Software

This course will make use of the following software:

* Operating system: an x86-64 ABI conforming OS such as many variants of Linux and macOS running on an x86-64 CPU. For [Windows 10 Users](https://www.cs.umd.edu/class/fall2022/cmsc430/Software.html#%28part._.Windows%29), see notes below.
* Racket: the implementation language and source language of our compilers.
* Racket langs package: a package containing utilities for this course.
* NASM: the Netwide Assembler, which we will use to assembly x86 programs.
* GCC: the GNU compiler collection or a GCC-compatible system such as clang.

2 Installing Racket

Racket is available for all major operating systems from:

<https://racket-lang.org/>

We will be using Racket 8.4, but any version from the past several years should work fine.

There are two essential references:

* [The Racket Guide](https://docs.racket-lang.org/guide/) - intended for those new to Racket, i.e. *you!*
* [The Racket Reference](https://docs.racket-lang.org/reference/) - the definitive, comprehensive manual for Racket

Racket is a large, full-featured, batteries-included language platform. However, we will be using only a small subset of Racket. This subset should be easy to pick up for anyone familiar with functional programming. If you’re comfortable with basic OCaml, Haskell, or even JavaScript, you shouldn’t have much trouble learning the Racket bits we will be using.

3 The langs package

After installing Racket, install the langs package which includes course utilities such as the [a86: a Little Assembly Language](https://www.cs.umd.edu/class/fall2022/cmsc430/a86.html) library.

To install, run the following command:

|  |
| --- |
| raco pkg install 'https://github.com/cmsc430/www.git?path=langs#main' |

To test the package works as expected, run:

|  |
| --- |
| raco test -p langs |

All of the tests should pass; if they don’t, consult course staff.

The package source is hosted on Github. To check for and install updates, run:

|  |
| --- |
| raco pkg update langs |

4 IDE

Racket comes with it’s own IDE: DrRacket, which is the recommended way to edit Racket files. We will also be running Racket and its associated tools from the command line.

If you’d like to use Emacs, there’s a good [Racket mode](https://www.racket-mode.com/), but we recommend using DrRacket for a while before switching to Emacs. Using any other editor is fine, too.

5 Windows 10 Users

For Windows 10 users, using WSL for testing is highly recommended. Students in the past have had trouble trying to configure this in the Windows environment, so an easier workaround is simply to enable WSL and run your tests through some Linux Distribution. Here is a breakdown of the steps:

1. Following the instructions at [this link](https://docs.microsoft.com/en-us/windows/wsl/install-win10), install a Linux Distro of your choice (e.g., Ubuntu). The instructions include a suggestion to upgrade to WSL2; this is not necessary but will improve efficiency in general.
2. Open your installed Linux distribution of choice and make any initial configurations necessary (user, pass, etc.). Run sudo apt update and follow with sudo apt upgrade. These two may take some time.
3. Run sudo apt install racket and sudo apt install nasm. These two should cover the necessary installations for this course.
4. Here is where to determine which IDE you would like to use.
	* Using vim (or Emacs as mentioned in the previous section) is simple. Git clone project repos into WSL. Modify files.
	* Previous students preferred installing VSCode (outside of WSL) from [this link](https://code.visualstudio.com/download). For each assignment, git clone somewhere on your Linux distro. For some .rkt file, call ’code some-rkt-file.rkt’ and after some automatic set up, VSCode should load up the file. Install Racket extensions from the VSCode Marketplace (a suggestion will also pop up once you open a .rkt file) to have colorized syntax, bracket matching, inteliSense, etc.
	* If you are intent on using DrRacket, you would also need to install Racket on your local machine (outside WSL). For each assignment, git clone into your normal file system and use DrRacket to edit files accordingly. To access from your Linux subsystem, create a soft symbolic link in your Linux distro to the project directory (or the parent directory so you do not need to make links with each new project).

Regardless of the IDE used, you can now run your tests from your Linux subsystem by entering the project directory and using the raco command.

6 Detailed compatiblity list

The course software has been successfully tested with the following:

* Operating systems:
	+ Ubuntu 20.04
	+ Ubuntu 18.04
	+ Red Hat Enterprise Linux 7.7
	+ macOS 11.0 (Big Sur)
	+ macOS 10.15 (Catalina)
* Racket:
	+ Racket 8.1 [cs]
	+ Racket 8.1 [bc]
	+ Racket 8.0 [cs]
	+ Racket 8.0 [bc]
	+ Racket 7.9 [cs]
	+ Racket 7.9 [bc]
	+ Racket 7.8 [cs]
	+ Racket 7.8 [bc]
* NASM:
	+ NASM version 2.13.02
	+ NASM version 2.15.05
* GCC:
	+ gcc 9.3.0
	+ gcc 7.5.0
	+ Clang/LLVM 12.0.0