962-1964

Year Implemented:

1965

SIMULA I was revised in 1967 to yield

SIMULA 67.

Heavily influenced by ALGOL 60, SIMULA 67 also introduced

co-routines.

Co-routines

are subprograms that **restart** at the **position** where they **previously stopped**.

Co-routines were implemented using the

class construct.

Classes introduced the concept of **data abstraction**, which is the **foundation** of **object-oriented programming**.

Thus, **SIMULA 67** is **sometimes considered** the **first object-oriented language**.

Class-related concepts such as **objects**, **inheritance** and **subclasses** were also introduced by SIMULA 67.

AND THEN, IN 1968...

Adriann van Wijngaarden just wouldn't give up on **ALGOL** and revised it in **1968**.

The aptly-named **ALGOL 68** was designed with a particular criterion in mind:

Orthogonality

Orthogonality

allows a **small number of primitive constructs** to be **combined** in a **small number of ways** to build control and data structures.

ALGOL 68's unprecedented support for orthogonality allowed

user-defined data types.

Also introduced by ALGOL 68 were

implicit-heap dynamic arrays.

ALGOL 68 eclipsed neither **FORTRAN** nor **PL/I**, partially because of **IBM**'s support for the latter two languages.

Despite this, **ALGOL 68 significantly influenced** many **imperative languages** that are still used today.

Two of the most notable descendants of ALGOL are:

PASCAL

and

C.

Pascal was designed by **Niklaus Wirth** in **1970** specifically for **instructional purposes**.

Designer:

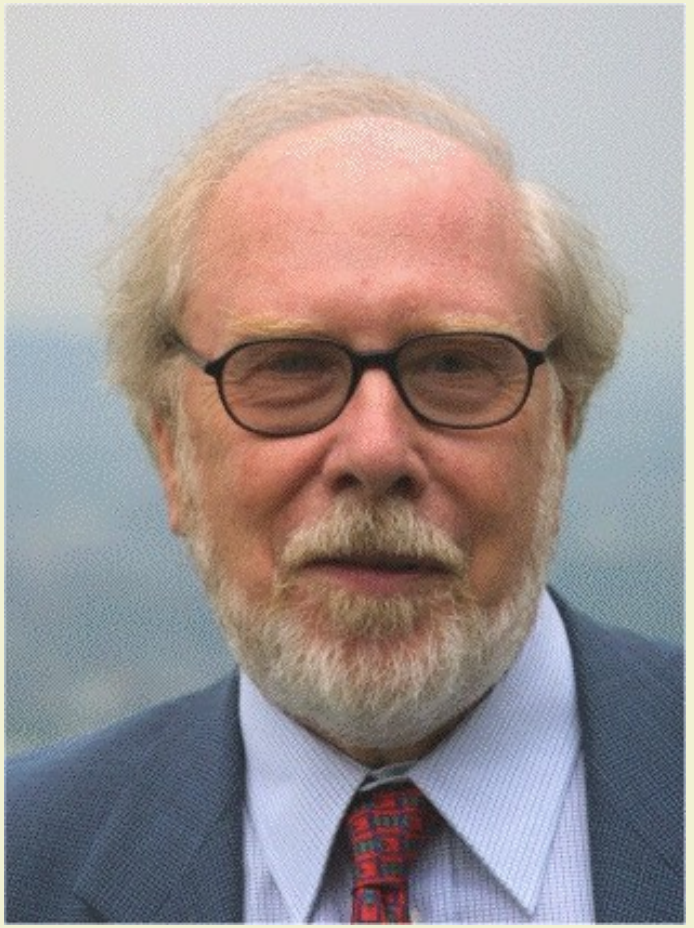
Niklaus Wirth

Years Designed:

1968-1969

Year Implemented:

1970



Pascal became **popular** in the **mid-1970s** because of its **simplicity** and **expressiveness**.

C was designed by **Dennis Ritchie** (of Bell Laboratories) in **1972**.

Designer:

Dennis Ritchie

Years Designed:

1969-1973

Year Implemented:

1972

The main purpose for C was

systems programming.

C, along with **assembly language**, was used in the development of the **UNIX operating system**.

One of the most prominent problems of C was its

lack of type checking.

Aside from ALGOL 68, **C** is also **descended** from **CPL** (**early 1960s**), **BCPL** (**1967**), and **B** (**1970**).

AND THEN...

FRANCE, 1972

Prolog was developed by **Alain Colmerauer** and **Philippe Roussel**.

Designer:

Alain Colmerauer

Year Designed:

1972

Year Implemented:

1972

Roussel was involved in the **implementation** of the first Prolog interpreter.

Prolog

≈

Programming **Log**ic

Prolog is rooted in **first-order (predicate) logic** or **predicate calculus**.

Logic programming does **not specify how to compute** a result.

Instead, it **describes** the **form** and/or **characteristics** of the result.

Logic programming never really gained a wide user-base because it was **highly inefficient** and applied to a **small number of problem areas**.

US DOD, 1974

There became an overwhelming number of languages in use for **embedded systems.**

Embedded systems

were programs installed on computer hardware that was embedded in devices to be controlled.

Because of the sheer number of **different contractors** for **different devices** in use in the US DoD, there were **more than 450 languages** in use at the time.

There was virtually
no reusabilty of code.

This problem resulted in the **single, most extensive and expensive language design effort** ever undertaken.

Language design began in **1975** and continued until **1980**, led by **Jean Ichbiah**.

Designer:

Jean Ichbiah

Year Designed:

1975-1980

Year Implemented:

1983

The language was called **Ada**, after **Ada Lovelace**, who is considered the **first computer programmer**.

ADA HAD FOUR MAJOR CONTRIBUTONS

1.

Support for **packages** (furthered support for data abstraction)

2.

Extensive exception handling, allowing **users to gain control** when an error is encountered,

3.

Generic program units, e.g., procedures whose parameters have unspecified data types.

4.

Concurrency support with the rendezvous mechanism.

The development and subsequent use of Ada **embodied** the **concept of software engineering** at the time.

Although Ada eventually became moderately successful, its **first usable compiler** came out in **1985** because the language was too **large** and **complicated**.

Ada is still in active development, with the **most recent release** in **2012** (**Ada 2012**).

The Ada **user base declined** with **Ada 95** because the **US DoD stopped requiring it** and because of the rise of **C++**.

SIMULA AND THE RISE OF OOP

SIMULA 67 directly influenced the design of what would be the **first programming language to fully support object-oriented programming**.

The language was called **Smalltalk** and was first publicly available as **Smalltalk 80** in **1980**.

Though designed by numerous people in the **Xerox Palo Alto Research Center**, the most prominent designer of Smalltalk is **Alan Kay**.

Designer:

Alan Kay

Years Designed:

1969-1972

Year Implemented:

1972

1980 (public)

Smalltalk made **everything an object**, and operations were done using **message passing**.

Aside from its **impact** on **OOP**, Smalltalk also promoted the use of **graphical user interfaces**.

Following the release of Smalltalk, **Bjarne Stroustrup modified C** to **support OOP** to come up with

C++.

Designer:

Bjarne Stroustrup

Year Designed:

1979-1983

Year Implemented:

1983

C++ attempted to fix C's problems with **type checking** and **conversion**, and implemented **method** and **operator overloading**.

Also, unlike most OOPLs today, C++ supports **multiple inheritance**.

Although categorized as an OOPL, C++ also **supports procedural programming**; it supports **methods** and **functions** separately.

C++'S POPULARITY IS THANKS TO SEVERAL FACTORS

1.

Good compilers are inexpensive.

2.

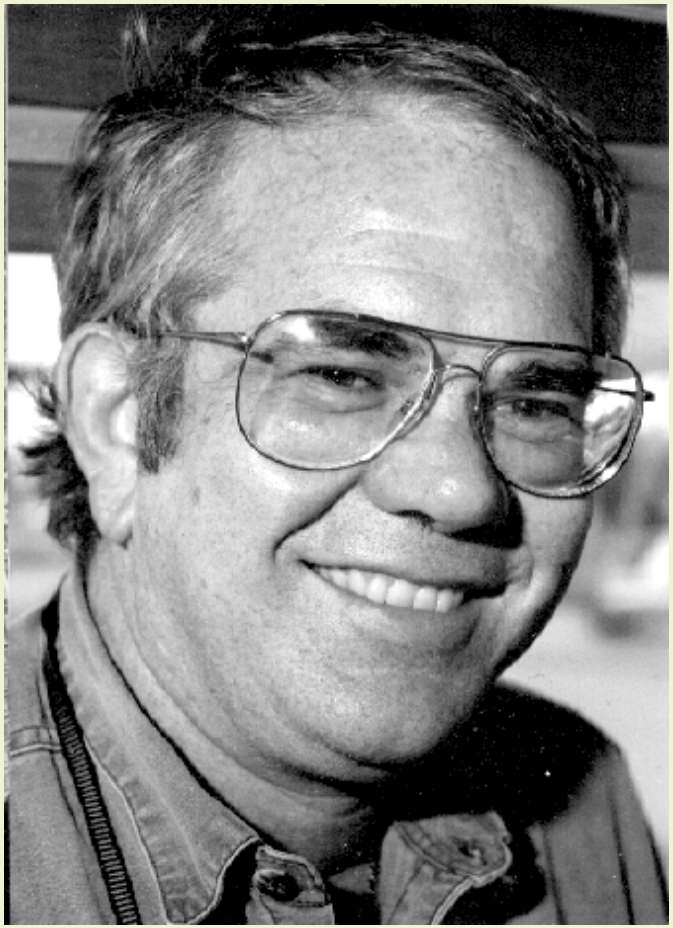
Backward-compatible with C.

3.

Its **applicability to large software projects** was unmatched at the time.

Being a **descendant of C**, C++ is still considered **unsafe**; Java and Ada are considered safer.

Another prominent C **descendant** is **Objective-C**, the **main programming language** for **Mac OSX** and **iOS** (Apple).



Designers: Brad Cox and Tom Love
Year Designed: 1981-1983
Year Implemented: 1983

However, **C and its descendants** were still generally considered **unsafe**, especially for **embedded systems** for **consumer electronics**.

This led **Sun Microsystems** to develop a language for this purpose. Under the leadership of **James Gosling**,

Java was born in **1995**.

Designer:

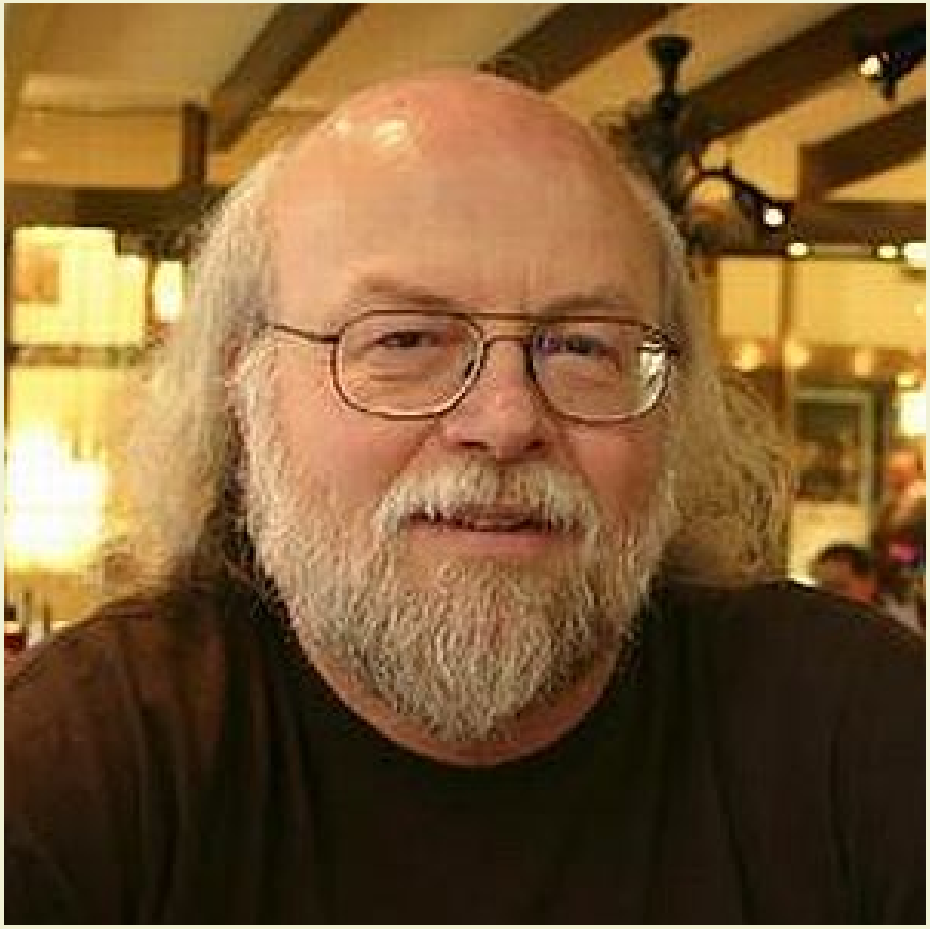
James Gosling

Year Designed:

1991-1995

Year Implemented:

1995



However, though originally intended for embedded systems, Java became **popular for web programming**.

WHYYYYYY?

The advent of graphical web browsers and Java's applets.

Though based on C++, Java was **simpler**, **smaller** and **more reliable**.

SOME DIFFERENCES BETWEEN JAVA AND C++

1.

No pointers.

2.

Garbage collection (implicit storage deallocation)

3.

Support for **single inheritance** only, but simulation of multiple inheritance using **interfaces**.

Java is **extremely portable** but at the **cost of efficiency**; Java programs could be **up to 10 times slower** than equivalent C programs.

Regardless, Java's **popularity grew faster than any other programming language**.

Of course, never one to be left behind, **Microsoft** created its own **C++ spin-off**.

Called **Visual C#**, it was released together with the **.NET platform** in **2002**.

Designed by: Anders Hejlsberg
Years Designed: 1999-2002
Year Implemented: 2002

A BRIEF LOOK AT SCRIPTING LANGUAGES

Scripting languages

were originally made to **automate tasks** involving calls to system subprograms.

Eventually, however, the **lines between scripting and programming languages blurred** with the former supporting concepts of the latter.

SOME NOTABLE SCRIPTING LANGUAGES

Larry Wall
Perl

Brendan Eich
Javascript

Rasmus Lerdorf
PHP

Guido van Rossum
Python

Yukihiro Matsumoto
Ruby