



# Advanced Programming

## Lab 1 of Rust

于仕琪, 王薇



# Topics

- **1. Tools of Rust**
  - rustc
  - cargo
    - new, check, build, run, test
- **2. Practices**



# Installation

Install rust in wsl, run the following commad in terminal of wsl:

```
curl --proto 'https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
```

after installation, using following command to check the installation is ok or not

```
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST$ cargo --version
cargo 1.86.0 (adf9b6ad1 2025-02-28)
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST$ rustc --version
rustc 1.86.0 (05f9846f8 2025-03-31)
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST$ rustdoc --version
rustdoc 1.86.0 (05f9846f8 2025-03-31)
```

- ✓ **cargo: Rust's compilation manager, package manager, and general tool.** Cargo can be used to create new projects, build and run programs, and manage external libraries that code depends on.
- ✓ **rustc: Rust compiler**, which can be called through cargo or directly used.
- ✓ **rustdoc: Rust documentation tool.** If the documentation is written in an appropriate format in the code comments, Rustdoc can generate formatted HTML based on them, which can be executed separately or by Cargo as well.



# rustc (rust compiler)

“**rustc** [rust\_souce\_file]” is used to compile the source file(here **main.rs** is the rust\_souce\_file), If no other parameters are used and there are no errors during compilation, an executable file with the same name but no suffix(here **main** is the execueteable file) will be generated .

```
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/rustc_hello_wold$ ls
main.rs
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/rustc_hello_wold$ cat main.rs
//this is a line comment not for rust doc
/*
this is a block comment not for rustdoc
*/
fn main() {
    let mut x:i32 = 3;
    x = true;
    println!("Hello, world!,x: {}",x);
}
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/rustc_hello_wold$ rustc main.rs
error[E0308]: mismatched types
  --> main.rs:7:9
6 |         let mut x:i32 = 3;
   |                        --- expected due to this type
7 |         x = true;
   |         ^^^^^ expected `i32`, found `bool`

error: aborting due to 1 previous error

For more information about this error, try `rustc --explain E0308`.
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/rustc_hello_wold$ ls
main.rs
```

Q. In the yellow box code in the screenshot above, what is the data type of ‘x’, is ‘x’ a variable that can be changed? and how to modify the code to perform forced type conversion here?

```
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/rustc_hello_wold$ ls
main.rs
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/rustc_hello_wold$ cat main.rs
//this is a line comment not for rust doc
/*
this is a block comment not for rustdoc
*/
fn main() {
    println!("Hello, world!");
}
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/rustc_hello_wold$ rustc main.rs
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/rustc_hello_wold$ ls
main main.rs
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/rustc_hello_wold$ ./main
Hello, world!
```



# cargo

## Rust's compilation manager, package manager, and general tool

Cargo commands:

- **build**, **b** Compile the current package
- **check**, **c** Analyze the current package and report errors, but don't build object files
- **clean** Remove the target directory
- **doc**, **d** Build this package's and its dependencies' documentation
- **new** Create a new cargo package
- **init** Create a new cargo package in an existing directory
- **add** Add dependencies to a manifest file
- **remove** Remove dependencies from a manifest file
- **run**, **r** Run a binary or example of the local package
- **test**, **t** Run the tests
- **bench** Run the benchmarks
- **update** Update dependencies listed in Cargo.lock
- **search** Search registry for crates
- **publish** Package and upload this package to the registry
- **install** Install a Rust binary
- **uninstall** Uninstall a Rust binary



# cargo new

```
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST$ tree
.
└── lab1 hello world
    ├── main
    └── main.rs

1 directory, 2 files
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST$ cargo new --bin hello_world
Creating binary (application) `hello_world` package
note: see more `Cargo.toml` keys and their definitions at https://doc.rust-lang.org/cargo/reference/manifest.html
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST$ tree
.
├── hello_world
│   ├── Cargo.toml
│   └── src
│       └── main.rs
└── lab1 hello world
    ├── main
    └── main.rs

3 directories, 4 files
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST$ cd hello_world
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ cat ./src/main.rs
fn main() {
    println!("Hello, world!");
}
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ cat ./Cargo.toml
[package]
name = "hello_world"
version = "0.1.0"
edition = "2024"

[dependencies]
```

generate executable project



# cargo check, cargo build

```
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ tree
.
├── Cargo.toml
└── src
    └── main.rs

1 directory, 2 files
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ cargo check
Checking hello_world v0.1.0 (/mnt/c/Users/sustech/Desktop/RUST/hello_world)
Finished `dev` profile [unoptimized + debuginfo] target(s) in 0.69s
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ tree
.
├── Cargo.lock
├── Cargo.toml
├── src
├── target
│   ├── CACHEDIR.TAG
│   └── debug
│       ├── build
│       ├── deps
│       │   ├── hello_world-0ff18d27cf616650.d
│       │   ├── hello_world-2443d5adfb8dea4e
│       │   ├── hello_world-2443d5adfb8dea4e.d
│       │   └── libhello_world-0ff18d27cf616650.rmeta
│       ├── examples
│       ├── hello_world
│       ├── hello_world.d
│       └── incremental
│           ├── hello_world-319q4a5mdyqdz
│           │   ├── s-h72yampqlx-06xjc81-f1ipf7nmyaqdhvlsy5gs3tmb
│           │   │   ├── dep-graph.bin
│           │   │   ├── query-cache.bin
│           │   │   └── work-products.bin
│           │   └── s-h72yampqlx-06xjc81.lock
│           └── hello_world-3s5grp0hlsc7p
│               ├── s-h72yctilxh-1d8g3di-7e17ehp4enu2vh4mbh7vekmln
│               │   ├── 0nhyfuteblamcsldmeu401pk2.o
│               │   ├── 64v5f7uky6ctm3nu8tv4zfn.d.o
│               │   ├── 7eoqdh5fkytfnfuyv2l3gehd.o
│               │   ├── 8kqzgs81n6ey97h6kfg4uh286.o
│               │   ├── czu7pz315pmqijm65wnc6oeh.o
│               │   ├── de74wtvhkg2dlywlwlp5kipip0.o
│               │   ├── dep-graph.bin
│               │   ├── query-cache.bin
│               │   └── work-products.bin
│               └── s-h72yctilxh-1d8g3di.lock
└── 9 directories, 10 files
```

cargo check: check file without executable file generated.

```
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ cargo build
Compiling hello_world v0.1.0 (/mnt/c/Users/sustech/Desktop/RUST/hello_world)
Finished `dev` profile [unoptimized + debuginfo] target(s) in 1.75s
ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ tree
.
├── Cargo.lock
├── Cargo.toml
├── src
├── target
│   ├── CACHEDIR.TAG
│   └── debug
│       ├── build
│       ├── deps
│       │   ├── hello_world-0ff18d27cf616650.d
│       │   ├── hello_world-2443d5adfb8dea4e
│       │   ├── hello_world-2443d5adfb8dea4e.d
│       │   └── libhello_world-0ff18d27cf616650.rmeta
│       ├── examples
│       ├── hello_world
│       ├── hello_world.d
│       └── incremental
│           ├── hello_world-319q4a5mdyqdz
│           │   ├── s-h72yampqlx-06xjc81-f1ipf7nmyaqdhvlsy5gs3tmb
│           │   │   ├── dep-graph.bin
│           │   │   ├── query-cache.bin
│           │   │   └── work-products.bin
│           │   └── s-h72yampqlx-06xjc81.lock
│           └── hello_world-3s5grp0hlsc7p
│               ├── s-h72yctilxh-1d8g3di-7e17ehp4enu2vh4mbh7vekmln
│               │   ├── 0nhyfuteblamcsldmeu401pk2.o
│               │   ├── 64v5f7uky6ctm3nu8tv4zfn.d.o
│               │   ├── 7eoqdh5fkytfnfuyv2l3gehd.o
│               │   ├── 8kqzgs81n6ey97h6kfg4uh286.o
│               │   ├── czu7pz315pmqijm65wnc6oeh.o
│               │   ├── de74wtvhkg2dlywlwlp5kipip0.o
│               │   ├── dep-graph.bin
│               │   ├── query-cache.bin
│               │   └── work-products.bin
│               └── s-h72yctilxh-1d8g3di.lock
└── 11 directories, 24 files
```

cargo build: generate executable file while not execute



# cargo run, cargo clean

```
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ cat ./src/main.rs
fn main() {
    println!("Hello, world!");
}
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ cargo run
Compiling hello_world v0.1.0 (/mnt/c/Users/sustech/Desktop/RUST/hello_world)
Finished `dev` profile [unoptimized + debuginfo] target(s) in 1.44s
Running `target/debug/hello_world`
Hello, world!
```

```
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ tree
```

```
├── Cargo.lock
├── Cargo.toml
├── src
│   └── main.rs
├── target
│   ├── CACHEDIR.TAG
│   └── debug
│       ├── build
│       ├── deps
│       ├── hello_world-2443d5adfb8dea4e
│       ├── hello_world-2443d5adfb8dea4e.d
│       ├── examples
│       ├── hello_world
│       ├── hello_world.d
│       ├── incremental
│       └── hello_world-3s5grp0hlsc7p
│           ├── s-h72y0rcrr1-1qo1coc-7el7ehp4enu2vh4mbh7vekm
│           ├── 0nhyfuteb1amcsldmeu401pk2.o
│           ├── 64v5f7uky6ctm3nu8tv4zfnd.o
│           ├── 7eoqdh5fktyfnfuyv2lz3gehd.o
│           ├── 8kqzgs81n6ey97h6kfg4uh286.o
│           ├── czu7pz315pmqijjm65wnc6oeh.o
│           ├── de74wtvhkg2dlywlp5kpi0.o
│           ├── dep-graph.bin
│           ├── query-cache.bin
│           ├── work-products.bin
│           └── s-h72y0rcrr1-1qo1coc.lock
```

```
9 directories, 18 files
```

```
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$
```

## cargo run:

generate executable file and run the executable file.

## cargo clean:

remove all the executable file and the intermediate files.

```
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ cargo clean
Removed 31 files, 7.8MiB total
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/hello_world$ tree

├── Cargo.lock
├── Cargo.toml
└── src
    └── main.rs

1 directory, 3 files
```





# cargo test

```
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST$ cargo new --lib lcm_test
  Creating library `lcm_test` package
  note: see more `Cargo.toml` keys and their definitions at https://doc.rust-lang.org/cargo/reference/manifest.html
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST$ tree ./lcm_test
./lcm_test
├── Cargo.toml
└── src
    └── lib.rs

1 directory, 2 files
```

generate library project

```
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/lcm_test$ cat ./src/lib.rs
pub fn add(left: u64, right: u64) -> u64 {
    left + right
}

#[cfg(test)]
mod tests {
    use super::*;

    #[test]
    fn it_works() {
        let result = add(2, 2);
        assert_eq!(result, 4);
    }
}
```

```
• ww2@DESKTOP-4NIH4UK:/mnt/c/Users/sustech/Desktop/RUST/lcm_test$ cargo test
  Compiling lcm_test v0.1.0 (/mnt/c/Users/sustech/Desktop/RUST/lcm_test)
  Finished `test` profile [unoptimized + debuginfo] target(s) in 3.25s
  Running unittests src/lib.rs (target/debug/deps/lcm_test-d86672db69939bab)
```

```
running 1 test
test tests::it_works ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

Doc-tests lcm_test

running 0 tests

test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

Q1. What's the difference between cargo new --lib and cargo new --bin ?

Q2. change "lib.rs" as following code, what happens ? fix it.

```
pub fn add(left:u8, right:u8) ->u8 {
    left+right
}

#[cfg(test)]
mod tests {
    usesuper::*;
    #[test]
    fn it_works() {
        let result=add(2, 2);
        assert_eq!(result, 4);
        let result:u16=add(255,
255).into();
        assert_eq!(result, 510asu16);
    }
}
```



# Exercise1

Write a **Rust** program that reads an integer from the command line and determines whether it is a prime number.

Option 1: Use rustc to compile the source code, and then run the generated program.

Option 2: Use Cargo to manage the source code, compile and run the program.



# Exercise2

2.1 Answer the question on page 4 and page9.

2.2 Please use “cargo test to” test the function “gcd” and function “lcm”, identify the issues in the code and solve them. If necessary, please further improve the test cases and conduct testing.

```
fn gcd(a:u8, b:u8) ->u8
{
    letmut a=a;
    letmut b=b;
    while b!=0 {
        let temp=b;
        b=a%b;
        a=temp;
    }
    a
}
```

```
fn lcm(a:u8, b:u8) ->u8 {
    let gcd_val=gcd(a, b);
    let product=a*b;
    let result:u8=product/gcd_val;

    result
}
```

```
#[test]
fn test_lcm() {

    assert_eq!(lcm(1,4),4);
    assert_eq!(lcm(8,9),72);
    assert_eq!(lcm(8,16),16);
    assert_eq!(lcm(1024,2),1024);
    assert_eq!(lcm(256,2),256);
    assert_eq!(lcm(256,0),0);
}
```