4

COSC 455

Programming Languages: Design and Implementation

Fall 2025

**Lab Assignment #2**

*You’re off to great places! Today is your day! Your mountain is waiting,*

*So get on your way!*

* Dr. Seuss

**Name:** Blessing Abumere **/ 10**

**Goals:** This lab consists of 5 required tasks. The intention of this lab exercise is to set up the development environment and tools that we will be using throughout most of the semester, including Slack (<https://slack.com/>), GitHub (<https://github.com/>), Visual Studio Code (<https://code.visualstudio.com/>) with Rust development tools (<https://code.visualstudio.com/docs/languages/rust>). In doing so, you should start to develop some familiarity with an industry-like development environment and tools and write and commit your very first Rust program to your GitHub account.

**Environment:** The Visual Studio Code IDE using the Rust development tools and cargo must be used in this lab along with Slack to communicate and submit portions of the lab. No other environment, programming language or submission will be allowed.

**Submission:** All lab exercises should be submitted via Slack/GitHub as indicated in the lab description. In addition,

**Deadline:** Submitted via Slack/GitHub by 11:59pm on Wednesday, [September](https://www.youtube.com/watch?v=rdpBZ5_b48g) 10, 2025

1. **Slack Installation & Introduction.** (1 points) Slack (<https://slack.com/>) is a widely used, cloud-based collaboration and communication/messaging tool used throughout the software development industry. This semester, we will adopt this tool to disseminate and discuss lab and project questions/issues and utilize it for students to collaborate and discuss anything pertaining to the class.

Slack is a freely available tool that can be installed on all platforms, including mobile/tablet devices. To install Slack on your own computer/device, you will want to follow the instructions at <https://slack.com/downloads>. All department computers will also have access to Slack but may have to be manually installed. To do so, navigate to the C:\cs\etc\slack directory and run through the installation setup from the \_SlackSetup-v2-1-0.exe file. You should also be able to access the Slack channel on a web browser at <https://cosc455dehlinger.slack.com/>.

Once installed, you can sign up for the course’s Slack channel for this course using your @students.towson.edu email address at <https://join.slack.com/t/cosc455dehlinger/shared_invite/zt-3c2028olu-7QDJ5qelmYJW0iIBoeTlJQ> and join the channel. When signing up and joining, please keep your name, username, picture, etc. appropriate and informative! To understand how to use Slack effectively, I recommend that you read through the Getting Started guide at <https://get.slack.help/hc/en-us/articles/218080037-Getting-started-for-new-users>. If [you don’t like reading](https://media.makeameme.org/created/reading-hate-reading.jpg), you can watch how to use Slack at <https://www.youtube.com/playlist?list=PLWlXaxtQ7fUb1WqLJDqJFGQsAXU7CjoGz>. In particular, you should know how to send direct messages (<https://get.slack.help/hc/en-us/articles/212281468-Direct-messages-and-group-DMs>), share files (<https://get.slack.help/hc/en-us/articles/218074537-Sharing-Files>) and post code snippets (<https://get.slack.help/hc/en-us/articles/204145658-Creating-a-snippet)>. Throughout the semester, I ***strongly*** encourage you to use Slack to communicate – I (or other students) will be able to answer your questions over Slack much easier and quicker than I will over email, and it will act as a repository of questions/answers that other students may have asked previously that can help you.

I have created a channel specific for this lab (#lab2), as I will do so for all labs and projects in the future. Please join the channel and make use of it to communicate with other students (appropriately! and minding the Academic Honor Code!). I have also created other channels (e.g., #random, #rust-resources, etc.) for you to use. I will dump Rust links and other resources under the #rust-resources channel for you to use throughout the semester.

Once you have played around with Slack and understand how to use it, ***you must send me @josh) a direct message on Slack indicating that you have completed this step.*** Without this, you will not receive credit for this portion of the lab.

1. **Visual Studio Code & Rust Installation & Introduction.** (1 points) For this lab (and most subsequent labs and projects, we will be using the Visual Studio Code IDE and Rust for our development environment and programming language. To install VS Code on your own computer, you should download/install it from <https://code.visualstudio.com/> and follow the installation instructions.

To be able to use Rust, we will need to install the Rust tooling. To do so, follow the steps at <https://code.visualstudio.com/docs/languages/rust> to install Rustup, rust-analyzer, and cargo – the main tooling needed for Rust. Per programming tradition when learning a new language, the first program we will create is the [Hello, World](https://en.wikipedia.org/wiki/%22Hello,_World!%22_programhttps://en.wikipedia.org/wiki/%22Hello,_World!%22_program) program in Rust; note the Rust installation link also walks you through this process.

Once you have entered this, you can run this (not very exciting) program by right-clicking on the Run button, not too hard, right? ☺ Of course, this is a very simple Rust program, but you will find that Rust is not very that from Java.

Once you have played around with VS Code and Rust and have an initial understanding on how to use the IDE and some initial basics of Rust, ***post a picture of your code/output to the #lab1 Slack channel to brag to everyone that you finished.*** Without this, you will not receive credit for this portion of the lab.

1. **Task 3 – Git for Version Control.** (Optional, but \***highly**\* recommended if you’ve not done it before!) Version control is an essential skill to understand for any software developer. Any/all software developer groups/companies will use some kind of version control (e.g., Git, Subversion, Mercurial, etc.) and it will be one of the first things you need to learn as an intern/full-time software developer. For software developers, version control systems allow you to keep track of different versions of software components. Throughout the semester, I encourage you to only use the command line Git tooling; this will provide you a better understanding of version control and additional command line experience. To become more familiar with configuration management and Git, I highly encourage you to go through one of the excellent, free Git tutorials, including <https://www.linkedin.com/learning/git-essential-training-19417064/get-started-with-git?u=56674649>, <https://www.w3schools.com/git/>, <https://gist.github.com/jaseemabid/1321592>, and/or <https://gitimmersion.com/lab_01.html>.
2. **GitHub Setup, Introduction & Committing Hello, World!.** (2 points) In order to give you a realistic experience in using a configuration management tool, each student should set up a GitHub account project that will be used to host and check in / check out your code throughout the semester. You will first need a GitHub account. To do so, you should set up a new GitHub account at <https://github.com/>. Students are eligible to apply for a GitHub student pack (<https://education.github.com/)> that includes access to several real-world software development tools. As you sign up, I highly encourage you to go through the Read the Guide ([https://docs.github.com/en/get-started/start-your-journey/hello-world)](https://guides.github.com/activities/hello-world/)) link that is offered so that you understand the idea of software repositories. Additional explanation of software repositories and, more specifically Git/GitHub, I strongly encourage you to read through <http://readwrite.com/2013/09/30/understanding-github-a-journey-for-beginners-part-1/>.

If you don’t have Git installed (you can quickly check within VS Code by going to the terminal tab and type *git*), you will need to follow the installation instructions at

<https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>. A full guide to sharing your code on a Git repository from IntelliJ can be found at <https://documentation.magnolia-cms.com/display/WIKI/Working+with+IntelliJ+IDEA+and+Git>.

With your GitHub repository and the developed Rust code from Step 2, it is vital to be able to commit your code to a repository and clone it from any computer – that way, your [dog will never eat your homework](https://blog.yacapaca.com/wp-content/uploads/2004/03/dog.png) again and you can work on your code from any computer, anytime. A full guide to sharing your code on GitHub from VS Code can be found at <https://code.visualstudio.com/docs/sourcecontrol/github> and/or <https://code.visualstudio.com/docs/sourcecontrol/intro-to-git>. With your GitHub account and within VS Code, you should be able to create a new public or private repository.

Once you are ready, you should commit and push your Hello Rust project to your GitHub account and confirm, on the website, that your code has been committed to the repository. Finally, you will need to add me as a collaborator/member to your labs and projects so that I can access your code. This must be done on the GitHub website through the repository Settings > Collaborators and add me (cosc455dehlinger@gmail.com) as a Developer so that I can access your project remotely – this is the only way that I will be able to help you with technical questions and/or grade your labs and projects!

Once you have successfully committed and pushed your project to GitHub and added me as a developer, ***please send me a direct message over Slack that includes your GitHub repository web address to let me know that you finished.*** Without this, you will not receive credit for this portion of the lab.

1. **Starting the Rust Learning Curve Climb.** (6 points) As mentioned in class, this semester will utilize the Rust programming language for a significant project implementing some aspects of a compiler/parser and will be used throughout to demonstrate various programming language aspects. While I will not directly teach Rust, we will (yes, I am suffering along with you ☺) utilize the excellent Rust Essential Training LinkedIn Learning course, found at <https://www.linkedin.com/learning/rust-essential-training>. Note that LinkedIn Learning is completely free to all TU students (you must sign in using your TU email address). This is course is 10+ hours, but much of it assumes very little prior programming experience and I have found that watching/following along at 1.25-1.5x speed works for me. We will also tackle this in smaller chunks over the next few weeks to not overwhelm.

For this lab, please complete video Chapters 1 – 5 (thru Program Control Flow); note that most of Chapter 1 will just be setting up the environment (similar to this lab’s Tasks 1-3). While this is long, it is essentially just converting your Java/Python syntax/understanding into Rust, so much of it can be done at a faster video speed.

While the payoff for completion of this task will mostly come when we get to implementing the simplified compiler/parser later this semester, to demonstrate your completion of this task for now, please utilize the main.rs Rust file and project used in Task 4 to work through the demonstrations/examples in the videos for your understanding. That is, I should see some of the examples in your committed source code from these chapters.

**Once you have completed this task, commit/push your project to the same GitHub repository from Task 4.** Without this, you will not receive credit for this portion of the lab.

**Finally, to indicate you have completed this lab, submit your GitHub repository link in a text/Word document to Blackboard.**

**Note:** There are at least 3 [Easter eggs](https://en.wikipedia.org/wiki/Easter_egg_(media)) (this isn’t one of them) in this lab as well as one in the Course Syllabus. If you find one, Slack me and let me know!