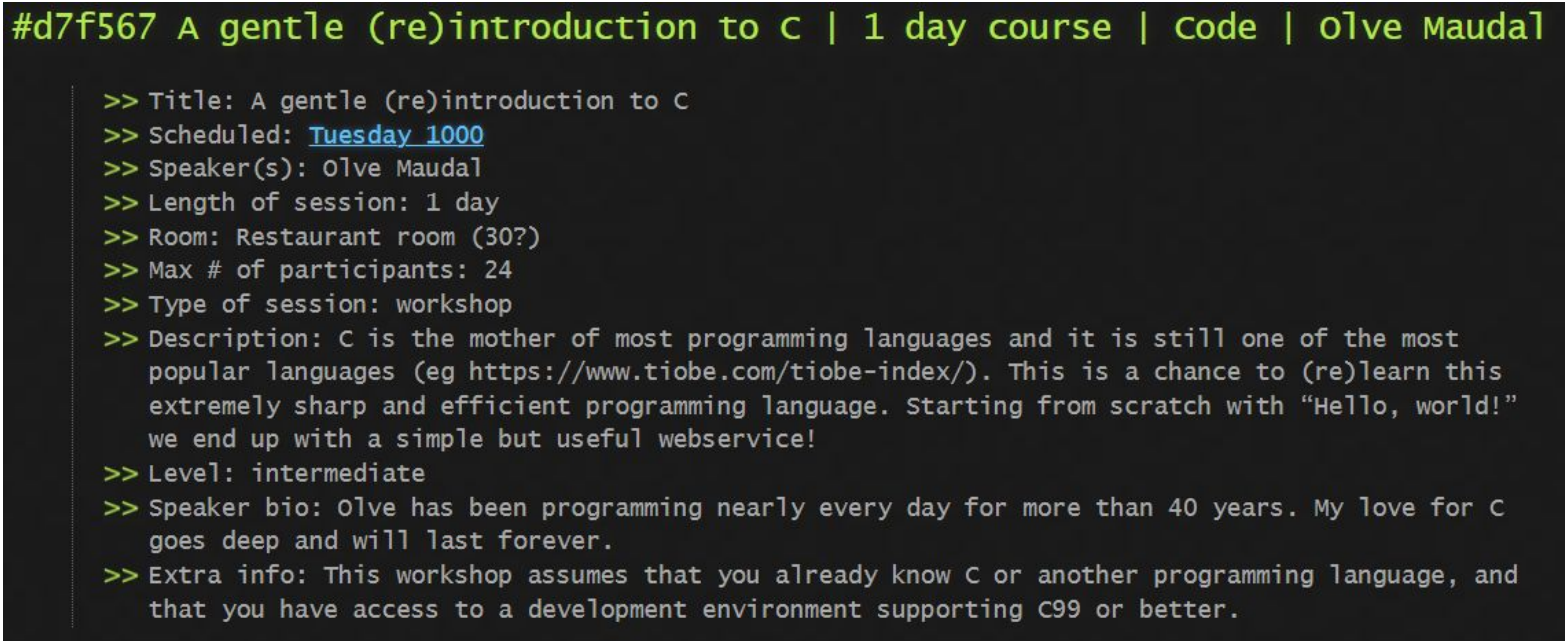
A gentle (re)introduction to C

with Olve Maudal

A one day workshop at EDC Software 2022, September 13

- Introduction
- Getting started
- Organizing
- Optimizing
- Pointers, arrays and structures
- Tools and services
- Outroduction

Introduction



#d7f567 A gentle (re)introduction to C | 1 day course | Code | Olve Maudal

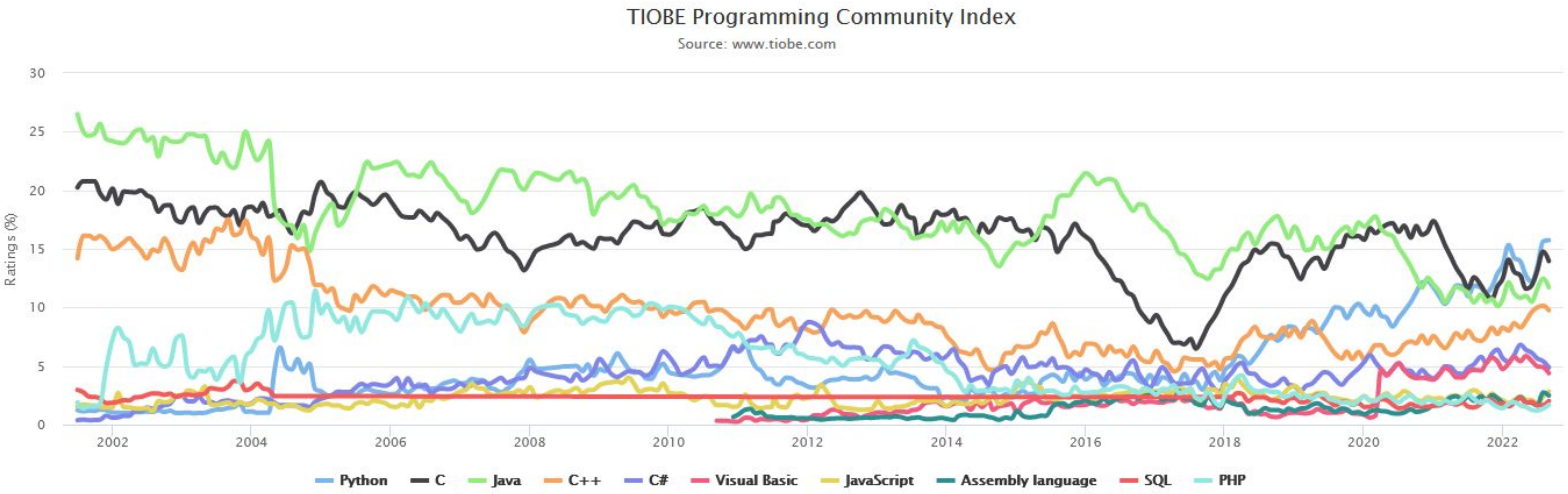
>> Title: A gentle (re)introduction to C
>> Scheduled: Tuesday 1000
>> Speaker(s): Olve Maudal
>> Length of session: 1 day
>> Room: Restaurant room (30?)
>> Max # of participants: 24
>> Type of session: workshop
>> Description: C is the mother of most programming languages and it is still one of the most popular languages (eg https://www.tiobe.com/tiobe-index/). This is a chance to (re)learn this extremely sharp and efficient programming language. Starting from scratch with “Hello, world!” we end up with a simple but useful webservice!
>> Level: intermediate
>> Speaker bio: Olve has been programming nearly every day for more than 40 years. My love for C goes deep and will last forever.
>> Extra info: This workshop assumes that you already know C or another programming language, and that you have access to a development environment supporting C99 or better.

What is your background with programming?

What is your experience with C?

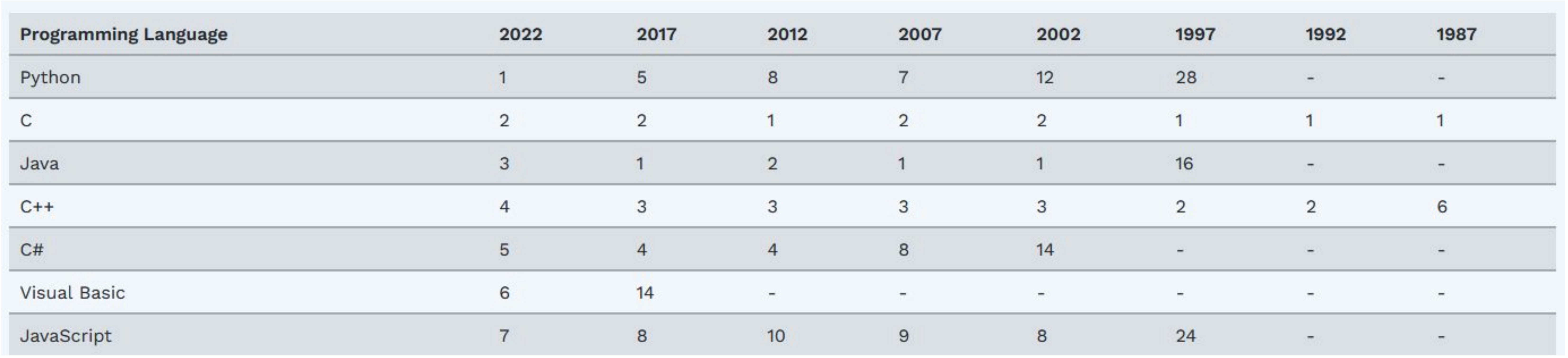
What is your expectations for this course?

Why would anyone want to learn about C (or C++) these days?



TIOBE Programming Community Index

Source: www.tiobe.com

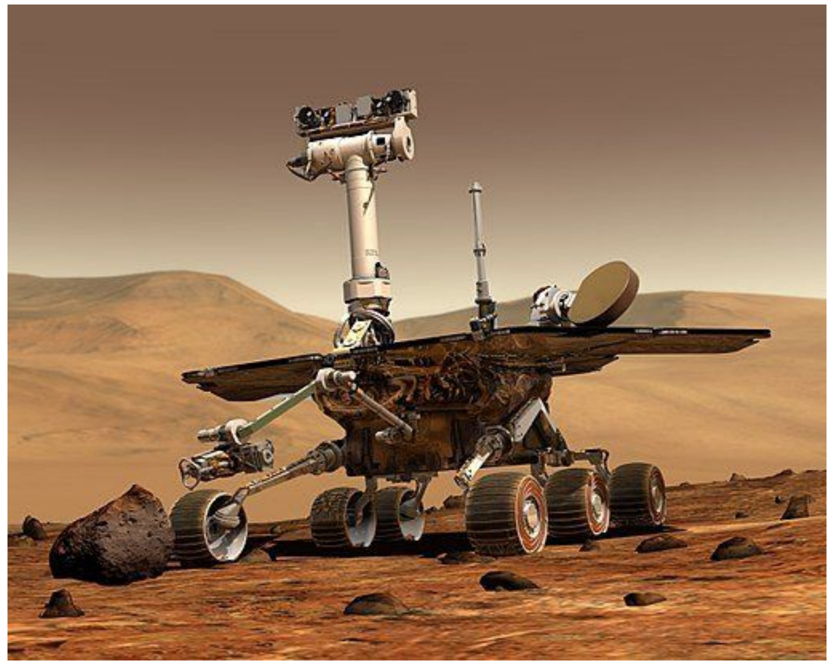


Programming Language	2022	2017	2012	2007	2002	1997	1992	1987
Python	1	5	8	7	12	28	-	-
C	2	2	1	2	2	1	1	1
Java	3	1	2	1	1	16	-	-
C++	4	3	3	3	3	2	2	6
C#	5	4	4	8	14	-	-	-
Visual Basic	6	14	-	-	-	-	-	-
JavaScript	7	8	10	9	8	24	-	-

ZWILLING

11:30
SENCOR
INNER SPACE +

Artificial cardiac
pacemaker



C is widely used for **systems programming in implementing operating systems and embedded system application**s. This is for several reasons:

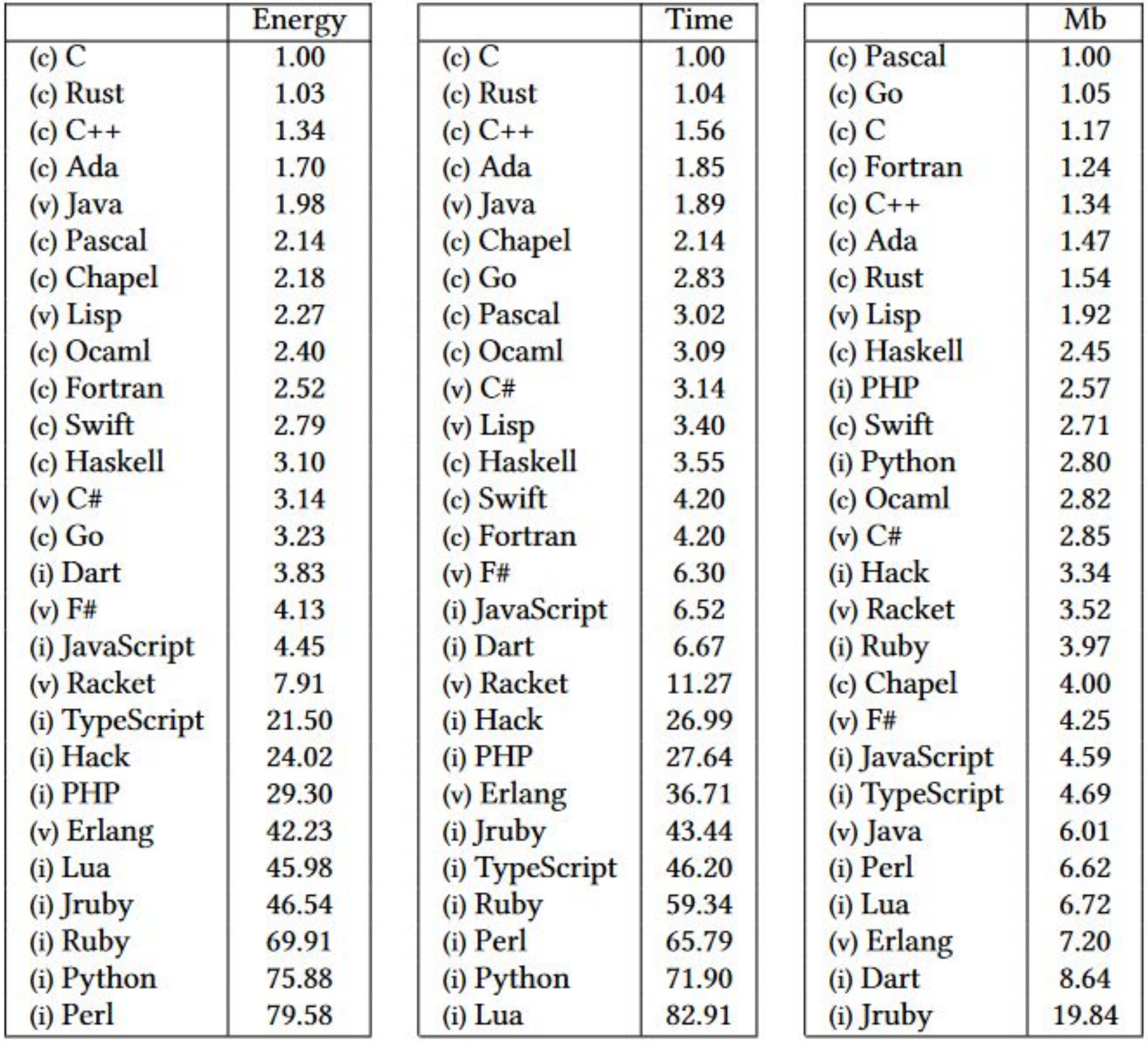
- The code generated **after compilation doesn't demand many system features**, and can be invoked from some boot code in a straightforward manner – **it's simple to execute**.
- The C language statements and expressions typically **map well on to sequences of instructions for the target processor**, and consequently there is a low run-time demand on system resources – **it's fast to execute**.
- The language makes it **easy to overlay structures onto blocks of binary data**, allowing the data to be comprehended, navigated and modified – it can write data structures, even file systems.
- The language supports a rich set of operators, including bit manipulation, for integer arithmetic and logic, and perhaps different sizes of floating point numbers – **it can process appropriately-structured data effectively**.
- Platform hardware can be accessed with pointers and type punning, so system-specific features (e.g. Control/Status Registers, I/O registers) can be configured and used with code written in C – **it interacts well with the platform it's running on**.
- Depending on the linker and environment, C code can also call libraries written in assembly language, and may be called from assembly language – **it interoperates well with other code**.
- **C has a very mature and broad ecosystem**, including open source compilers, debuggers and utilities, and is the de facto standard. It's **likely the drivers already exist in C**, or that there is a similar CPU architecture as a back-end of a C compiler, so there is **reduced incentive to choose another language**.

A consequence of C's wide availability and efficiency is that compilers, libraries and interpreters of other programming languages are often implemented in C. **For example, the reference implementations of Python, Perl, Ruby, and PHP are written in C**.

C enables programmers to create efficient implementations of algorithms and data structures, because the layer of abstraction from hardware is thin, and its overhead is low, an important criterion for computationally intensive programs. **For example, the GNU Multiple Precision Arithmetic Library, the GNU Scientific Library, Mathematica, and MATLAB are completely or partially written in C**. Many languages support calling library functions in C, **for example, the Python-based framework NumPy uses C for the high-performance and hardware-interacting aspects**.

C is sometimes used as an **intermediate language by implementations of other languages**. This approach may be used for portability or convenience; by using C as an intermediate language, additional machine-specific code generators are not necessary. **C has some features, such as line-number preprocessor directives and optional superfluous commas at the end of initializer lists, that support compilation of generated code**. However, some of C's shortcomings have prompted the development of other C-based languages specifically designed for use as intermediate languages, such as C--.

https://en.wikipedia.org/wiki/C_(programming_language)



	Energy
(c) C	1.00
(c) Rust	1.03
(c) C++	1.34
(c) Ada	1.70
(v) Java	1.98
(c) Pascal	2.14
(c) Chapel	2.18
(v) Lisp	2.27
(c) Ocaml	2.40
(c) Fortran	2.52
(c) Swift	2.79
(c) Haskell	3.10
(v) C#	3.14
(c) Go	3.23
(i) Dart	3.83
(v) F#	4.13
(i) JavaScript	4.45
(v) Racket	7.91
(i) TypeScript	21.50
(i) Hack	24.02
(i) PHP	29.30
(v) Erlang	42.23
(i) Lua	45.98
(i) Jruby	46.54
(i) Ruby	69.91
(i) Python	75.88
(i) Perl	79.58

	Time
(c) C	1.00
(c) Rust	1.04
(c) C++	1.56
(c) Ada	1.85
(v) Java	1.89
(c) Chapel	2.14
(c) Go	2.83
(c) Pascal	3.02
(c) Ocaml	3.09
(v) C#	3.14
(v) Lisp	3.40
(c) Haskell	3.55
(c) Swift	4.20
(c) Fortran	4.20
(v) F#	6.30
(i) JavaScript	6.52
(i) Dart	6.67
(v) Racket	11.27
(i) Hack	26.99
(i) PHP	27.64
(v) Erlang	36.71
(i) Jruby	43.44
(i) TypeScript	46.20
(i) Ruby	59.34
(i) Perl	65.79
(i) Python	71.90
(i) Lua	82.91

	Mb
(c) Pascal	1.00
(c) Go	1.05
(c) C	1.17
(c) Fortran	1.24
(c) C++	1.34
(c) Ada	1.47
(c) Rust	1.54
(v) Lisp	1.92
(c) Haskell	2.45
(i) PHP	2.57
(c) Swift	2.71
(i) Python	2.80
(c) Ocaml	2.82
(v) C#	2.85
(i) Hack	3.34
(v) Racket	3.52
(i) Ruby	3.97
(c) Chapel	4.00
(v) F#	4.25
(i) JavaScript	4.59
(i) TypeScript	4.69
(v) Java	6.01
(i) Perl	6.62
(i) Lua	6.72
(v) Erlang	7.20
(i) Dart	8.64
(i) Jruby	19.84

https://greenlab.di.uminho.pt/wp-content/uploads/2017/10/sleFinal.pdf