Introduction to Rust

Presented by AcademiaEdge

Teachers: Jacoby Yliniemi (10th)
Assistant Teacher: Eamon Mukhopadayay (10th)

Date and Timings: 2/21 - 5/16 on Sunday from 7-8 P.M. EST
Contact Email: contact@academiaedge.com

Requirements: In this class students are recommended to have some programming knowledge, though it is not a requirement.

About me:

I am a junior in High School with the plan of teaching people how to code. I qualify for this role with my knowledge of Rust, low level concepts, and passion for teaching. Through this course I will show you what I know: low level concepts, everything you need to start programming in Rust, ect. Then you will have a new blazing fast and memory safe systems programming language under your belt!

Class Description:

In this course, students will learn the fundamentals of low level programming with Rust and understand the advantages of Rust's memory system. Along with learning the syntax and some of Rust's quirks, they will build fun applications and algorithms that are often used in coding interviews! Students can ask questions at any time during the class and the assistant or teacher will answer them. Additionally, students may message teachers via google classroom or by email and our teachers will respond as soon as possible. In order to give this individualized experience filled with fun projects and assignments guided towards students, classes will be limited to 10-20 students so that teachers can give high-quality attention to each student. Sign up is first come first serve and a waitlist may be created if there is excessive student participation. This course is guided towards students of grades 7 and up, but any student may join if interested.

Course Structure

- **Unit 1:** Go over the course info
    - Course road map
    - What is Rust and where is it used?
    - How to install Rust?
    - What is Cargo?
    - VScode
- **Unit 2:** Rust syntax
    - Setup a rust project
    - The *main* function
    - VS Code and extensions
    - Print to console
    - Variable types ("String" vs "str", integer types, floats, bool, ect)
    - Mutable and immutable variables
    - Print variables
- **Unit 3:** ownership and borrowing
    - Variables vs constants
    - Ownership and borrowing
    - Stack and the heap
    - Variable shadowing
- **Unit 4:** control flow of the program
    - Operators (==, >, ect)
    - Operator and assignment (-=, +=, ect)
    - If then else statement
    - If let
    - Loops (loop, while, for, ect.)
    - Nesting
    - Interview algorithms with loops ("fizz buzz")
- **Unit 5:** Cargo and crates
    - What is a crate?
    - The cargo.toml file
    - crates.io
    - Input with std::io crate
    - Number guessing game
- **Unit 5 Project:** adding features to guessing game
- **Unit 6:** functions
    - Functions
    - Return from functions
    - Ownership in parameters
    - Slice type (own function)

    - Input (own function)
- **Unit 6 Project:** text adventure game
- **Unit 7:** structs and enums
    - What is a struct?
    - Making a struct
    - Assigning a variable to the structure
    - Implementing methods to struct
    - What is an Enum?
    - Making an enum
    - What is a variant?
    - Assigning a variable to a variant of the enum
    - Storing variables on variants
    - Implementing methods in enums
- **Unit 7 Project:** Console shopping
- **Unit 8:** Modules
    - What is a module?
    - Making our own nested module
    - Accessing our module namespace
    - Using namespaced modules
    - “Use” and “pub” keywords
    - Public functions
    - Module paths (“self, crate, super” keywords)
    - Splitting code into different files
- **Unit 9:** more modules!
    - Public constants
    - Public structs
    - Public enums
    - Review so far
- **Unit 10:** hashmaps, parsing, and file editing
    - Standard collections namespace
    - What is a hashmap?
    - Making an hashmap
    - Editing a map
    - Parsing info
    - Read and write from a file
    - Read file to hashmap
- **Unit 10 Project:** custom console commands to read and write to text file
- **Unit 11:** standard enums
    - Result enum
    - Option enum

    - Std enum examples
- **Unit 12; Final:** create your own program! (impress me)
    - You could make…
    - Advance on old projects
    - Advance text adventure
    - Your own object notation language
    - Research a crate and use it (rocket-rs, serenity-rs, rusqlite)
- **Last day:** Project showcase

Rules and Expectations

Classroom Procedures:

Students are to stay muted at all times except if they have a question or when asked to be unmuted. The student may temporarily unmute himself to ask his/her question. Alternatively, if the student would not like to speak in front of the class, then the student may ask his/her question in the Zoom chat. We encourage students to ask questions and regularly participate in class. Also, we would like students to be respectful to their classmates and teachers.

Students, please do not:

- Eat or drink with your microphone turned on
- Be disrespectful to teachers or other students
- Put inappropriate pictures on your webcam
- Send inappropriate messages in the class chat

Please do:

- Ask questions
- Be attentive
- Be engaged and active throughout the class
- Make sure to have your camera on throughout the class
- Do assignments thoroughly
- Submit assignments before the deadline
- Have Fun!

Google Classroom Layout:

Each lesson's recording will be found on google classroom along with the class's slides and notes. Homework assignments will be assigned and submitted via google classroom as well. Students can ask questions through the messaging system in google classroom or via email.

Homework procedures:

Students will be given homework in google classroom via google docs, which will consist of inserting screenshots or short-answer/multiple-choice questions, or google forms. The google forms will mainly be used for knowledge checks, while the google docs will be used for general homework assignments. Each assignment is due 24 hours before the next class to give ample time for teachers to grade students' assignments. Students should send a message or an email if they are unable to turn in their homework by then with a valid explanation of why they will not be able to turn in their homework by the deadline, and the teachers will come up with a possible solution. This also applies to missing a class. Course projects will also be assigned and submitted through Google Classroom. If a student misses an assignment deadline repeatedly an email will be sent to his/her parents.