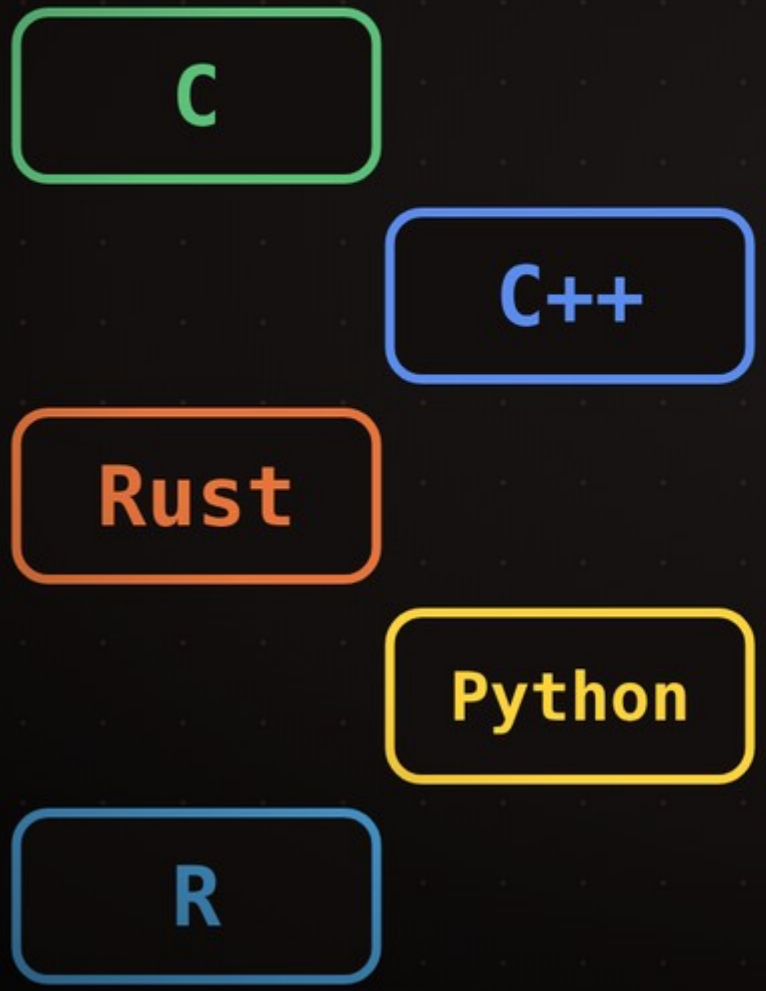
@MyWebUniversity

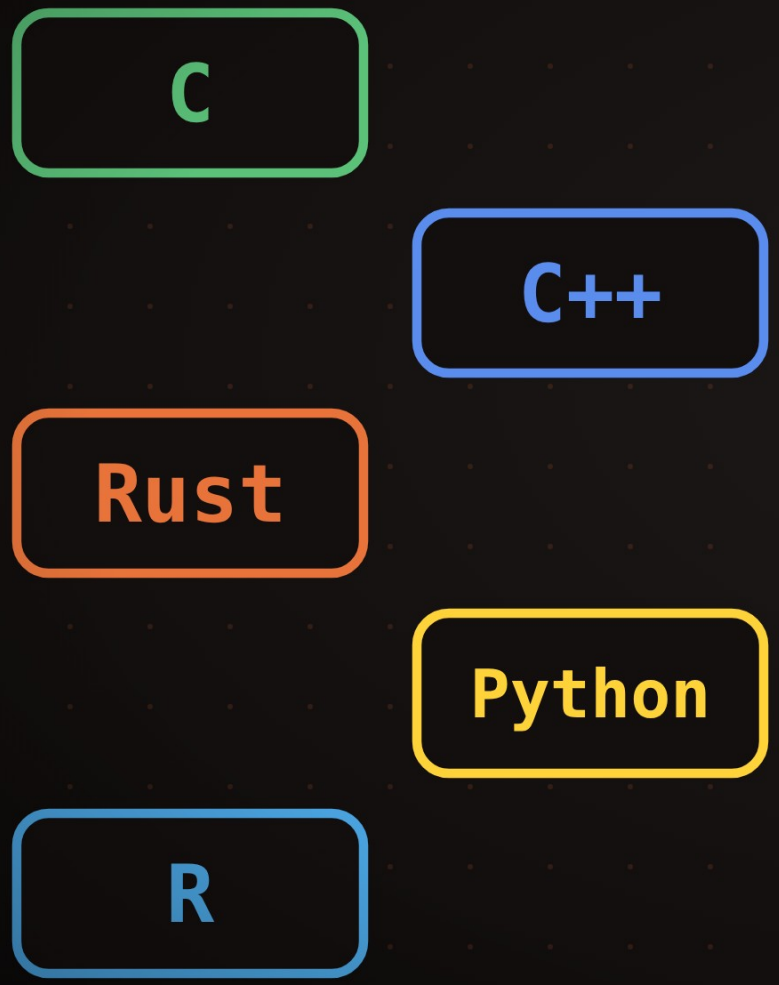
Rust · C · C++
Python · R



Which One, When & Why?

speed · safety · ease · libraries

70% OF BUGS = MEMORY

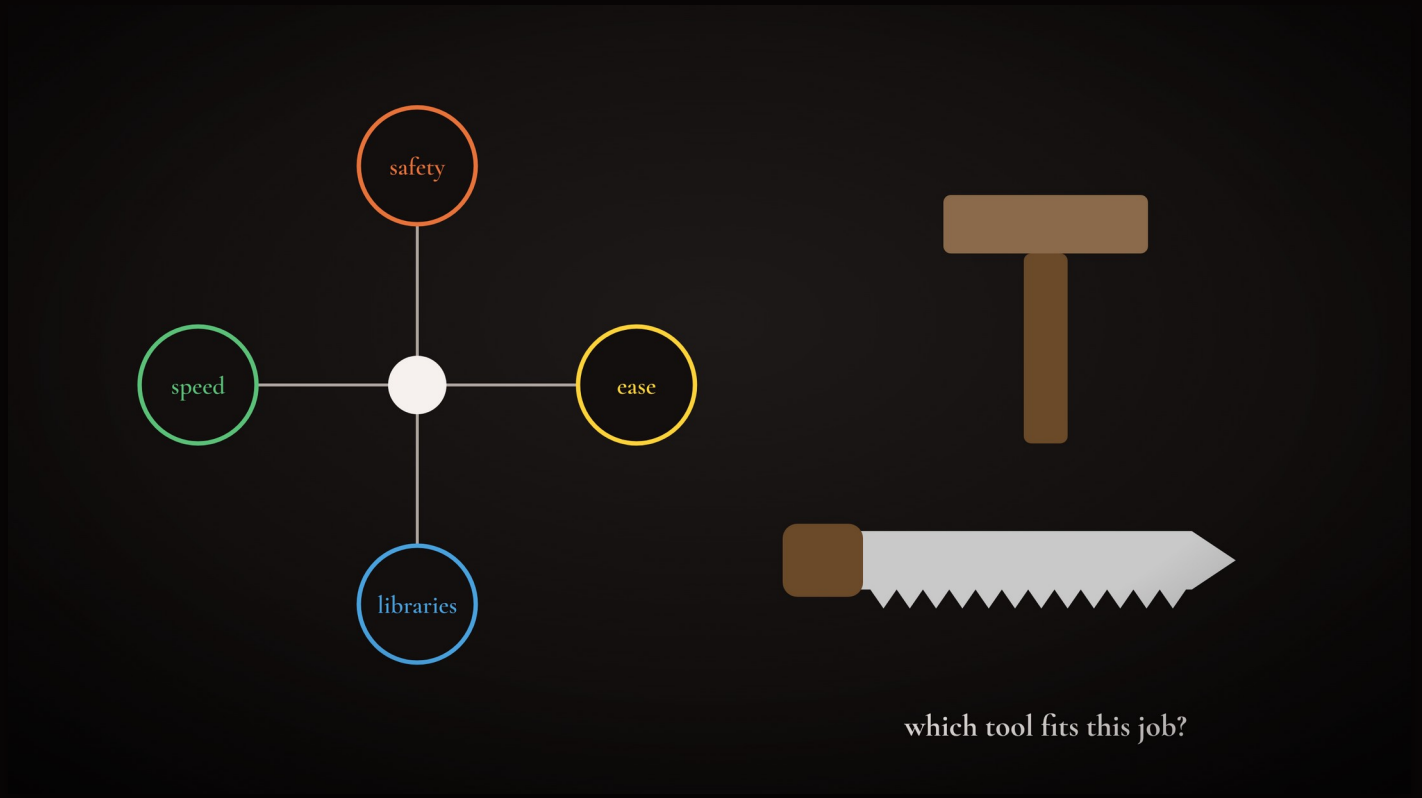


A MYWEBUNIVERSITY STORY

Rust · C · C++ Python · R

Which Language, When, and Why?

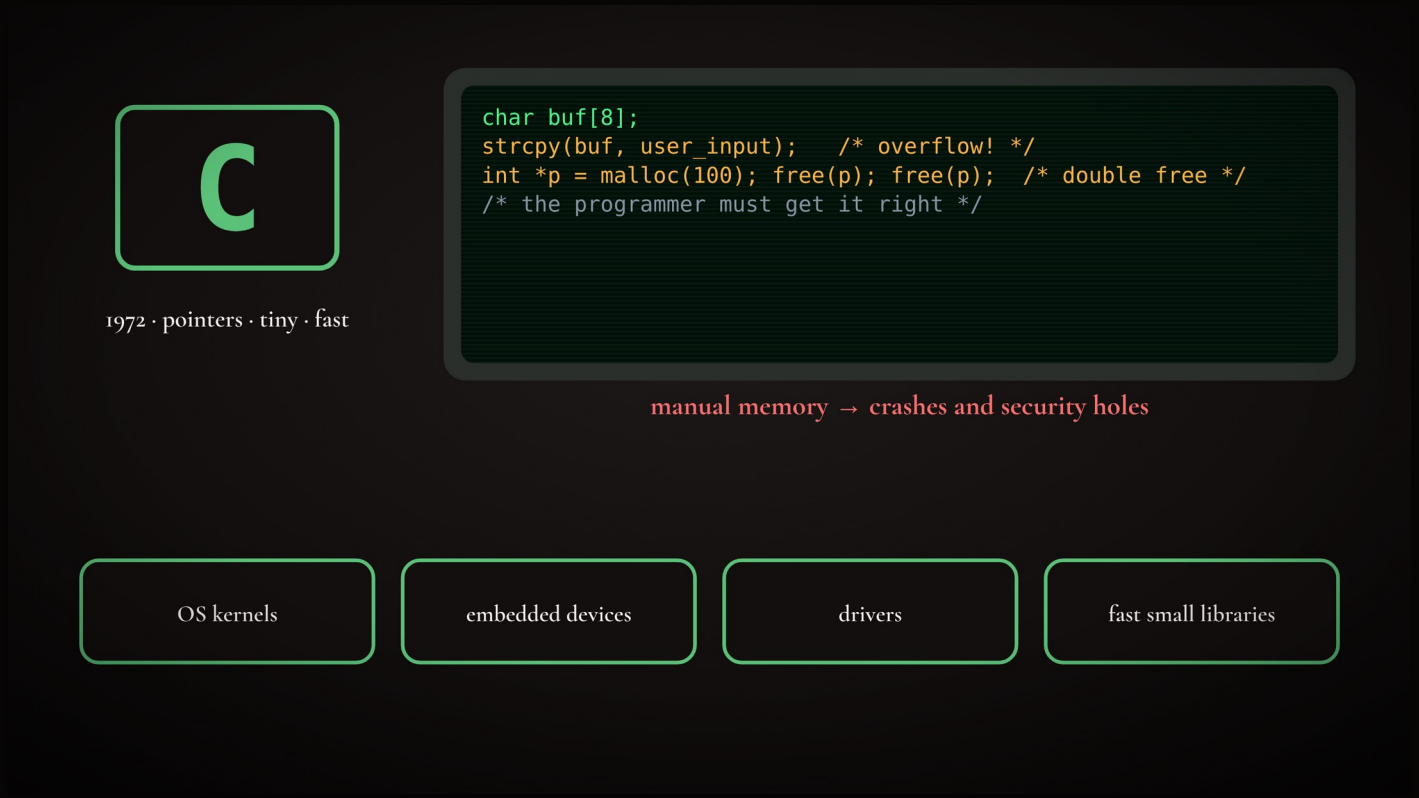
There Is No Best Language



- Speed · safety · ease · libraries
- No language wins on everything
- Pick the tool that fits the job

01

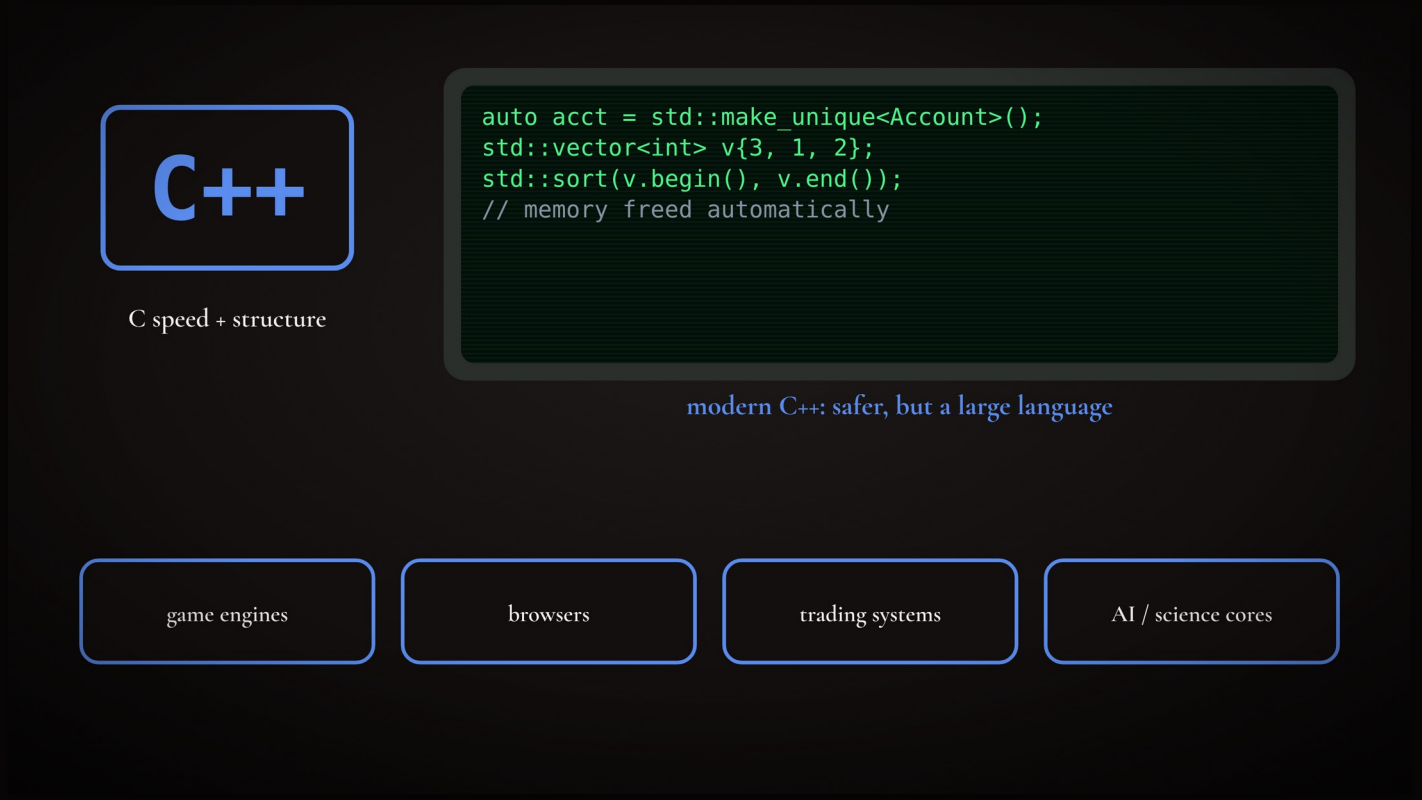
C: Close to the Machine



- 1972 · pointers · tiny and fast · runs everywhere
- Manual memory → crashes and security holes
- Kernels, embedded, drivers, small libraries

02

C++: Speed with Structure



- Classes, templates, STL at C speed
- Modern C++ + smart pointers = safer, but large
- Games, browsers, trading, AI/scientific cores

03

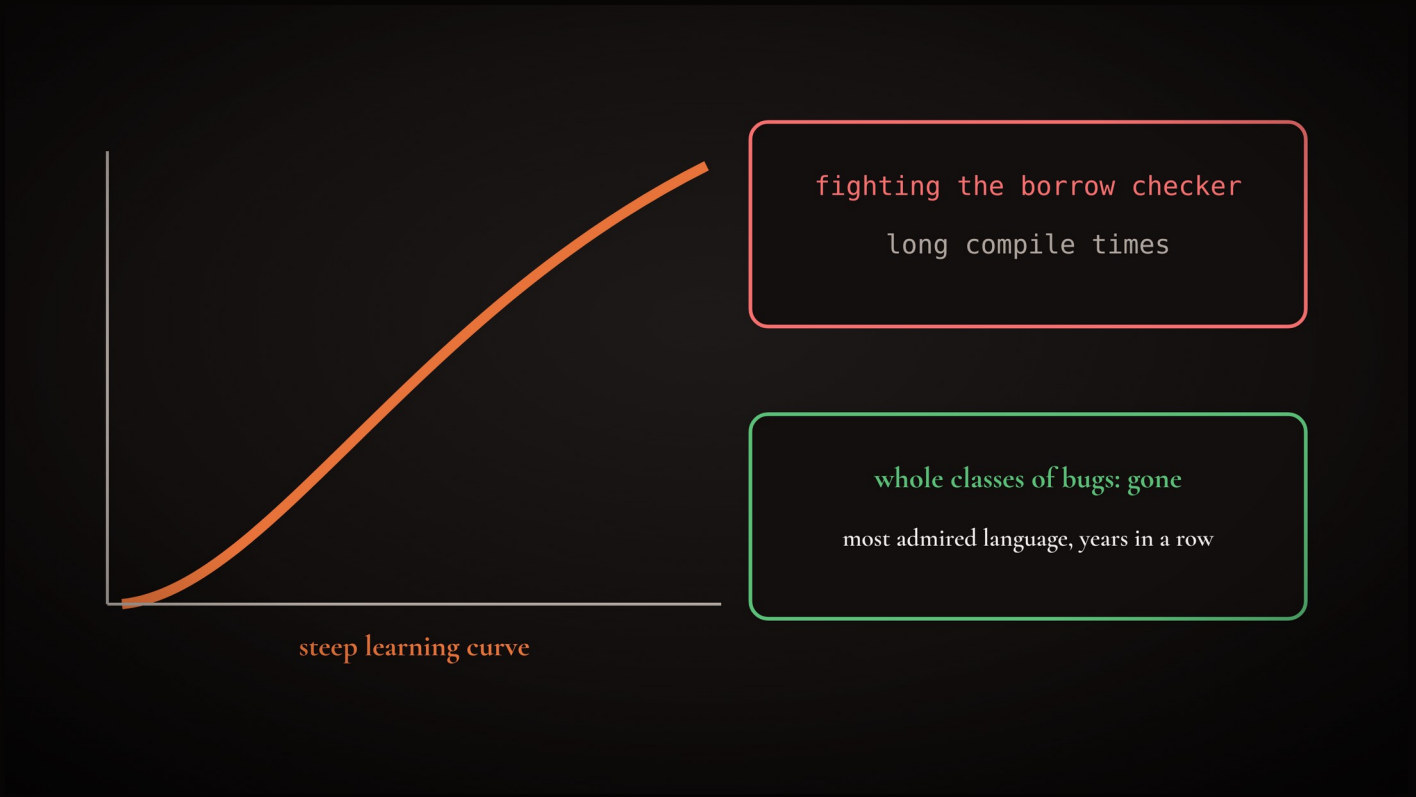
Rust: Fast and Memory-Safe



- 2006 Graydon Hoare · Mozilla · 1.0 in May 2015
- Ownership + borrow checker: safe, no GC, C-like speed
- ~70% of serious bugs are memory-safety bugs (Microsoft, Google) · Linux kernel since 2022

04

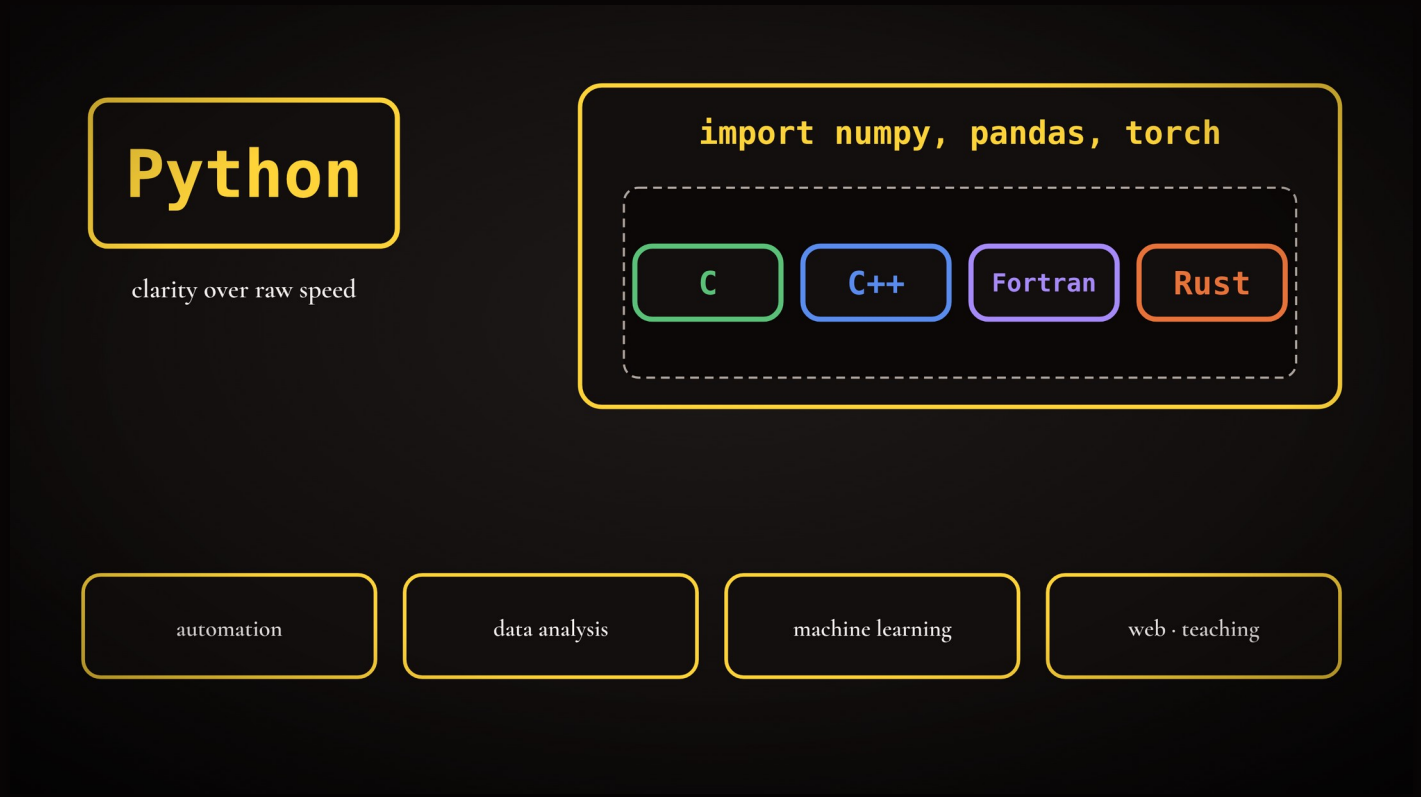
Rust's Price



- Steep learning curve
- Long compile times
- Whole classes of bugs eliminated

05

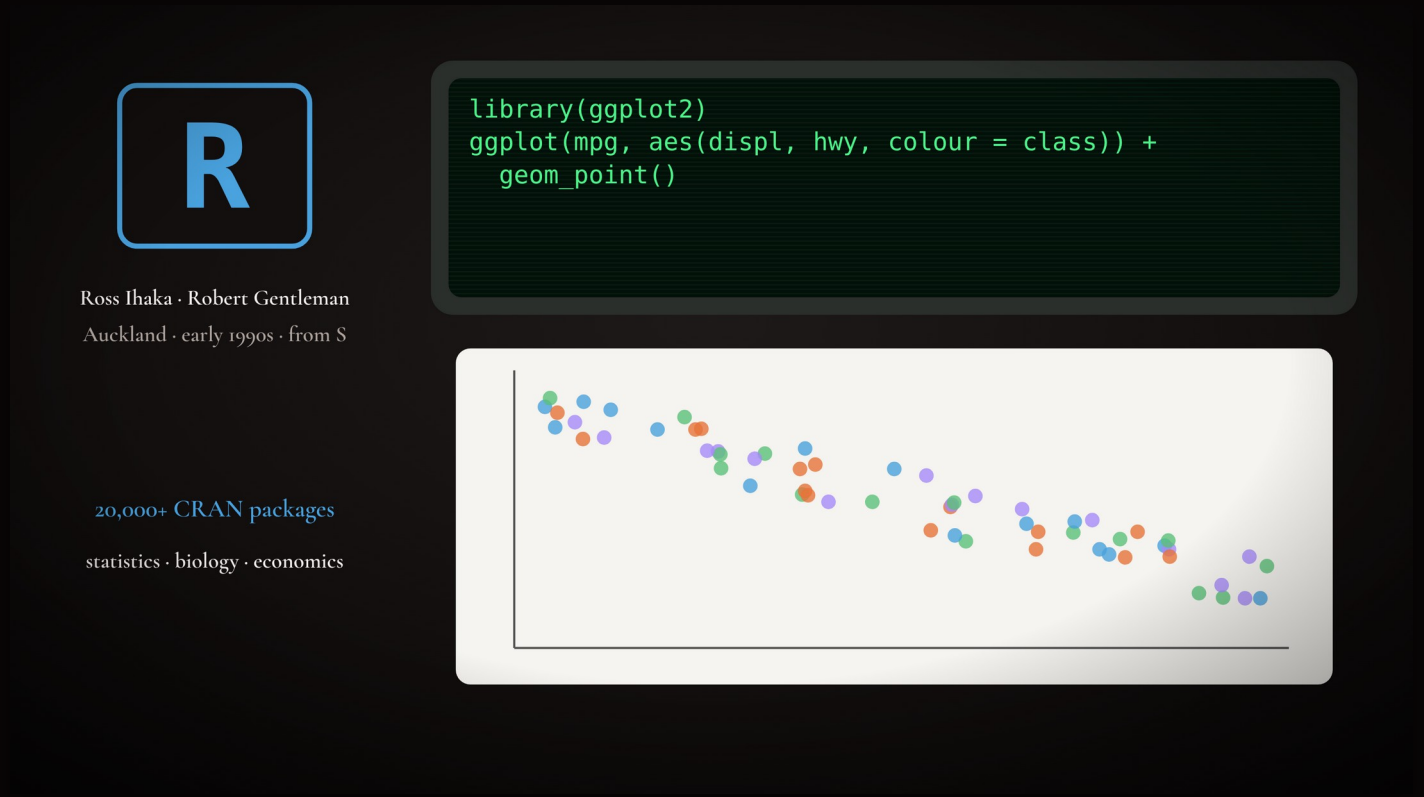
Python: Clarity and Libraries



- Clarity over raw speed
- NumPy, pandas, PyTorch: C/C++/Fortran (and Rust) inside
- Automation, data, ML, web, teaching

06

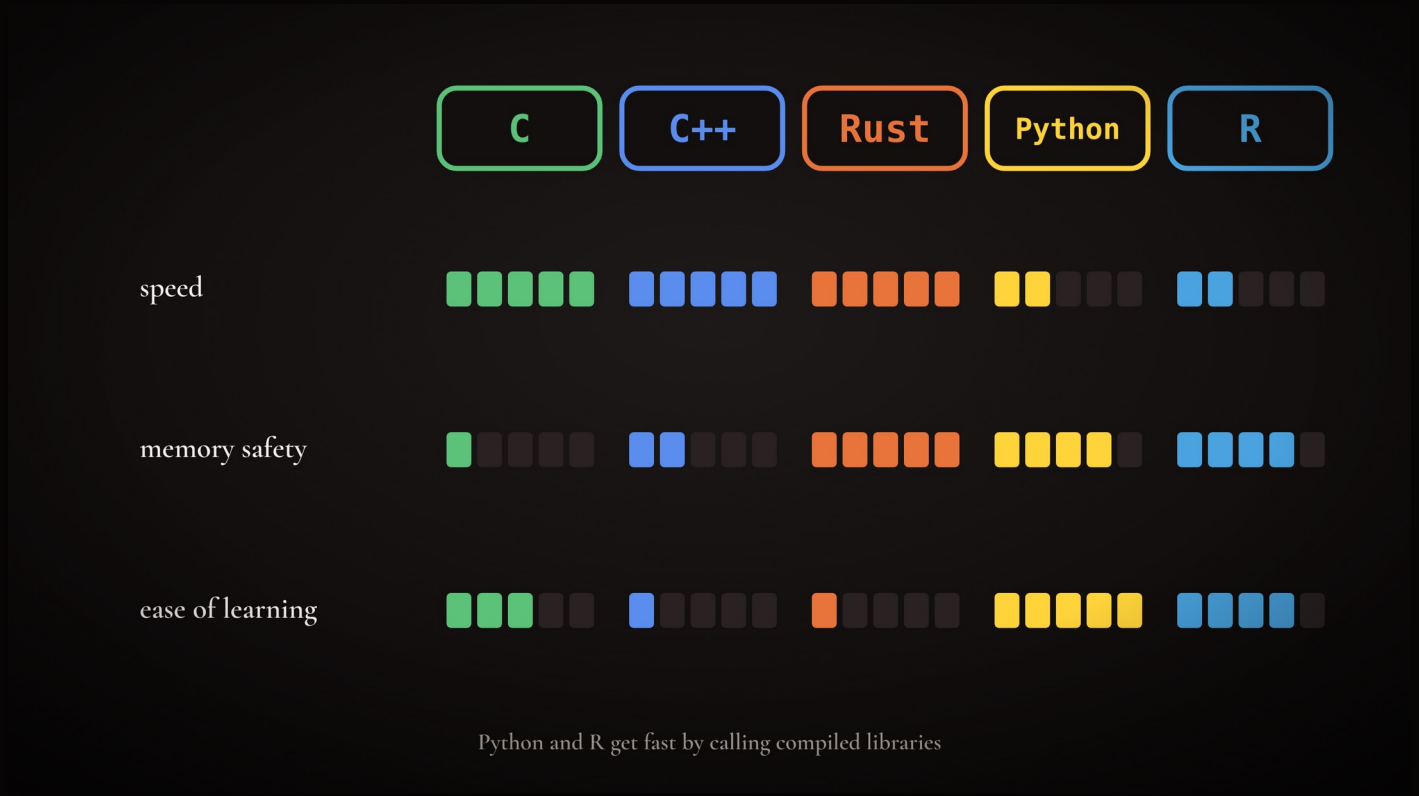
R: The Statistician's Language



- Early 1990s: Ross Ihaka & Robert Gentleman (Auckland)
- Based on S (Bell Labs) · statistics + graphics
- ggplot2 · tidyverse · 20,000+ CRAN packages

07

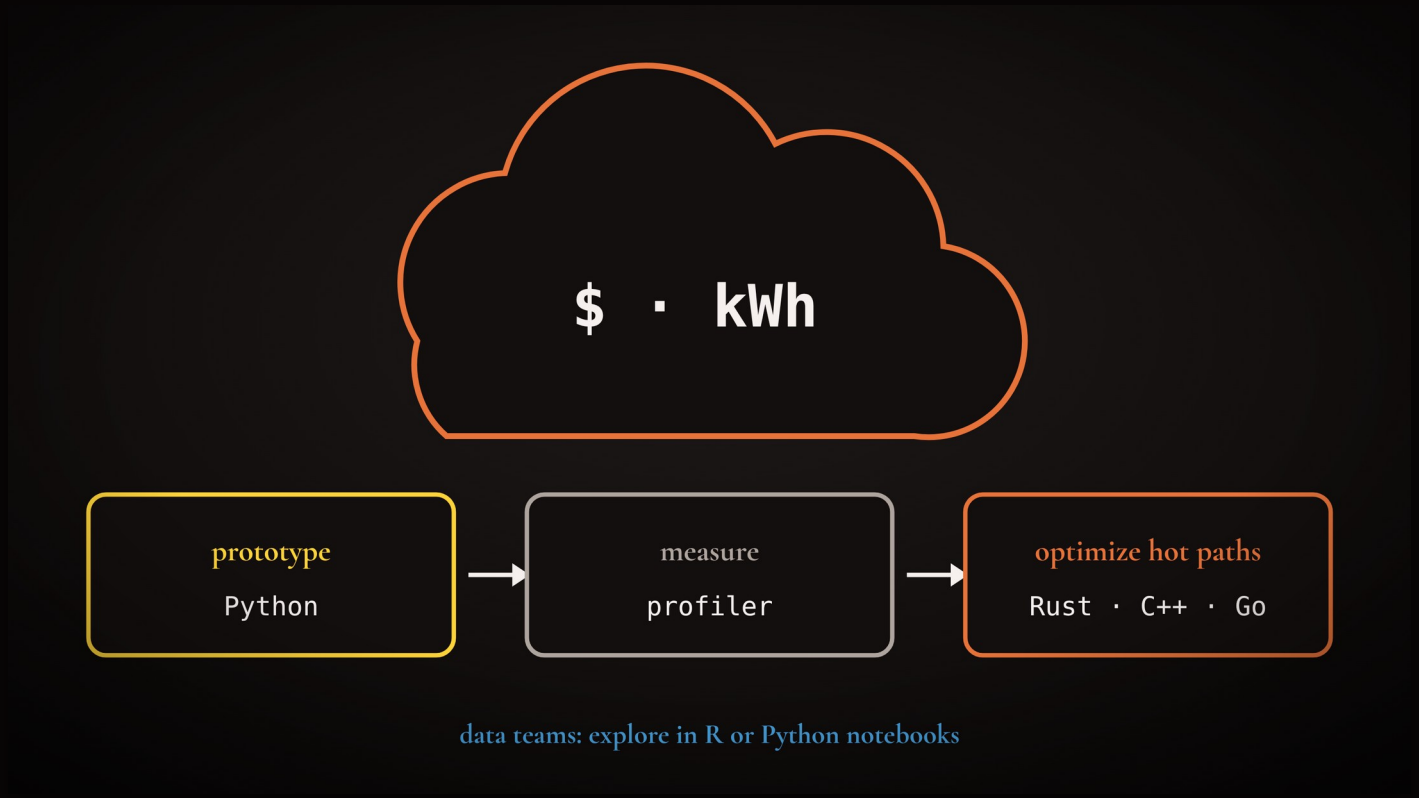
Side by Side



- Speed: C ≈ C++ ≈ Rust ≫ Python, R
- Memory safety: Rust, Python, R by default
- Ease: Python > R > C > C++, Rust

08

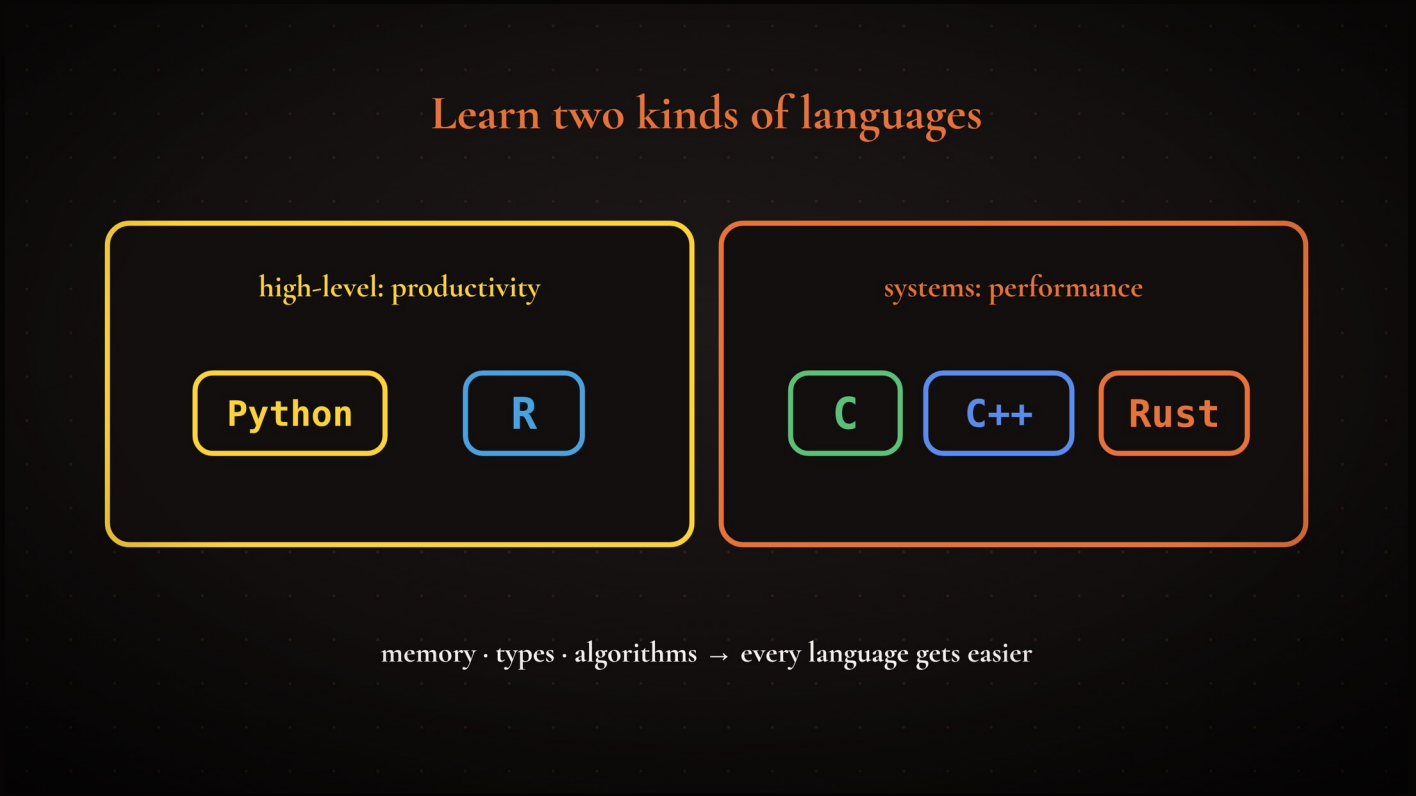
In the Cloud



- Cloud: speed and memory = money
- Prototype in Python, optimize hot paths in Rust/C++/Go
- Explore in R/Python notebooks

09

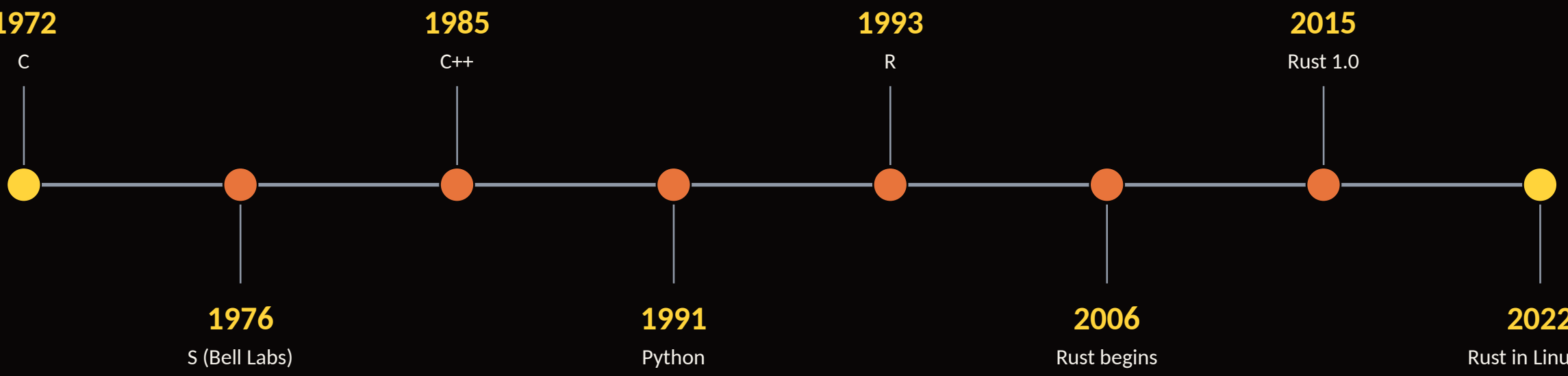
Learn Two Kinds



- Know one high-level language (Python or R)
- Know one systems language (C, C++ or Rust)
- Ideas transfer: memory, types, algorithms

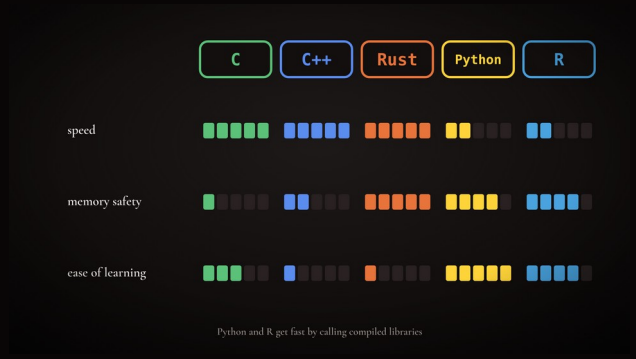
10

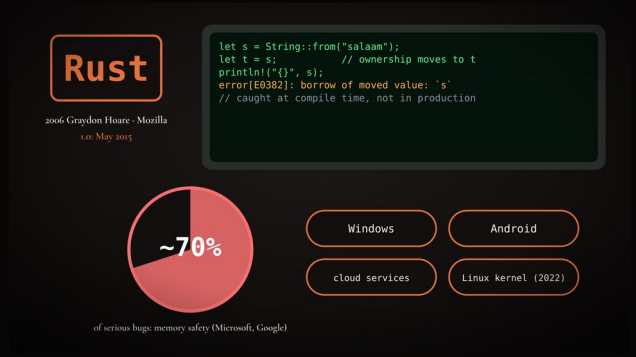
Timeline: 1972 to 2022



Sources and Further Reading

1. The Rust Programming Language (“The Book”) — doc.rust-lang.org/book
2. Microsoft Security Response Center, “A proactive approach to more secure code” (2019) · Chromium memory-safety statistics
3. Linux 6.1 release notes (initial Rust support), December 2022
4. R Project — r-project.org · Ihaka & Gentleman, “R: A Language for Data Analysis and Graphics” (1996)
5. CISA, “The Case for Memory Safe Roadmaps” (2023)





Thank you for watching

MyWebUniversity.com

Free technical education since 2000 — English and Dari

@MyWebUniversity · @MyWebUniversity-Dari