Computer Languages

Arden Rasmussen

January 5, 2018

Abstract

This document aims at compairing a large number of the posible different programming languages that are currently avalible, for both their efficiency, code size, support, readability and many other features. This document does not aim to single out a single language as the best or worst, but to mearly show the variations between the languages, and to provide some introduction to the languages.

Contents

1 Ada

`Ada` is a structured, statically types, imperative, wide-spectrum, and object-oriented high-level computer programming language, extended from `sec:pascal` and other Languages. It has built-in language support for design-by-contract, extremely strong typing, explicit concurrency, offering tasks, synchronous message passing, protected objects, and non-determinism. `Ada` improves code safety and maintainability by using the compiler to find errors in favor of runtime errors. `Ada` is an international standard; the current version is defined by ISO/IEC 8652:2012.

`Ada` was originally designed by a team lead by Jean Ichbiah of CII Honeywell Bull under contract to the United States Department of Defense (DoD) from 1977 to 1983 to supersede over 450 programming languages used by the DoD at that time. `Ada` was named after Ada Lovelace (1815–1852), who has been credited with being the first computer programmer.

1.1 Influence

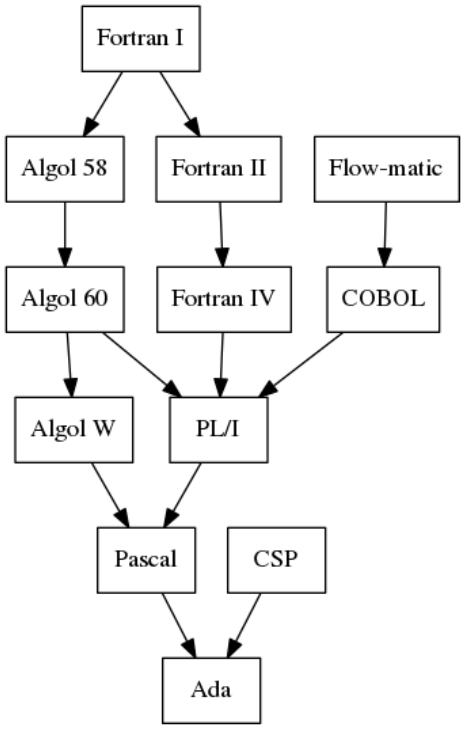

Figure 1: Inheritance diagram for `Ada`.

`Ada` was primarily influenced by the languages `Algol`, `Pascal`, `C++`, `Smalltalk`, `Modula-2`, `Java`, and `Eiffel`.

1.2 Hello World

```
with Ada.Text_IO; use Ada.Text_IO;
procedure Hello is
begin
  Put_Line ("Hello, world!");
end Hello;
```

2 Bash

`Bash` is a Unix shell and command languages written by Brian Fox for the GNU Project as a free software replacement for the Bourne shell. First released in 1989, it has been distributed widely as the default login shell for most Linux distributions and Apple's macOS. A version is also available for Windows 10.

`Bash` is a command processor that typically runs in the text window, where the user types commands that cause actions. `Bash` can also read and execute commands from a files, called a script. Like Unix shells, it supports filename globbing (wildcard matching), piping, here documents, command substitution, variables, and control structures for condition-testing and iteration. The keywords, syntax and other basic features of the language are all copied from `sh`. Other features, e.g., history are copied from `csh`, and `ksh`. `Bash` is a POSIX-compliant shell, but with a number of extensions.

The shell's name is an acronym for *Bourne-again shell*, punning on the name of the Bourne shell that it replaces and on the term "born again" that denotes spiritual rebirth in contemporary American Christianity.

A security hole in `Bash` dating from version 1.03, dubbed Shellshock, was discovered in early September 2014 and quickly led to a range of attacks across the Internet. Patches to fix the bugs were made available soon after the bugs were identified, but not all computers have been updated.

2.1 Influence

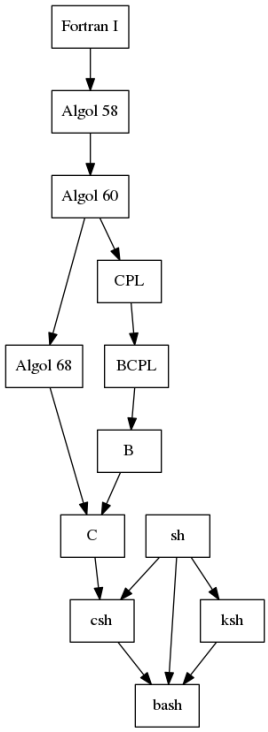

Figure 2: Inheritance diagram for `Bash`.

2.2 Hello World

```
#!/bin/bash
printf "Hello, world!\n"
```

3 C

`C` is a general-purpose, imperative computer programming languages, supporting structured programming, lexical variable scope and recursion, while a static type system prevents many unintended operations. By design, `C` provides constructs that map efficiently to typical machine instructions, and therefore it has found lasting use in applications that had formerly been coded in assembly languages, including operation systems, as well as various application software for computers ranging from supercomputers to embedded systems.

`C` was originally developed by Dennis Ritchie between 1969 and 1973 at Bell Labs, and used to re-implement the Unix operation system. It has since becomes one of the most widely used programming languages of all time, with `C` compilers from various vendors available for the majority of existing computer architectures and operating systems. C has been standardized by the American National Standards Institute (ANSI) since 1989 and subsequently by the International Organization for Standardization (ISO).

`C` is an imperative procedural language. It was designed to be compiled using a relatively straightforward compiler, to provide low-level access to memory, to provide language constructs that map efficiently to machine instructions, and to require minimal run-time support. Despite its low-level capabilities, the language was designed to encourage cross-platform programming. A standards-compliant and portably written `C` program can be compiled for a very wide variety of computer platforms and operating systems with few changes to its source code. The language has become available on a very wide range of platforms, from embedded micro controllers to supercomputers.

3.1 Influence

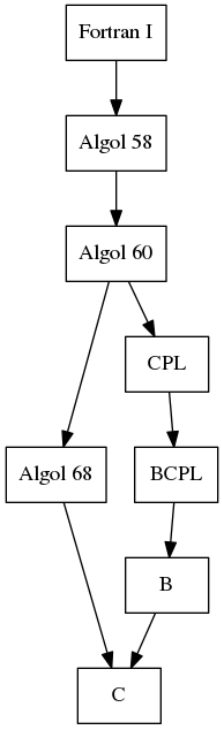

Figure 3: Inheritance diagram for `C`.

`C` was primarily influenced by the languages of `B`, `Algol`, `Assembly`, and `Fortran`.

3.2 Hello World

```c
#include <stdio.h>

int main(void)
{
  printf("Hello, world!\n");
  return 0;
}
```

3.3 Syntax

3.3.1 Basic Syntax

Semicolons In C the semicolon is a statement terminator. So every statement must be ended with a semicolon.

```
printf("Hello, world!\n");
return 0;
```

Comments Comments begin with a /* and end with a */.

```
/* A comment in C */
```

Identifiers Identifiers can start with any character A to Z, a to z, or _, and can be followed by any characters that are not @, $, %. The identifiers are case sensitive.

3.3.2 Data Types

Basic Types

Integer Types

char	1 byte, -128 to 127 or 0 to 255.
unsigned char	1 byte, 0 to 255.
signed char	1 byte, -128 to 127.
int	2 or 4 bytes, -32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
unsigned int	2 or 4 bytes, 0 to 65,535 or 0 to 4,294,967,295

4 C++

C++ is a general-purpose programming languages. It has imperative, object-oriented and generic programming features, while also providing facilities for low-level memory manipulation.

It was designed with a bias toward system programming and embedded, resource-contained and large systems, with performance efficiency and flexibility to use as its design highlights. C++ has also been found useful in many other contexts, with key strengths being software infrastructure and resource constrained applications, including desktop applications, servers (e.g. e-commerce, web search, or SQL servers), and performance-critical applications (e.g. telephone switches or space probes). C++ is a compiled languages, with implementations of it available on many platforms. Many vendors provide C++ compilers, including the Free Software Foundation, Microsoft, Intel, and IBM.

C++ is standardized by the International Organization for Standardization (ISO), with the latest standard version ratified and published by ISO in December 2017 as ISO/IEC 14882:2017 (informally known as C++17). The C++ programming languages was initially standardized in 1998 as ISO/IEC 14882:1998, which was then amended by the C++03, C++11, and C++14 standards. The current C++17 standard supersedes these with new features and an enlarged standard library. Before the initial standardization in 1998, C++ was developed by Bjarne Stroustrup at Bell Labs since 1979, as an extension of the C languages as he wanted an efficient and flexible language similar to C which also provided high-level features for program organization. C++20 is the next planned standard thereafter.

Many other programming languages have been influenced by C++, including C#, D, Java, and newer versions of C.

4.1 Influence

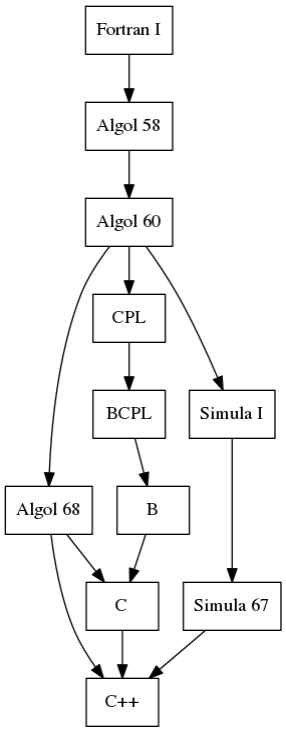

Figure 4: Inheritance diagram for `C++`.

`C++` was primarily influenced by the languages `Ada`, `Algol`, `C`, `CLU`, `ML`, `Simula`, and `Python`.

4.2 Hello World

```
#include <iostream>

int main() {
  std::cout << "Hello, world!\n";
}
```

4.3 Syntax

5 C#

`C#` is a multi-paradigm programming language encompassing strong typing, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft within it **.NET** initiative and later approved as a standard by Ecma (ECMA-334) and ISO (ISO/IEC 23270:2006). `C#` is one of the programming languages designed for the Common Language Infrastructure.

`C#` is a general-purpous, object-oriented programming language. Its development team is lead by Anders Hejlsberg. The most recent verseion is `C# 7.2`, which was released in 2017 along with Visual Studio 2017 version 15.5.

5.1 Influence

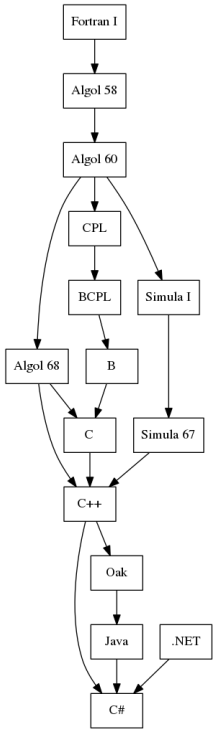

Figure 5: Inheritance diagram for `C#`.

6 Go

`Go` (often referred to as `golang`) is a programming language created at Google in 2009 by Robert Griesemer, Rob Pike, and Ken Thompson. It is a compiled, statically typed language in the tradition of `sec:algol` and `sec:c`, with garbage collection, limited structural typing, memory safety features and CSP-style concurrent programming features added. The compiler and other language tools originally developed by Google are all free and open source.

6.1 Influence

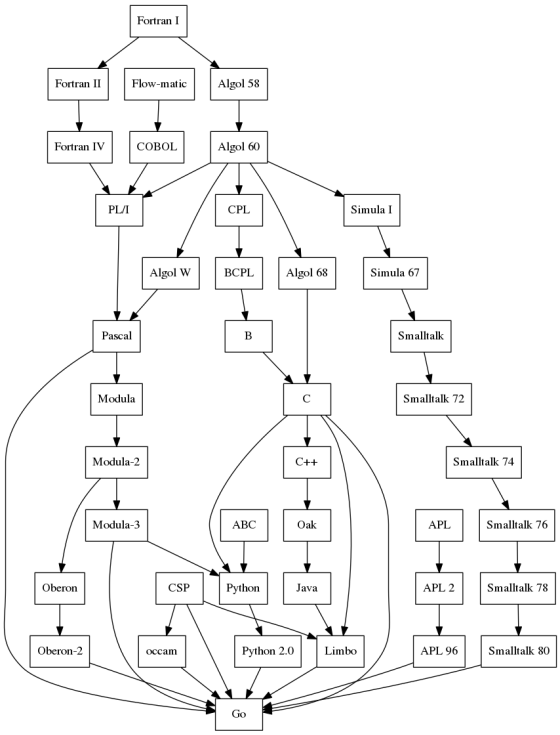

Figure 6: Inheritance diagram for `Go`.

7 Java

`Java` is a general-purpose computer programming language that is concurrent, class-based, object oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled `Java` code can run on all platforms that support `Java` without the need for recompilation. `Java` applications are typically compiled to byte code that can run on any *Java virtual machine* (JVM) regardless of computer architecture. As of 2016, `Java` is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. `Java` was originally developed by James Gosling at Sum Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sum Microsystems' Java platform. The languages derives much of its syntax from `C` and `C++`, but it has fewer low-level facilities than either of them.

The original and reference implementation `Java` compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sum relicensed most of its `Java` technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for `Java` (byte code compiler), GNU ClassPath (standard libraries), and IcedTea-Web (browser plugin for applets).

The latest version is `Java 9`, released on September 21, 2017, and is one of the two version currently supported for free by Oracle. Versions earlier than `Java 8` are supported by companies on a commercial basis; e.g. by Oracle back to `Java 6` as of October 2017.

7.1 Influence

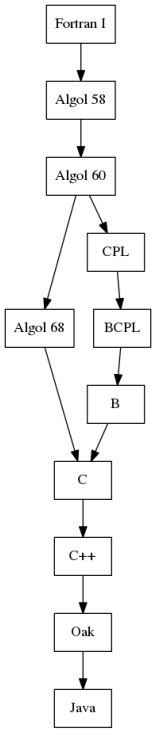

Figure 7: Inheritance diagram for `Java`.

8 JavaScript

`JavaScript`, often abbreviated as `JS`, is a high-level, dynamic, weakly typed, prototype-based, multi-paradigm, and interpreted programming language. Alongside `HTML` and `CSS`, `JavaScript` is one of the three core technologies of the world wide web content production. It is used to make web pages interactive and provide online programs, including video games. The majority of websites employ it, and all modern web browsers support it without the need for plug-ins by means of a build in `JavaScript` engine. Each of the many `JavaScript` engines represent a different implementation of `JavaScript`, all based on the ECMAScript specification, with some engines not supporting the spec fully, and with many engines supporting additional features beyond ECMA.

As a multi-paradigm language, `JavaScript` supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles. It has an API for working with text, arrays, dates, regular expressions, and basic manipulation of the DOM, but the language itself does not include any I/O, such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded.

Initially only implemented client-side in web browsers, `JavaScript` engines are now embedded in many other types of host software, including server-side in web server and databases, and in non-web programs such as word processors and PDF software, and in runtime environments that make `JavaScript` available for writing mobile and desktop applications, including desktop widgets.

Although there are strong outward similarities between `JavaScript` and `Java`, including language name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design; `JavaScript` was influenced by programming languages such as `Self` and `Scheme`.

8.1 Influence

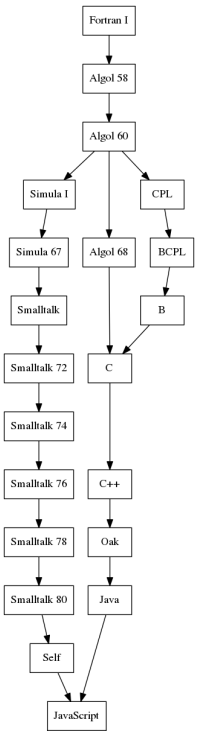

Figure 8: Inheritance diagram for `JavaScript`.

9 Lisp

`Lisp` (historically, `LISP`) is a family of computer programming languages with a long history and distinctive, fully parenthesized prefix notation. Originally specified in 1958, `Lisp` is the second-oldest high-level programming languages in widespread use today. Only `Fortran` is older, by one year. `Lisp` has changed since its early days, and many dialects have existed overs its history. Today, the best known general-purpose `Lisp` dialects are *Common Lisp* and *Scheme*.

`Lisp` was originally created as a practical mathematical notation for computer programs, influenced by the notation of Alonzo Church's lambda calculus. It quickly became the favored programming language for artificial intelligence (AI) research. As one of the earliest programming languages, `Lisp` pioneered many ideas in computers sciences, including tree data structures, automatic storage management, dynamic typing, conditionals, higher-order functions, recursion, the self-hosting compiler, and the read-eval-print loop.

The name *`LISP`* derives form "LISt Processor". Linked lists are one of `Lisp`'s major data structures, and `Lisp` source code is made of lists. Thus, `Lisp` programs can manipulate source code as a data structure, giving rise to the macro systems that allow programmers to create new syntax or new domain-specific languages embedded in `Lisp`.

The interchangeability of code and data gives `Lisp` its instantly recognizable syntax. All program code is written as s-expressions, or parenthesized lists. A function call or syntactic form is written as a list with the function or operator's name first, and the arguments following; for instance, a function `f` that takes three arguments would be called as `(f arg1 arg2 arg3)`.

9.1 Influence

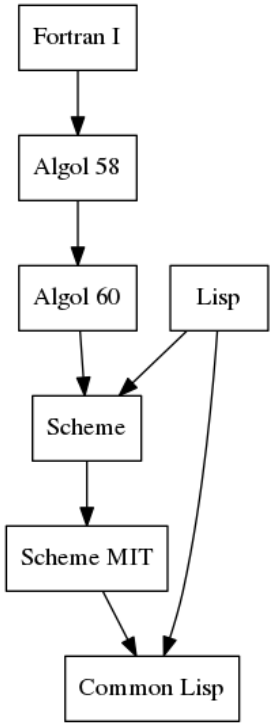

Figure 9: Inheritance diagram for `Lisp`.

10 Lua

`Lua` is a lightweight, multi-paradigm programming language designed primaryily for embedded systems and clients. `Lua` is cross-platform, since the interpreter is written in `C`, and had a relatively simple `C` API.

`Lua` was originally designed in 1993 as a languages for extending software applications to meet the increasing demand for customization at the time. It provided the basic facilities of most procedural programming languages, but more complicated or domain-specific features were not included; rather, it included mechanisms for extending the language, allowing programmers to implement such features. As `Lua` was intended to be a general embeddable extension language, the designers of `Lua` focused on improving its speed,portability, extensibility, and ease-of-use in development.

10.1 Influence

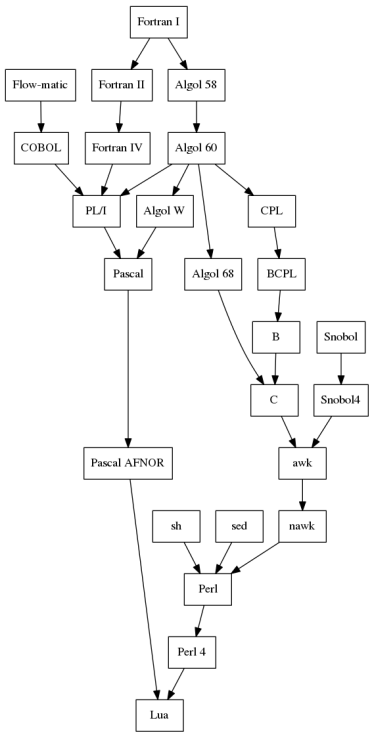

Figure 10: Inheritance diagram for `Lua`.

11 Perl

`Perl` is a family of high-level, general-purpose, interpreted, dynamic programming language. The languages in this family include `Perl 5` and `Perl 6`.

Though `Perl` is not officially an acronym, there are various backronyms in use, including "Practical Extraction and Reporting Language". `Perl` was originally developed by Larry Wall in 1987 as a general-purpose Unix scripting language to make report processing easier. Since then, it has undergone many changes and revisions. `Perl 6` which began as a redesign of `Perl 5` in 2000, eventually evolved into a separate language. Both languages continue to be developed independently by different development teams and liberally borrow ideas from one another.

The `Perl` languages borrow features from other programming languages including `C`, `sh`, `AWK`, and `sed`; Wall also alludes to `Basic` and `Lisp` in the introduction to *Learning Perl* and so on. They provide powerful text processing facilities without the arbitrary data-length limits of many contemporary Unix command line tools, facilitating easy manipulation of text files. `Perl 5` gained widespread popularity in the late 1990s as a CGI scripting language, in part due to its then unsurpassed regular expression and string parsing abilities.

In addition to CGI, `Perl 5` is used for system administration, network programming, finance, bioinformatics, and other applications, such as for GUIs. It has been nicknamed "the Swiss Army chainsaw of scripting languages" because of its flexibility and power, and also its ugliness. In 1998, it was also referred to as the "duct tape that holds the Internet together", in reference to both its ubiquitous use as a glue language and its perceived inelegance.

11.1 Influence

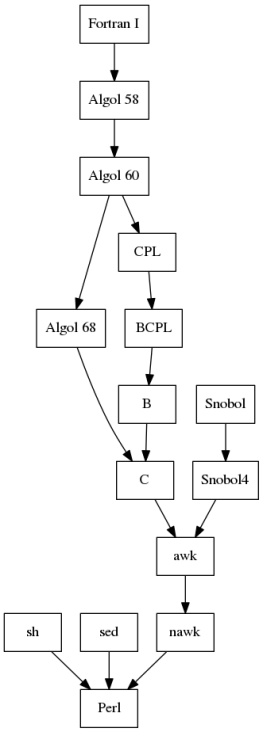

Figure 11: Inheritance diagram for `Perl`.

12 Python 2

`Python` is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, `Python` was a esign philosophy that emphasized code readability, and a syntax that allows programmers to express concepts in fewer lines of code, notably using siginificant whitespae. It provides contructs that enable clear programming on both small and large scales.

`Python` features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

`Python` interpreters are available for many operating systems. `CPython`, the reference implementation of `Python`, is open sourcce software and has a community-based development model, as do nearly all of its variant implmentations. `CPython` is managed by the non-profit Python Software Foundation.

12.1 Influence

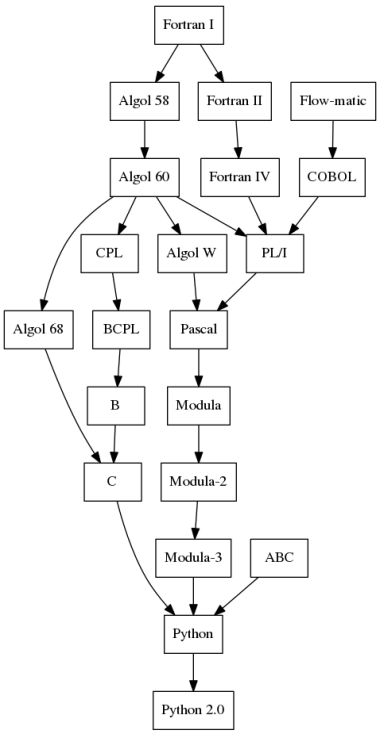

Figure 12: Inheritance diagram for `Python 2`.

13 Python 3

`Python` is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, `Python` was a esign philosophy that emphasized code readability, and a syntax that allows programmers to express concepts in fewer lines of code, notably using siginificant whitespae. It provides contructs that enable clear programming on both small and large scales.

`Python` features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

`Python` interpreters are available for many operating systems. `CPython`, the reference implementation of `Python`, is open sourcce software and has a community-based development model, as do nearly all of its variant implmentations. `CPython` is managed by the non-profit Python Software Foundation.

13.1 Influence

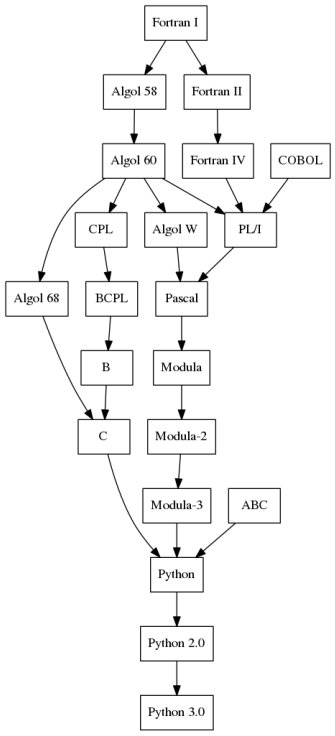

Figure 13: Inheritance diagram for `Python 3`.

14 R

`R` is a free programming language and software enviornment for statistical computing and graphics that is supported by the R Foundation for Statistical Computing. The `R` language is widely used among statisticians and data miners for developing statistical software and data analysis. Polls, surveys of dat aminers, and studies of scholarly literature databases show that `R`'s popularity has increased substantially in recent years.

`R` is a GNU package. The source code for the `R` software environment is written primarily in `C`, `Fortran`, and `R`. `R` is freely available under the GNU General Public License, and pre-compiled binary versions are provided for various operating systems. While `R` has a command line interface, there are several graphical front-ends available.

14.1 Influence

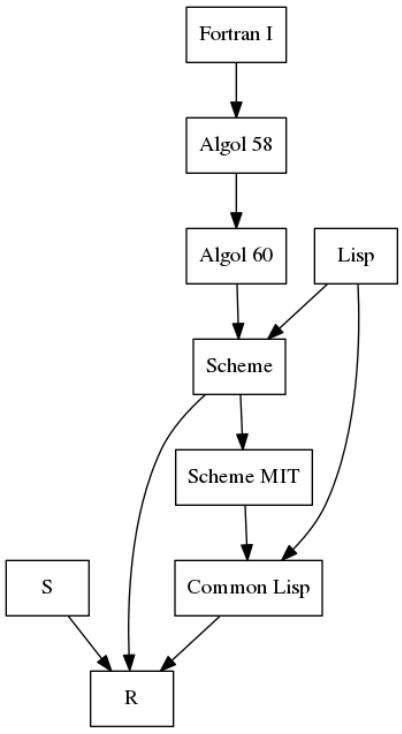

Figure 14: Inheritance diagram for `R`.

15 Ruby

`Ruby` is a dynamic, reflective, object-oriented, general-purpose programming language. It was designed and developed in the mid-1990s by Yokihiro "Matz" Matsumoto in Japan.

According to its creator, `Ruby` was influenced by `Perl`, `Smalltalk`, `Eiffel`, `Ada`, and `Lisp`. It supports multiple programming paradigms, including functional, object-oriented, and imperative. It also has a dynamic type system and automatic memory management.

15.1 Influence

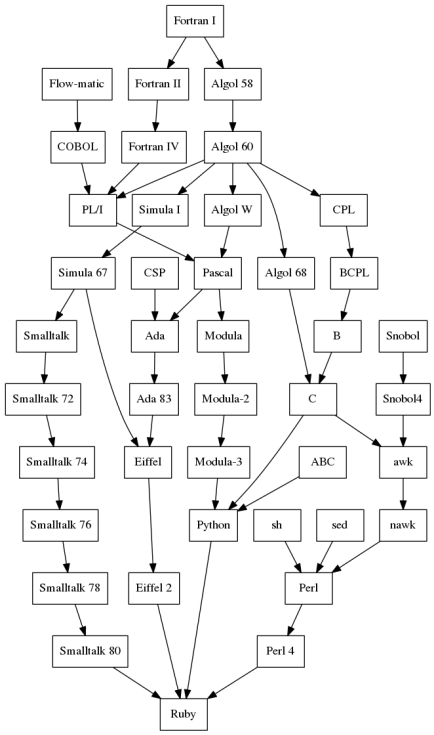

Figure 15: Inheritance diagram for `Ruby`.

16 Rust

`Rust` is a systems programming languages sponsored by Mozilla Research, which describes it as a "safe, concurrent, practice language", supporting functional and imperative-procedural paradigm. `Rust` is syntactically similar to `C++`, but its designers intended it to provide better memory safety whilst maintaining performance.

`Rust` is an open source programming language. Its designers have refined the language through the experiences of writing the Servo web browser layout engine and the `Rust` compiler. A large portion of current commits to the project are from community members.

`Rust` won first place for "most loved programming language" in Stack Overflow Developer Survey in 2016 and 2017; it is referenced in The Book of Mozilla as "oxidized metal".

16.1 Influence

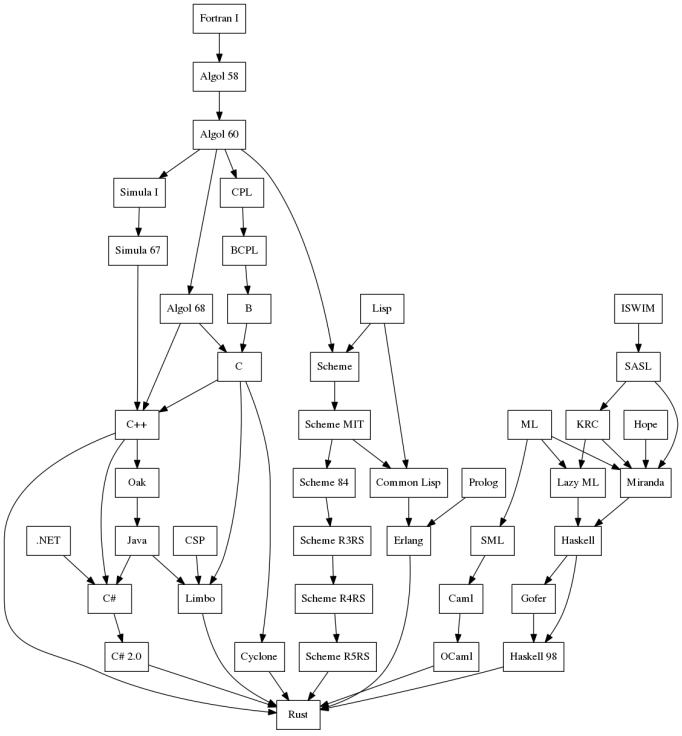

Figure 16: Inheritance diagram for `Rust`.

17 Scala

`Scala` is a general-purpose programming language providing support for functional programming and a strong static type stream. Designed to be concise, many of `Scala`'s design decisions aimed to address criticisms of `Java`.

`Scala` source code is intended to be compiled to `Java` bytecode, so that the resulting executable code runs on a `Java` virtual machine. `Scala` provides languages interoperability with `Java`, so that libraries written in both languages may be referenced directly in `Scala` or `Java` code. Like `Java`, `Scala` is object-oriented, and use a curly-brace syntax reminiscent of the `C` programming languages like `Scheme`, `Standard ML` and `Haskell`, including currying, type interface, immutability, lazy evaluation, and pattern matching. It also has an advance type system supporting algebraic data types, covariance and contravariance, higher-order types (but not higher-rank types), and anonymous types. Other features of `Scala` not present in `Java` include operator overloading, optional parameters, named parameters, and raw strings. Conversely, a features of `Java` not `Scala` is checked exceptions, which have proved controversial.

The name `Scala` is a portmanteau of *scalable* and *language*, signifying that it is designed to grow with the demands of its users.

17.1 Influence

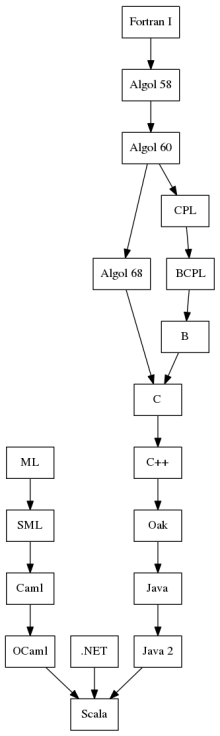

Figure 17: Inheritance diagram for `Scala`.