

The Rust Programming Language

Pramode C.E

<http://pramode.net>

September 28, 2016























What is common to all these?

```
0 1 1 0 1 1 0 1 1 0 0 0 1 1 0 ...
```

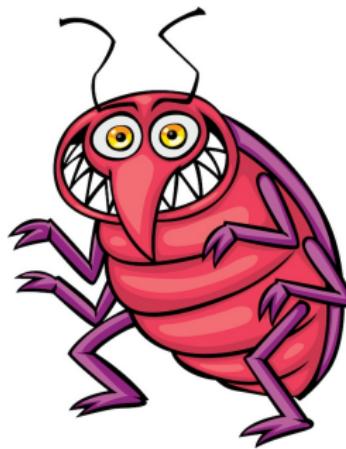
A lot of code ...

- Cardiac Pacemaker: 80,000 lines of code
- MRI Scanner: 70 lakh lines
- Boeing 787 dreamliner: 70 lakh lines
- Ford GT(Car): 100 lakh lines
- Light bulb: ??
- Door lock: ??
- Oven: ??

Ref: <http://www.informationisbeautiful.net/visualizations/million-lines-of-code/>

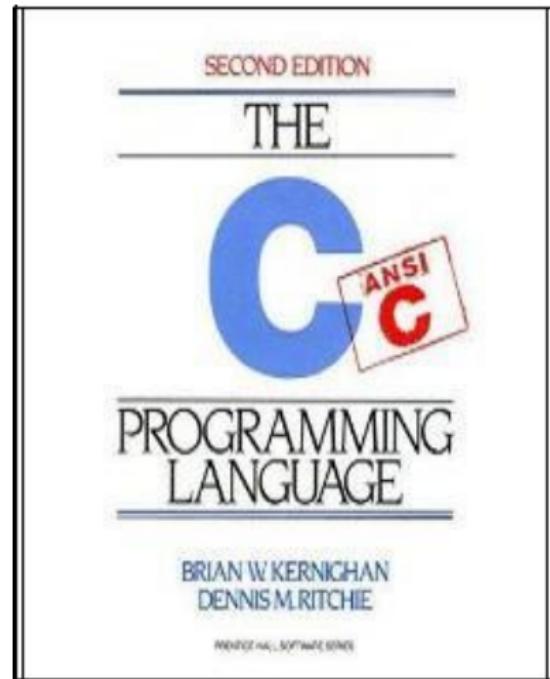


With code comes bugs ...



Code bugs can kill ... and create havoc!





What comes to your mind when you think of C?





- Buffer overflow / off-by-one errors
- Null dereferencing
- Use-after-free
- Memory leaks
- Dangling Pointers
- Undefined Behaviours

C Evil Spirits

"The second big thing each student should learn is: How can I avoid being burned by Cs numerous and severe shortcomings? This is a development environment where only the paranoid can survive"
John Regehr — <http://blog.regehr.org/archives/1393>

The Promise of Rust

- Guaranteed Memory SAFETY without using Garbage Collection
- As fast as C/C++
- Can write OS kernels, embedded systems code
- High level abstractions like algebraic data types / pattern matching with very little performance overhead
- Minimal run-time
- Easy interfacing with C
- Threads without data races

Ref: <https://www.rust-lang.org/>

The Promise of Rust

- Graydon Hoare started working on Rust in 2006
- Mozilla became interested - idea was to use it in the experimental Servo browser engine project
- Rust becomes an official Mozilla project
- Several major design changes before hitting v1.0 in May 2015.
Language stabilizes
- Regular six week release cycle

- Download from
<https://www.rust-lang.org/en-US/downloads.html>
- Untar and add folders containing "rustc" and "cargo" to the PATH

Or, you can just run:

```
curl -sSf https://static.rust-lang.org/rustup.sh | sh
```

- Official "book": <https://doc.rust-lang.org/book/>
- New version of the book (work-in-progress):
<http://rust-lang.github.io/book/>
- Rust By Example: <http://rustbyexample.com/>
- O'Reilly book:
<http://shop.oreilly.com/product/0636920040385.do>

Who is using it?

Remember, Rust is just a 1 year old baby!

- Organizations running Rust in Production:
<https://www.rust-lang.org/en-US/friends.html>
- Mozilla, for the Servo project.
- Dropbox story: <http://www.wired.com/2016/03/epic-story-dropboxs-exodus-amazon-cloud-empire/>
- Coursera is using Rust:
<https://building.coursera.org/blog/2016/07/07/rust-docker-in-production-coursera/>

Interesting Projects

- Servo: the secure, parallel browser engine from Mozilla: <https://servo.org/>
- Redox OS: Unix-like microkernel OS written in Rust: <https://www.redox-os.org/>
- Tock OS: an embedded OS for microcontrollers written in Rust: <http://www.tockos.org/about/>

Rust, the most loved programming language (stackoverflow developer survey, 2016):

<http://stackoverflow.com/research/developer-survey-2016>



About this workshop

This workshop is divided into 3 parts:

- Simple
- Cool
- Hard (and Cool!)

The Simple parts

- We will quickly look at: variables, control structures, function basics, basic types etc.
- A demo of my first Rust project - a Raspberry Pi GPIO library (ported from Python)

The Cool parts

- Algebraic Data types
- Pattern Matching
- Generics and Traits
- Iterators
- Closures
- Cargo

and some other things ...

The Hard(and Cool) Parts

This is the **core** of Rust - where the real innovation comes in.

- Systems programming basics - understanding buffer overflows, use-after-free, memory leaks, dangling pointers etc.
- Ownership and move semantics
- Borrowing
- Lifetimes
- Unsafe Rust

Let's dive into code!

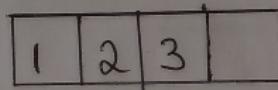
2000



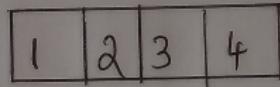
3400



5600



5600



8230

