Software Hardware List

Chapter number	Software required (With version)	Free/Proprietary	If proprietary, can code testing be performed using a trial version	If proprietary, then cost of the software	Download links to the software	Hardware specifications	OS required
1 - 12	`rustc` and `cargo`	Free			`rustup.rs`	x64 CPU, 4 GB RAM	Windows, Linux, MacOS
1 - 12	Visual Studio Code	Free			`code.visualstudio.com`	x64 CPU, 4 GB RAM	Windows 7 or later, Linux, or MacOS 10.9+
1 - 12	Rust for Visual Studio Code	Free			`marketplace.visualstudio.com`	x64 CPU, 4 GB RAM	Windows 7 or later, Linux, or MacOS 10.9+

Detailed installation steps (software-wise):

Follow the instructions listed on `rustup.rs` for your operating system to set up the basic tools. Once installed, open a terminal (or command window on Windows) to check whether the required binaries are available by typing `cargo` and confirming with return. The output should roughly match:

```
$ cargo
Rust's package manager

USAGE:
    cargo [OPTIONS] [SUBCOMMAND]

OPTIONS:
    -V, --version Print version info and exit
        --list List installed commands
        --explain <CODE> Run `rustc --explain CODE`
    -v, --verbose Use verbose output (-vv very verbose/build.rs output)
    -q, --quiet No output printed to stdout
        --color <WHEN> Coloring: auto, always, never
        --frozen Require Cargo.lock and cache are up to date
        --locked Require Cargo.lock is up to date
    -Z <FLAG>... Unstable (nightly-only) flags to Cargo, see 'cargo -Z
```

```
help' for details
    -h, --help Prints help information

Some common cargo commands are (see all commands with --list):
    build Compile the current package
    check Analyze the current package and report errors, but don't build
object files
    clean Remove the target directory
    doc Build this package's and its dependencies' documentation
    new Create a new cargo package
    init Create a new cargo package in an existing directory
    run Build and execute src/main.rs
    test 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. Default location is $HOME/.cargo/bin
    uninstall Uninstall a Rust binary

See 'cargo help <command>' for more information on a specific command.
```

Obtain the code and change into the directory with:

```
$ cd path/to/directory/Hands-On-Data-Structures-and-Algorithms-with-Rust
```

The code is provided separately for each chapter, but the entire directory is a cargo workspace with individual projects. Each project is a library project, so in order to run the code it has to be *tested*. Testing will also download and build any dependencies as well as build the projects. Do that by running:

```
$ cargo test
```

Or run each project individually by issuing:

```
$ cargo test -p ch1
```

Alternatively, there are benchmarks to create performance indications. Run those with:

```
$ cargo bench
```

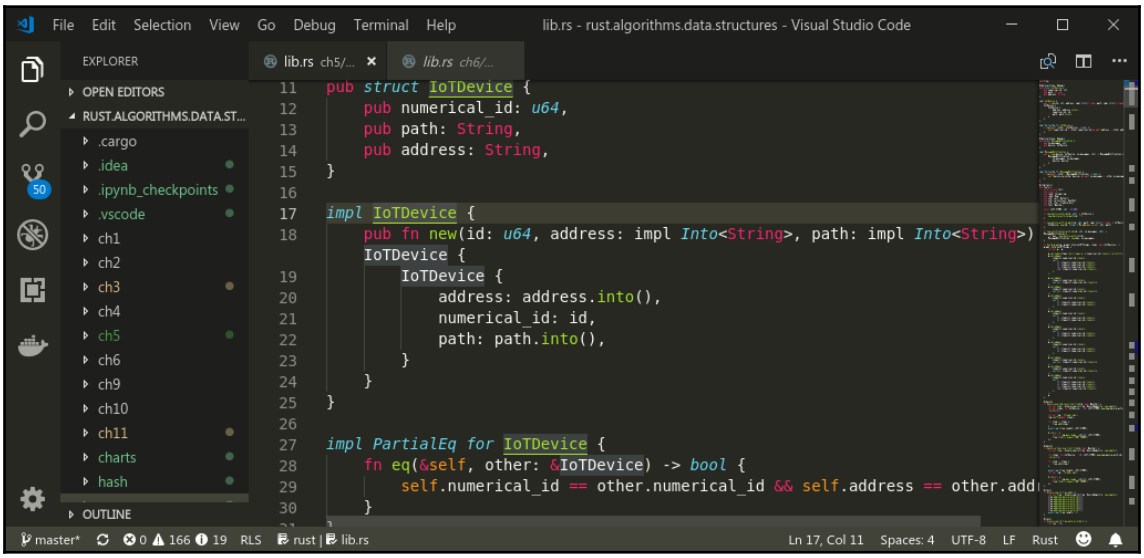
... or

```
$ cargo bench -p ch1
```

respectively.

Start Visual Studio Code and open the directory in order to visually edit and look at the

code, the Rust extension from the marketplace will add syntax highlighting and provide auto-completion. This is what Visual Studio Code could look like when the code workspace is loaded:



Of course, Visual Studio Code is only one editor for Rust, there are others (`https://intellij-rust.github.io/`) that might fit your development habits better.

With that said, edit and re-run the code as needed. Rust is a tricky language and it's easier to learn if you are getting deep into the weeds and try out your ideas. Run the tests and see what your changes did!