CS 261
Fall 2022

Mike Lam, Professor

Command Line Introduction

Linux and the command line

- Linux is a popular operating system in computer science
  - We'll study operating systems in more detail later this semester
  - For now, think of it as the low-level software that manages the hardware
- You interact with your operating system through an **interface**
  - In Windows and macOS, the primary interface is **graphical**
  - In Linux, the primary interface is **textual**

Graphical interfaces   **Textual interface**

"Windows 10 Creators Update" by okubax
is licensed under CC BY 2.0

"about-this-mac" by Wang Liang
is licensed under CC BY-NC 2.0

Why use the command line?

- **More speed**
  - Typing is often quicker than clicking
  - Easy to create shortcuts for common tasks

- **More power**
  - Write custom software / scripts
  - Chain together programs (e.g., w/ pipes)

- **More flexibility**
  - Command-line arguments to customize behavior
  - Work remotely even if your connection is slow

Computer conversations

- **Shell** programs facilitate a command/response conversation with a computer system

  - Example: sh, bash, zsh, PowerShell

- **Terminal** programs provide a visual interface

  - Examples: Terminal, iTerm2, gnome-terminal

- **SSH clients** provide remote access

  - Examples: ssh, PuTTY, MobaXterm

- **Text editors** allow you to edit files

  - Examples: nano, emacs, vim

Context is important

- In a shell, the conversation context is the working directory
  - This is your current location in the file system (also called '.')
  - Use 'pwd' to see the full path
  - Use 'ls' to list the files in the working directory
  - Other useful commands:

```
cd <dir>             "change directory" to the given directory name
cd ..                change to the parent of the current directory
cd                   change to your home directory
less <file>          view a long file one screen at a time
cp <src> <dest>      copy a file from "src" to "dest"
mv <src> <dest>      move/rename a file from "src" to "dest"
rm <file>            remove a file
mkdir <dir>          create a new directory
rmdir <dir>          remove a directory (it must be empty first!)
man <command>        read help text about the given command
tar -xvf <file>      extract a "tarball" (similar to a zip file)
nano <file>          open a file in the Nano text editor
```

Command-line text editing

- Recommended: learn emacs or vim
  - Run "`emacs`" and press Ctrl-'h' followed by 't'
  - Run "`vimtutor`"
- Alternative: just use nano
  - Run "`nano <filename>`" to open or create a file
  - Useful key combinations:

Ctrl-O	Save ("Write Out")
Ctrl-X	Exit
Ctrl-W	Search
Ctrl-K	Cut line
Ctrl-U	Paste ("uncut") line
Ctrl-C	See current line number
Ctrl-_	Go to a specific line number

JMU CS student server

- "stu" is a server running Linux that is available to all JMU CS students
  - Fully-qualified name: `stu.cs.jmu.edu`
  - Pre-configured for CS 261
  - Use for all project distribution and submission
  - You should already have an account
    - Username is your eID
    - Password is the same as your eID password
    - For technical support, email `cs-sysadmin@jmu.edu`
      - **(also CC me!)**

Accessing stu

- Windows
  - Run PowerShell (or download PuTTY or MobaXterm)
- macOS
  - Run Terminal (or download iTerm2)
- Debian-based Linux (e.g., Ubuntu, Mint)
  - Run gnome-terminal
- Run this command:
  - `ssh lam2mo@stu.cs.jmu.edu`   *(use your eID!)*

More info here: `wiki.cs.jmu.edu/student/stu/basics`

Working on CS 261 projects

- **New for 2022**: support for Visual Studio Code
  - Install VS Code on your local computer
    - `https://code.visualstudio.com`
  - Install Remote SSH extension
  - Set up password-less SSH access to stu
  - Connect to stu in VS Code
    - Submit on stu via the command line
  - Video tutorials:
    - `https://youtu.be/-RBHz-Ep6Y0`
    - `https://youtu.be/WQTwtnrk8hc`

"Hello, World" in C

```c
#include <stdio.h>

int main()
{
    printf("Hello, world!\n");
    return 0;
}
```

hello.c

Compiling C programs


```
                                            printf.o
                                               |
hello.c    Pre-      hello.i   Compiler  hello.s  Assembler  hello.o   Linker   hello
  →      processor      →        (cc1)      →       (as)        →       (ld)      →
           (cpp)
Source               Modified           Assembly          Relocatable        Executable
program               source            program             object            object
(text)               program            (text)             programs           program
                      (text)                               (binary)           (binary)
```
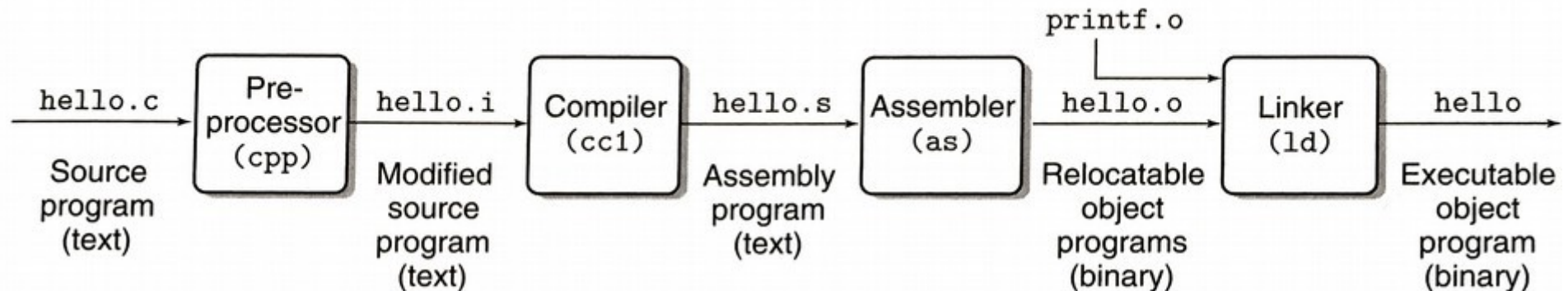
Figure 1.3   The compilation system.

```
linux> gcc -o hello hello.c
```

Here, the GCC compiler driver reads the source file hello.c and translates it into an executable object file hello. The translation is performed in the sequence of four phases shown in Figure 1.3. The programs that perform the four phases (*preprocessor*, *compiler*, *assembler*, and *linker*) are known collectively as the *compilation system*.

To run the program (after compiling): `./hello`

Using a build system

- The compilation process is usually streamlined using a build system (we'll use Make)

  - Provide a text file (must be called "Makefile") that contains individual commands to compile and link the project

  - Run the entire build with one command: "`make`"

- Example Makefile:

target   dependency

```
hello: hello.c
        gcc -o hello hello.c
```

build command

Getting started

- Complete today's command line and compilation lab
- Read or skim the project guide and P0 description
  - Project guide: `w3.cs.jmu.edu/lam2mo/cs261/project_guide.html`
  - P0 description: `w3.cs.jmu.edu/lam2mo/cs261/p0-intro.html`
- Watch and follow along with the P0 setup video
  - Link: `https://youtu.be/NrdZYWmknO4`
- Take the project guide quiz (due Friday)
  - Re-read the project guide and P0 description as needed
- Watch and follow along with the VS Code setup videos
  - `https://youtu.be/-RBHz-Ep6Y0`
  - `https://youtu.be/WQTwtnrk8hc`
- Begin work on P0 (intro to C)
  - **Don't panic!** There's no need to finish it immediately
  - Work on it incrementally over the next couple of weeks as you learn more about C