Programming in Rust

Introduction and course organisation

Szymon Zyguła, MSc Eng.

Faculty of Mathematics and Information Science
Warsaw University of Technology

Winter semester 2026/27

Organisation

Key rules for passing the course

- Attendance:
  - lectures – optional,
  - laboratories – mandatory (at most 4 unexcused absences from graded ones).
- Points:
  - 3 homework assignments/projects: 15 pts, 40 pts, 15 pts,
  - 10 graded laboratories: 3 pts each,
    - 1 pt for work during the laboratory (does not apply today),
    - 1 pt for work at home,
    - 1 pt for presenting the solution and answering a question,
  - 100 pts in total,
- Late projects: -5 pts for each week of delay of a homework assignment.
- All assignments are done individually.

Rules on AI

- Using AI to generate solutions is forbidden (ChatGPT, Copilot, etc.).
- AI may be used to explain errors, assess a solution, suggest other approaches, etc.

Lecture plan for the semester I

1. Language basics: cargo basics, data types, control flow, functions, structs, references.
2. Ownership, moves, the borrow checker, lifetimes.
3. Algebraic data types, pattern matching, if-let, let-else and while-let expressions.
4. The standard library and error handling: collections, Option, Result, strings.
5. Generic types, traits, complex lifetimes.
6. Advanced cargo: structuring projects into modules, splitting into libraries, dependency management.
7. Functional programming: higher-order functions, closures, iterators, monadic operations, the Itertools library.
8. Smart pointers and memory management: Box, Rc, Weak, RefCell, Cell, Cow, Borrow(Mut), LazyCell, OnceCell.

Lecture plan for the semester II

9. Object-oriented programming: trait objects, dynamic dispatch, design patterns.
10. Parallel programming and the Rayon library: exclusive access, synchronisation primitives, channels, atomics basics.
11. Advanced traits and types.
12. Declarative macros.
13. Unsafe code: raw pointers, unbounded lifetimes, PhantomData, coercions, transmutes, uninitialised memory, destructors, leaks.
14. Procedural macros: derive, attribute and function-like.
15. Other topics.

Course website

pages.mini.pw.edu.pl/~zygulas/rust/en

Literature

1. Klabnik, Steve, Carol Nichols, and Chris Krycho. *The Rust Programming Language*. 3rd ed. No Starch Press, 2026.
2. Gjengset, Jon. *Rust for Rustaceans: Idiomatic Programming for Experienced Developers*. No Starch Press, 2021.
3. Blandy, Jim, Jason Orendorff, and Leonora FS Tindall. *Programming Rust*. O'Reilly Media, 2021.
4. McNamara, Tim. *Rust in Action*. Simon and Schuster, 2021.

Office hours

Remotely or after the lecture, by prior appointment.

About the language

Timeline

- Graydon Hoare (an employee of Mozilla Research) started working on the language in 2006 as a side project. Motivation: a language in which it is hard to write broken code.
- Since 2009 Mozilla officially supported the development of the language.
- First public release (0.1): early 2012.
- Version 1.0: 2015.
- Current version: 1.99 (released 28 September 2026).
- A new version every 6 weeks.
- Currently sponsored by the Rust Foundation.

What kind of language is it?

- Compiled (LLVM backend).
- No garbage collection.
- Purpose and applications similar to C++.
- Mixed paradigm (neither object-oriented nor functional).
- A much stricter compiler than in C++, e.g. the type system enforces reference validity (a “dangling reference” is impossible).
- Many “ergonomic” features (a powerful macro system, algebraic data types).

Where is it used?

- Firefox,
- Linux,
- Windows,
- Android,
- Discord,
- Amazon Web Services,
- Microsoft Azure,
- Dropbox,
- ...
- open source,
- programming language researchers

Demand

- Demand for Rust programmers keeps growing.
- There are still very few experienced programmers in this language.

Other things

Development environments

- Rust Rover (by JetBrains)
- Visual Studio Code
- Vim / Neovim / Helix / Emacs / any other generic editor

Links

- The book: `https://doc.rust-lang.org/stable/book/`
- Standard library reference: `https://doc.rust-lang.org/stable/std/index.html`