

EE538: Computing Principles for Electrical Engineers

Discussion 1: Environment Setting

University of Southern California

05/19 Summer 2022

Instructor: Arash Saifhashemi

TA and Mentors: Zixu Wang, Aditi Bodhankar

Introduction



Aditi Bodhankar
Mentor
bodhanka@usc.edu



Zixu (Leon) Wang
Mentor
zixuwang@usc.edu

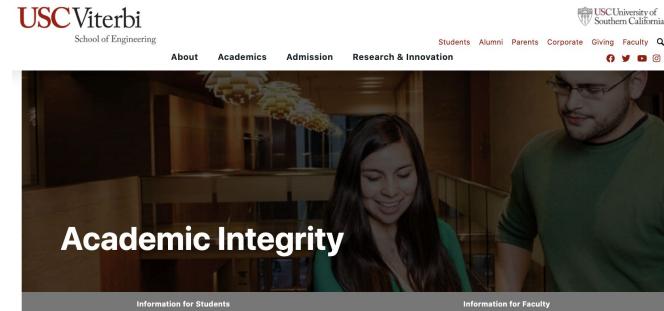
- Office Hours:
 - 2 hours for each Mentor in person
 - Will post to Piazza

The most important thing - Academic Integrity

- University Policy
 - <https://sjacs.usc.edu/students/academic-integrity/>
 - <https://viterbischool.usc.edu/academic-integrity/>

We hope you could...

- Be responsible for yourself!
- Learn some useful knowledge!



The Viterbi Code of Integrity expresses the standards we seek to uphold as Trojans and Viterbi engineers. The resources offered within this site are intended to provide students and faculty help in understanding and maintaining the highest level of academic integrity.

Our goal is to have this site be helpful in meeting daily academic challenges. All members of the Viterbi community are invited to contribute to this site by way of commentary and sharing supporting information. Suggested content can be sent to sbucher@usc.edu.

Statement of Academic Integrity

"A Community of Honor"

We are the USC Viterbi School of Engineering, a community of academic and professional integrity. As students, faculty, and staff our fundamental purpose is the pursuit of knowledge and truth. We recognize that ethics and honesty are essential to this mission and pledge to uphold the highest standards of these principles. As responsible men and women of engineering, our lifelong commitment is to respect others and be fair in all endeavors. Our actions will reflect and promote a community of honor.

Dean's Note

"Excellence in all our endeavors" is part of my vision for the Viterbi School, and our shared endeavors as students and faculty include how we conduct ourselves in the classroom, in the lab, and in all academic activities. Our school is a vibrant cross-section of academic and social cultures, and we look for these experiences to merge into a common principle of high ethical standards. As students and faculty we recognize that our education and research are governed by the principles of USC and the Viterbi School of Engineering.

The Viterbi School is a "Community of Honor" - the actions of one of us represent us all. Rather than having an academic culture driven by rules and warnings, we aspire to a higher standard of conduct, one driven by a lifelong commitment to excellence in all our endeavors.

This site is meant to be a resource for students and faculty, and its

Mask Policy: It is not mandatory but feel free to keep your mask in class

- University Policy
 - <https://www.provost.usc.edu/covid-19/spring-2022-faculty-faq/>
 - <https://sjacs.usc.edu/disruptive-behavior/>

We hope you could...

- Please stay home if you feel sick. We'll help you catch up.

Course Platform

- Piazza - <https://piazza.com/class/l33ln1082u84cd?cid=1>
 - All the lecture material will be posted here!
 - **Homework assignment will be released here!**
- GitHub classroom
 - **Submit your homework through GitHub before the deadline!**
- Blackboard/DEN
 - **Your scores will be posted here!**
- DEN
 - TBA

Outline

- Introduction
- Terminal Basics
 - Operating System
 - Terminal Commands
- GitHub Usage
 - GitHub Account
 - Git Command
- Bazel
- Environment Setup
 - Bazel Installation for Linux / Mac
 - Install Ubuntu on VirtualBox
- Programming Practice

Operating System

- For Linux (Ubuntu) and Mac users
 - You're all set!
- For Windows users, it is recommended to have a unix-like shell environment
 - Install a Linux system on your computer.
 - **Use a Virtual Machine (VM)!**
 - Install VirtualBox <https://www.virtualbox.org/wiki/Downloads>
 - Install Ubuntu system <https://ubuntu.com/>

Visual Studio Code

- IDE(Integrated development environment) help you code faster!
- Please go to <https://code.visualstudio.com/Download> and download VSC

Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.



↓ Windows

Windows 7, 8, 10



↓ .deb

Debian, Ubuntu

↓ .rpm

Red Hat, Fedora, SUSE



↓ Mac

macOS 10.11+

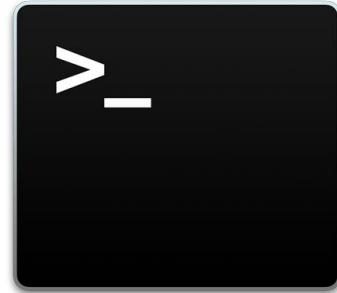
User Installer [64 bit](#) [32 bit](#) [ARM](#)
System Installer [64 bit](#) [32 bit](#) [ARM](#)
.zip [64 bit](#) [32 bit](#) [ARM](#)

.deb [64 bit](#) [ARM](#) [ARM 64](#)
.rpm [64 bit](#) [ARM](#) [ARM 64](#)
.tar.gz [64 bit](#) [ARM](#) [ARM 64](#)

.zip [Universal](#) [Intel Chip](#) [Apple Silicon](#)

[Snap Store](#)

What is Terminal or CLI ?



- Terminal, or the command-line interface (CLI)
 - processes commands to a computer program in the form of lines of text
- CLI used to be the only way to accomplish anything on a computer!
- In this class, we would use CLI as a means of manipulating the computer and get it to perform tasks instead of using a mouse to get things done!

A screenshot of a macOS Terminal window titled "max -- bash -- 80x24". The window shows a welcome message for switching to zsh:

```
Last login: Wed Aug 25 00:04:38 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Maxs-MacBook-Pro:~ max$
```

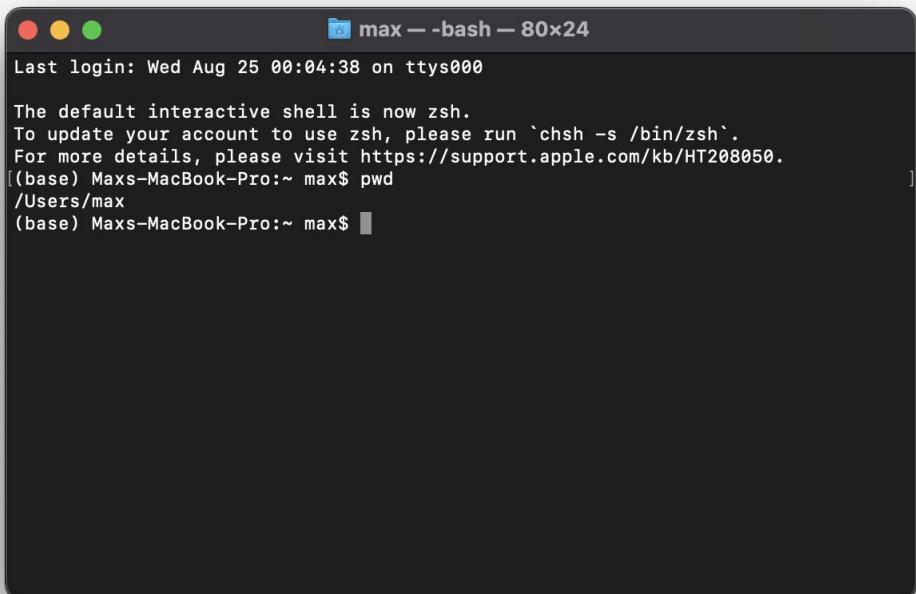
>-

Terminal Commands:

0. Print working directory

Command: **pwd**

Purpose: Print Working Directory



```
Last login: Wed Aug 25 00:04:38 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[(base) Maxs-MacBook-Pro:~ max$ pwd
/Users/max
(base) Maxs-MacBook-Pro:~ max$ ]
```

>-

Terminal Commands:

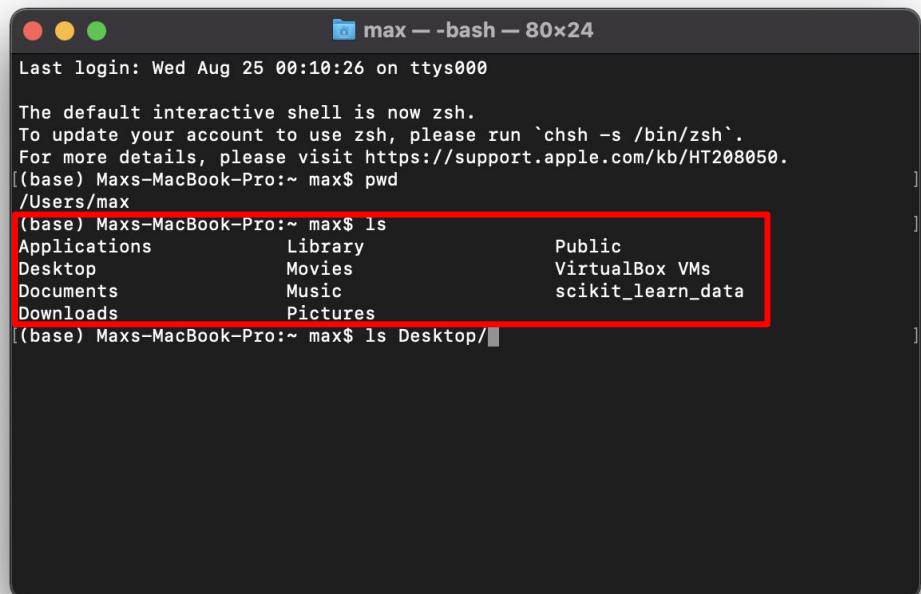
1. Listing directory

Command: **ls**

Purpose: **List the contents of a directory.**

Syntax: **ls “path-to-directory”**

Example: **ls Desktop**



The screenshot shows a terminal window titled "max -- bash -- 80x24". The window has a dark background with light-colored text. It displays the following information:

```
Last login: Wed Aug 25 00:10:26 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[(base) Maxs-MacBook-Pro:~ max$ pwd
/Users/max
(base) Maxs-MacBook-Pro:~ max$ ls
Applications           Library          Public
Desktop                Movies           VirtualBox VMs
Documents              Music            scikit_learn_data
Downloads              Pictures
[(base) Maxs-MacBook-Pro:~ max$ ls Desktop/]
```

A red rectangular box highlights the output of the `ls` command, which lists the contents of the `Desktop` directory: Applications, Library, Public, Desktop, Movies, VirtualBox VMs, Documents, Music, scikit_learn_data, Downloads, and Pictures.

>-

Terminal Commands:

2. Change directory

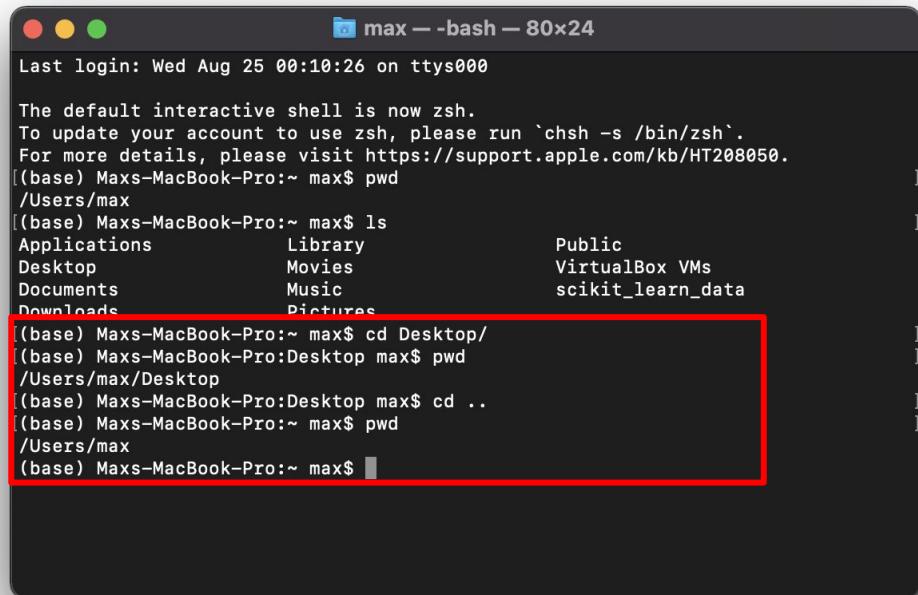
Command: **cd**

Purpose: **Change the Directory**
of the command line path.

Syntax: **cd “path-to-directory”**

Example: **cd Desktop**

cd .. (move up one directory)



```
Last login: Wed Aug 25 00:10:26 on ttys000
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[base] Maxs-MacBook-Pro:~ max$ pwd
/Users/max
[base] Maxs-MacBook-Pro:~ max$ ls
Applications           Library          Public
Desktop                Movies           VirtualBox VMs
Documents              Music            scikit_learn_data
Downloads              Pictures
[base] Maxs-MacBook-Pro:~ max$ cd Desktop/
[base] Maxs-MacBook-Pro:Desktop max$ pwd
/Users/max/Desktop
[base] Maxs-MacBook-Pro:Desktop max$ cd ..
[base] Maxs-MacBook-Pro:~ max$ pwd
/Users/max
[base] Maxs-MacBook-Pro:~ max$
```

>-

Terminal Commands:

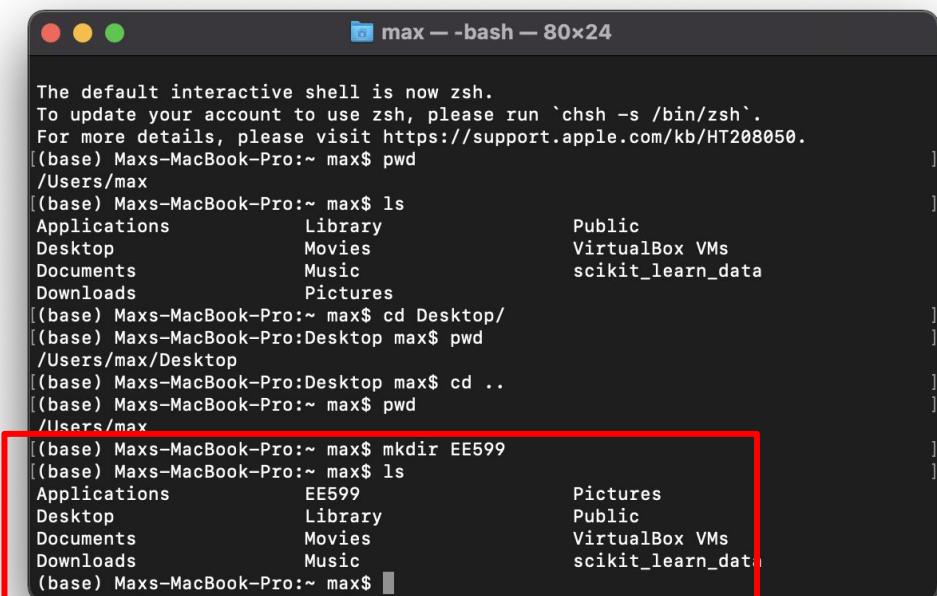
2. Create directory

Command: **mkdir**

Purpose: **Make a new Directory**

Syntax: **mkdir “path”**

Example: **mkdir EE538**



The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit <https://support.apple.com/kb/HT208050>.

```
(base) Maxs-MacBook-Pro:~ max$ pwd  
/Users/max  
(base) Maxs-MacBook-Pro:~ max$ ls  
Applications Library Public  
Desktop Movies VirtualBox VMs  
Documents Music scikit_learn_data  
Downloads Pictures  
(base) Maxs-MacBook-Pro:~ max$ cd Desktop/  
(base) Maxs-MacBook-Pro:Desktop max$ pwd  
/Users/max/Desktop  
(base) Maxs-MacBook-Pro:Desktop max$ cd ..  
(base) Maxs-MacBook-Pro:~ max$ pwd  
/Users/max  
(base) Maxs-MacBook-Pro:~ max$ mkdir EE599  
(base) Maxs-MacBook-Pro:~ max$ ls  
Applications EE599 Pictures  
Desktop Library Public  
Documents Movies VirtualBox VMs  
Downloads Music scikit_learn_data  
(base) Maxs-MacBook-Pro:~ max$
```

>-

Terminal Commands:

4. Copy a file to another directory

Command: cp

Syntax: cp "*filename*" "*path-to-directory*"

5. Move a file

Command: mv

Syntax: mv "*filename*" "*path-to-directory*"

6. Remove a file

Command: rm

Syntax: rm "*filename*"

7. Remove a directory

Command: rm -r

Syntax: rm -r "*path-to-directory*"

8. Execute commands with root privileges

Command: sudo

Syntax: sudo "command"

9. List actively running computer processes

Command: top

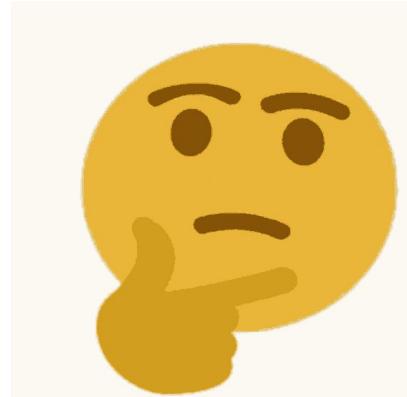
Questions?

1. How to copy contents of one directory to another?

```
cp -r "source directory" "target directory"
```

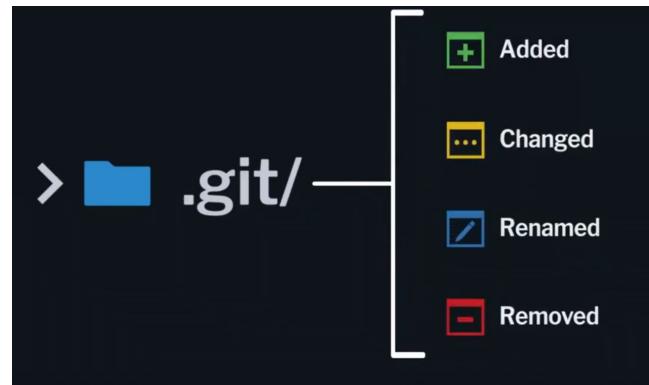
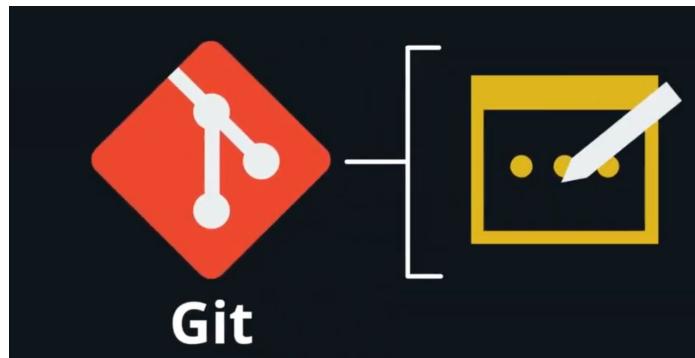
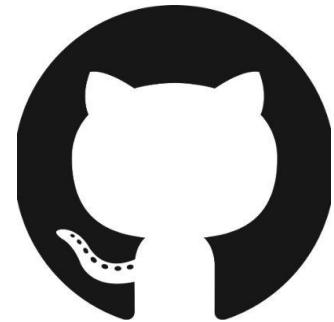
2. How to clear the terminal?

```
clear
```



GitHub Usage

- Distributed version-control system
- Tracks changes you make to your files over the time
- Cooperate with others on the same project



GitHub Account

- Please go to <https://github.com>, register a github account
- Set your real name, **usc email address** as primary public email address

Account settings

Profile

Account

Appearance

Account security

Billing & plans

Security loa

Public profile

Name

Hong-Shuo Chen (Max)

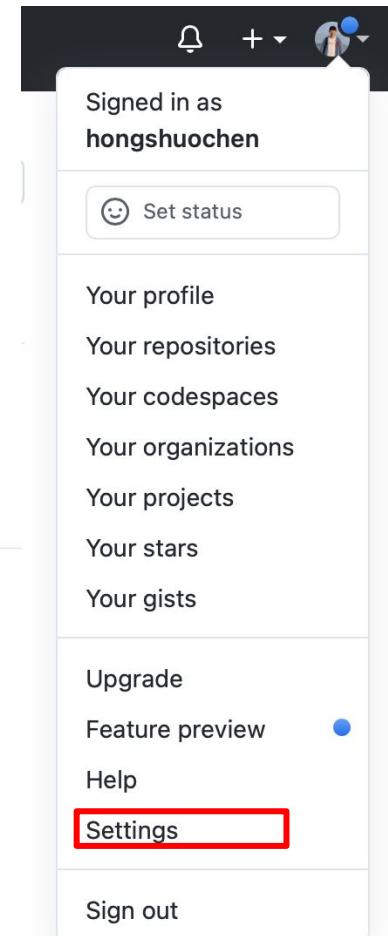
Your name may appear around GitHub where you contribute or are mentioned. You can remove it at any time.

Public email

hongshuo@usc.edu

X Remove

You can manage verified email addresses in your [email settings](#).



GitHub Account

Account settings

Profile

Account

Appearance

Account security

Billing & plans

Security log

Security & analysis

Sponsorship log

Emails

Emails

hongshuo@usc.edu – Primary ⓘ

- Visible in emails ⓘ
- Receives notifications ⓘ



Add email address

Email address

Add

Primary email address

hongshuo@usc.edu will be used for account-related notifications and can be used for password resets.

hongshuo@usc.edu ⓘ

Save

Clone a repository

- git clone <https://github.com/ourarash/cpp-template>

The screenshot shows the GitHub repository page for 'ourarash / cpp-template'. The repository is a fork from 'ratanparai/cpp-template'. The 'Code' tab is selected. The 'master' branch is current, with 1 branch and 0 tags. A message indicates the branch is 58 commits ahead of 'ratanparai:master'. The commit history for 'master' shows the following changes:

Commit	Message	Date	Commits
.vscode	Added test for search	14 months ago	66
src	Added gmock	4 months ago	
tests	Added gmock	4 months ago	
third_party	Added glog	15 months ago	

Create a New Repository

The screenshot shows a user profile for "hongshuochen" with a dropdown menu. Below it, there's a "Repositories" section with a "New" button and a search bar. A large "Find a repository..." placeholder is present.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Repository template

Start your repository with a template repository's contents.

No template ▾

Owner *



Repository name *

EE538



Great repository names are short and memorable. Need inspiration? How about [vigilant-journey](#)?

Description (optional)

Repository for EE538

Public

Anyone on the internet can see this repository. You choose who can commit.

Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

Add a README file

This is where you can write a long description for your project. [Learn more](#).

Add .gitignore

Choose which files not to track from a list of templates. [Learn more](#).

Choose a license

A license tells others what they can and can't do with your code. [Learn more](#).

Create repository

Create a New Repository

Quick setup — if you've done this kind of thing before

 Set up in Desktop

or

HTTPS

SSH

<https://github.com/hongshuochen/EE538.git>



Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# EE538" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/hongshuochen/EE538.git
git push -u origin main
```



Copy and paste to Terminal under EE538 directory!

Markdown basics

- Basics - [Markdown-guide](#)

- Add summary and details for the codes
- Insert both code and text for referencing

Inline `code` has `back-ticks around` it.

Inline `code` has back-ticks around it.

Footnotes

Footnotes aren't part of the core Markdown spec, but they [supported by GFM](#).

Here is a simple footnote^[^1].

A footnote can also have multiple lines^[^2].

You can also use words, to fit your writing style more closely^[^note].

[^1]: My reference.

[^2]: Every new line should be prefixed with 2 spaces.

This allows you to have a footnote with multiple lines.

[^note]:

Named footnotes will still render with numbers instead of the text but allow easier identification

This footnote also has been made with a different syntax using 4 spaces for new lines.

- Reference any materials used

Basic operation and workflow

- Clone the repository or Create the repository
- Make changes to files!
- Add your changes
 - `git add "filename"` (e.g. `git add test.cc`)
 - `git add "directory"` (e.g. `git add src`)
 - `git add .` (add all changes in the current folder)
- Commit changes with a message, like creating a snapshot of your repository
 - `git commit -m "update files"`
- Push local changes to GitHub
 - `git push`
- Pull changes from GitHub
 - `git pull`

Questions?

1. Use which command to download the repository from GitHub?

ANS: `git clone`

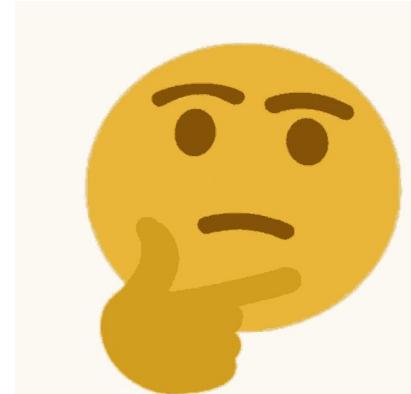
2. Use which two commands to create a version your current changes?

ANS: `git add`

`git commit`

3. Use which command to upload your changes to GitHub?

ANS: `git push`



Bazel

- Bazel is a tool to build and test your code, similar to “Make”
- Build and test software of any size, quickly and reliably

Trusted by industry leaders



Bazel Installation on MacOS

- <https://docs.bazel.build/versions/4.2.2/install-os-x.html#install-on-mac-os-x-homebrew>

Installing using Homebrew

Step 1: Install Homebrew on macOS

Install Homebrew (a one-time step):

```
/bin/bash -c "$(curl -fsSL \
https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

Step 2: Install Bazel via Homebrew

Install the Bazel package via Homebrew as follows:

```
$ brew install bazel
```

All set! You can confirm Bazel is installed successfully by running the following command:

```
$ bazel --version
```

Once installed, you can upgrade to a newer version of Bazel using the following command:

```
$ brew upgrade bazel
```

Bazel Installation on Ubuntu

- <https://docs.bazel.build/versions/4.2.0/install-ubuntu.html>

Using Bazel's apt repository

Step 1: Add Bazel distribution URI as a package source

Note: This is a one-time setup step.

```
sudo apt install curl gnupg
curl -fsSL https://bazel.build/bazel-release.pub.gpg | gpg --dearmor > bazel.gpg
sudo mv bazel.gpg /etc/apt/trusted.gpg.d/
echo "deb [arch=amd64] https://storage.googleapis.com/bazel-apt stable jdk1.8" | sudo tee /etc/apt/sources.list.d/bazel.list
```

The component name “jdk1.8” is kept for legacy reasons only and doesn’t relate to supported or included JDK versions anymore. In the past, when Bazel did not yet bundle a private JRE, we had two release versions, one compatible with JDK 7 and one with JDK 8. However, since we dropped Java 7 support and started bundling a private runtime, Bazel releases are Java version agnostic. Changing the “jdk1.8” component name would break existing users of the repo though.

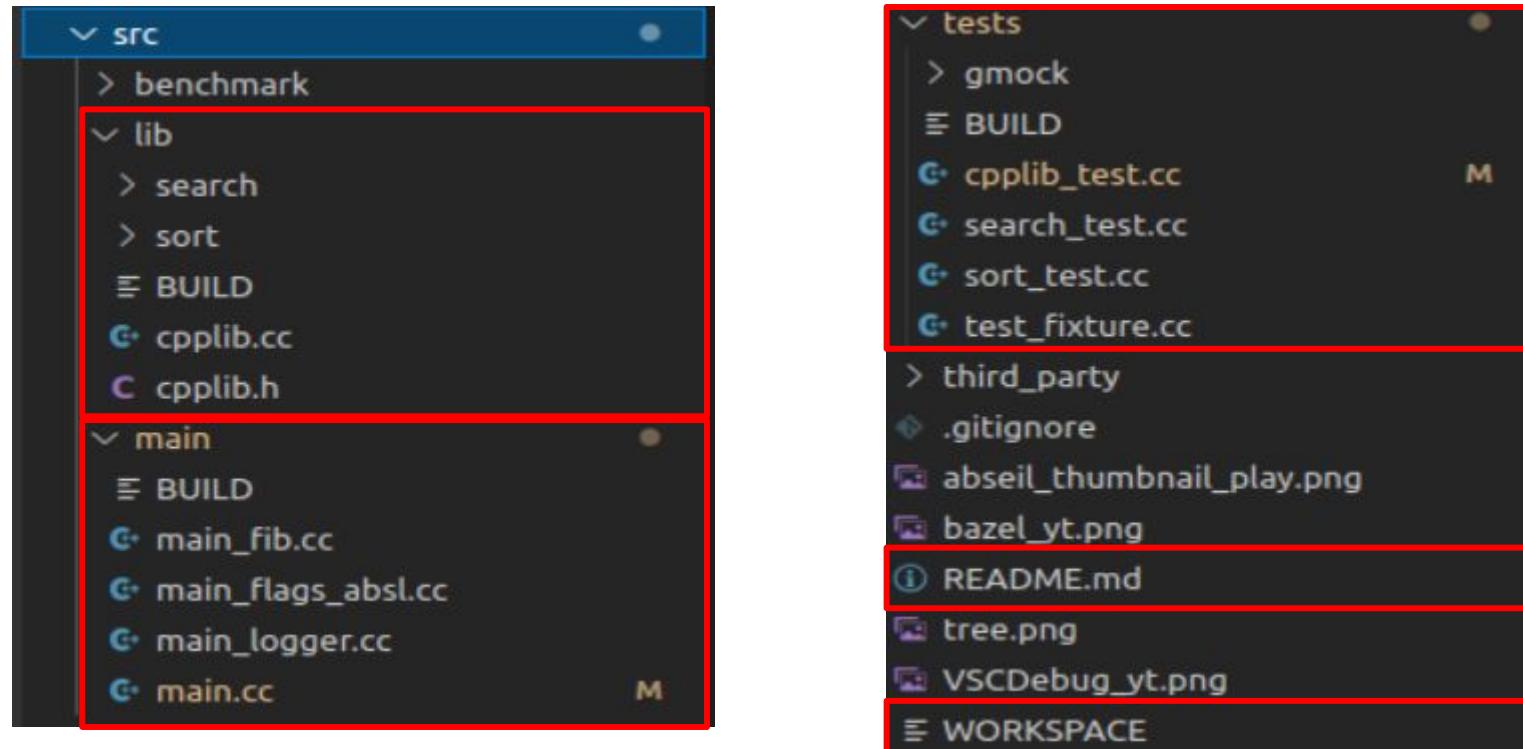
Step 2: Install and update Bazel

```
sudo apt update && sudo apt install bazel
```

Once installed, you can upgrade to a newer version of Bazel as part of your normal system updates:

```
sudo apt update && sudo apt full-upgrade
```

Bazel project structure



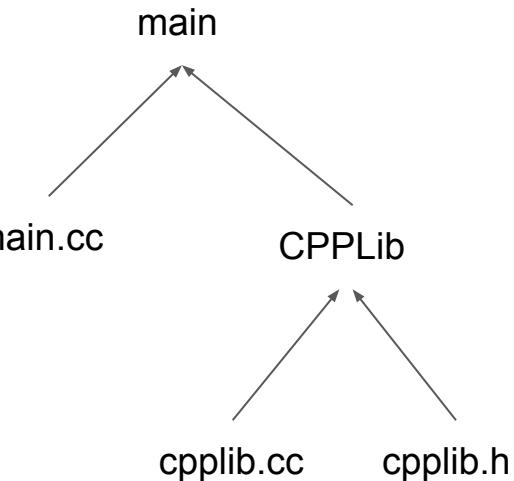
Understand BUILD file

- BUILD file is telling Bazel how to compile codes and headers of this project.
 - We need to use cc_binary, cc_library, cc_test
- Example on: /src/main/BUILD
- The BUILD file looks like:

```
cc_binary(  
    name = "main",  
    srcs = ["main.cc"],  
    deps = ["//src/lib:CPPLib"],  
)
```

```
cc_library(  
    name = "CPPLib",  
    srcs = ["cpplib.cc"],  
    hdrs = ["cpplib.h"],  
    visibility = ["//visibility:public"],  
)
```

- Bazel will then compile and build the executable binary
- To run the binary file, run `bazel run src/main:main`



src/main/main.cc

```
#include <iostream>

#include "src/lib/cplib.h"

int main() {
    // Print Hello world!
    CPPLib s;
    std::cout << s.PrintHelloWorld() << std::endl;

    // std::cout << s.fib(6) << std::endl;
    // std::cout << s.fib(4) << std::endl;
    // std::cout << s.fib(8) << std::endl;
    return 0;
}
```

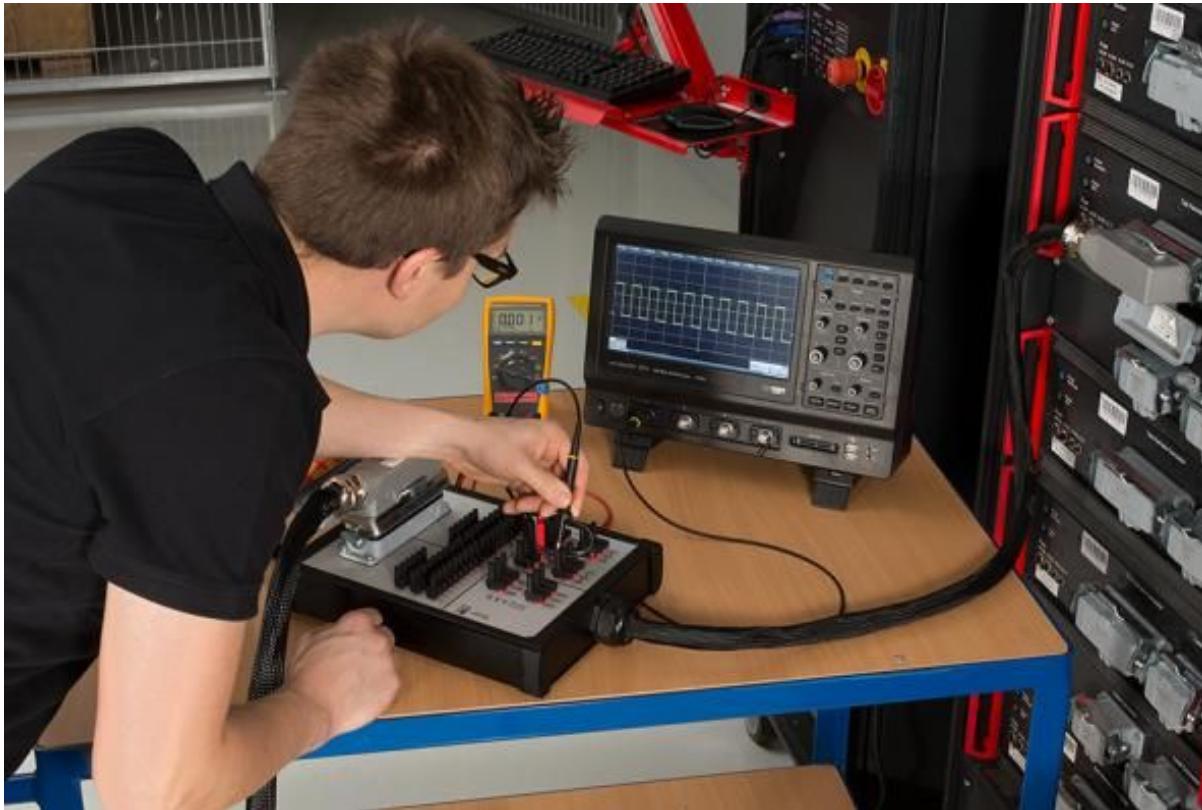
Bazel Commands

- Run

- **bazel run src/main:main**

```
ee538@ee538-VirtualBox:~/Desktop/cpp-template$ bazel run src/main:main
INFO: Analyzed target //src/main:main (0 packages loaded, 0 targets configured).
INFO: Found 1 target...
Target //src/main:main up-to-date:
  bazel-bin/src/main/main
INFO: Elapsed time: 0.131s, Critical Path: 0.01s
INFO: 1 process: 1 internal.
INFO: Build completed successfully, 1 total action
INFO: Build completed successfully, 1 total action
**** Hello World ****
```

Test



GoogleTest

- Refer to this video:
 - <https://youtu.be/0wMNtl2xDT0>
 - <https://www.youtube.com/watch?v=JJqRISTQlh4>
- A testing tool developed by Google is widely used in many other places.
- Purpose: independently test whether each function or class is working as expected.

In /tests/cplib_test.cc file:

In /tests/BUILD file:

```
cc_test(  
    name = "cplib_test",  
    srcs = ["cplib_test.cc"],  
    deps = [  
        "//src/lib:CPPLib",  
        "@com_google_gtest//:gtest_main",  
    ],  
)
```

- To test, run `bazel test tests/gtest_demo:cplib_test`

```
#include "src/lib/cplib.h"  
  
#include <map>  
#include <vector>  
  
#include "gtest/gtest.h"  
  
TEST(CPPLibTest, ReturnHelloWorld) {  
    CPPLib cpplib;  
    std::string actual = cpplib.PrintHelloWorld();  
    std::string expected = "**** Hello Wold ****";  
    EXPECT_EQ(expected, actual);  
}
```

GoogleTest

- Find more assertions on this link
- <https://google.github.io/googletest/reference/assertions.html>
- Basic Assertions

These assertions do basic true/false condition testing.

Fatal assertion	Nonfatal assertion	Verifies
ASSERT_TRUE(condition);	EXPECT_TRUE(condition);	condition is true
ASSERT_FALSE(condition);	EXPECT_FALSE(condition);	condition is false

```
TEST(CPPLibTest, ReturnHelloWorld) {
    CPPLib cpplib;
    std::string actual = cpplib.PrintHelloWorld();
    std::string expected = "**** Hello Wold ****";
    EXPECT_EQ(expected, actual);
}
```

- Binary Assertions

Fatal assertion	Nonfatal assertion	Verifies
ASSERT_EQ(val1, val2);	EXPECT_EQ(val1, val2);	val1 == val2
ASSERT_NE(val1, val2);	EXPECT_NE(val1, val2);	val1 != val2
ASSERT_LT(val1, val2);	EXPECT_LT(val1, val2);	val1 < val2
ASSERT_LE(val1, val2);	EXPECT_LE(val1, val2);	val1 <= val2
ASSERT_GT(val1, val2);	EXPECT_GT(val1, val2);	val1 > val2
ASSERT_GE(val1, val2);	EXPECT_GE(val1, val2);	val1 >= val2

Bazel Commands

- Test

- `bazel test tests/gtest_demo:cpplib_test`

```
ee538@ee538-VirtualBox:~/Desktop/cpp-template$ bazel test tests:cpplib_test
...
INFO: Analyzed target //tests:cpplib_test (11 packages loaded, 316 targets configured).
INFO: Found 1 test target...
Target //tests:cpplib_test up-to-date:
  bazel-bin/tests/cpplib_test
INFO: Elapsed time: 15.922s, Critical Path: 1.84s
INFO: 29 processes: 10 internal, 19 linux-sandbox.
INFO: Build completed successfully, 29 total actions
//tests:cpplib_test                                              PASSED in 0.0s

Executed 1 out of 1 test: 1 test passes.
INFO: Build completed successfully, 29 total actions
```

Install Ubuntu on VM

Step 1: Download and Install VirtualBox

<https://www.virtualbox.org/wiki/Downloads>

VirtualBox 6.1.26 platform packages

- [Windows hosts](#)
- [OS X hosts](#)
- [Linux distributions](#)
- [Solaris hosts](#)
- [Solaris 11 IPS hosts](#)

Install Ubuntu on VM

Step 2 Download Ubuntu

<https://ubuntu.com/#download>

[Ubuntu Desktop ›](#)

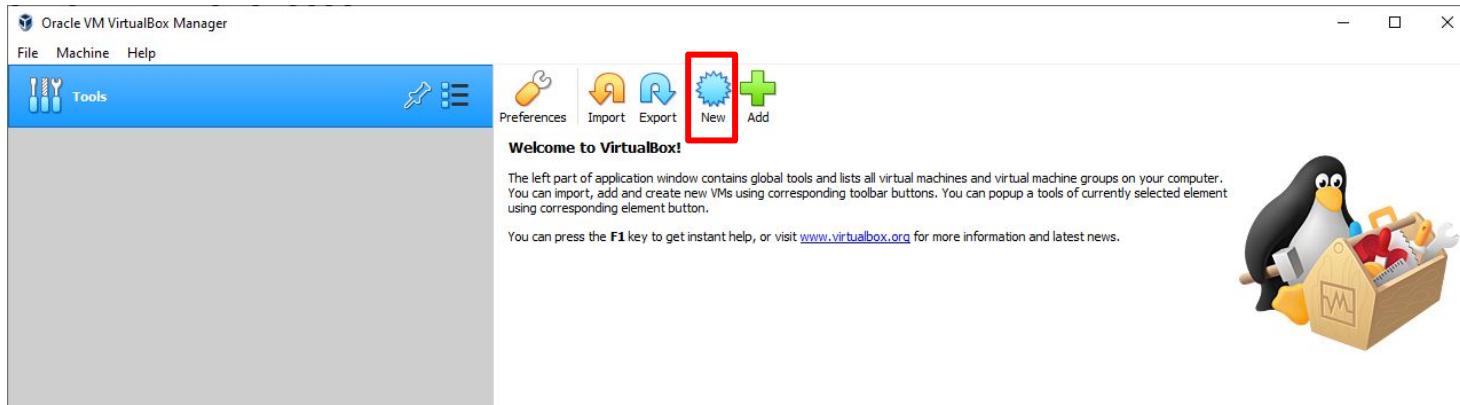
Download Ubuntu desktop and
replace your current operating
system whether it's Windows or Mac
OS, or, run Ubuntu alongside it.

20.04 LTS

21.04

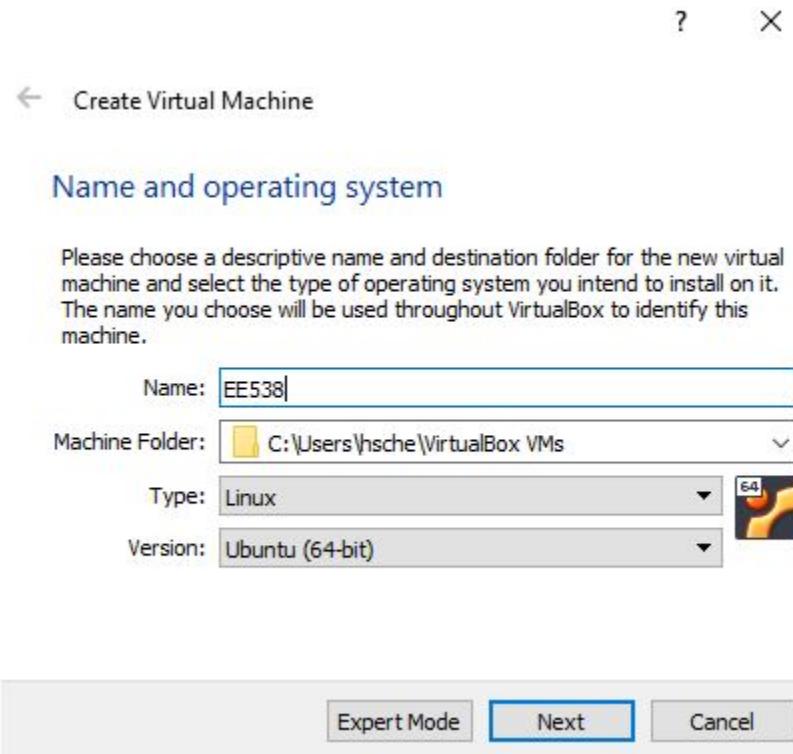
Install Ubuntu on VM

Step 3 Open VirtualBox and press New



Install Ubuntu on VM

Step 4

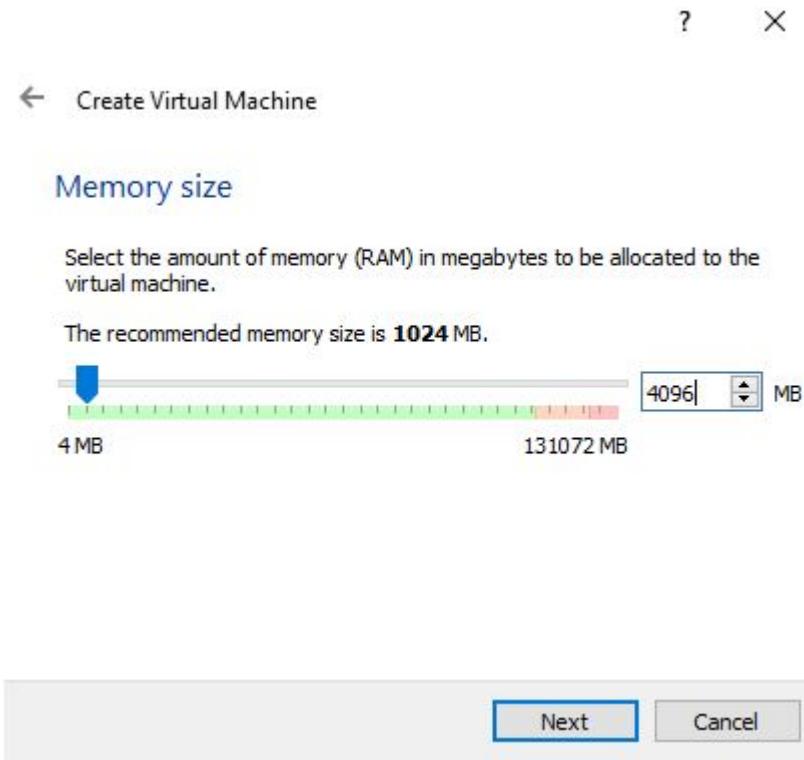


Install Ubuntu on VM

Step 5 Larger Memory is Faster

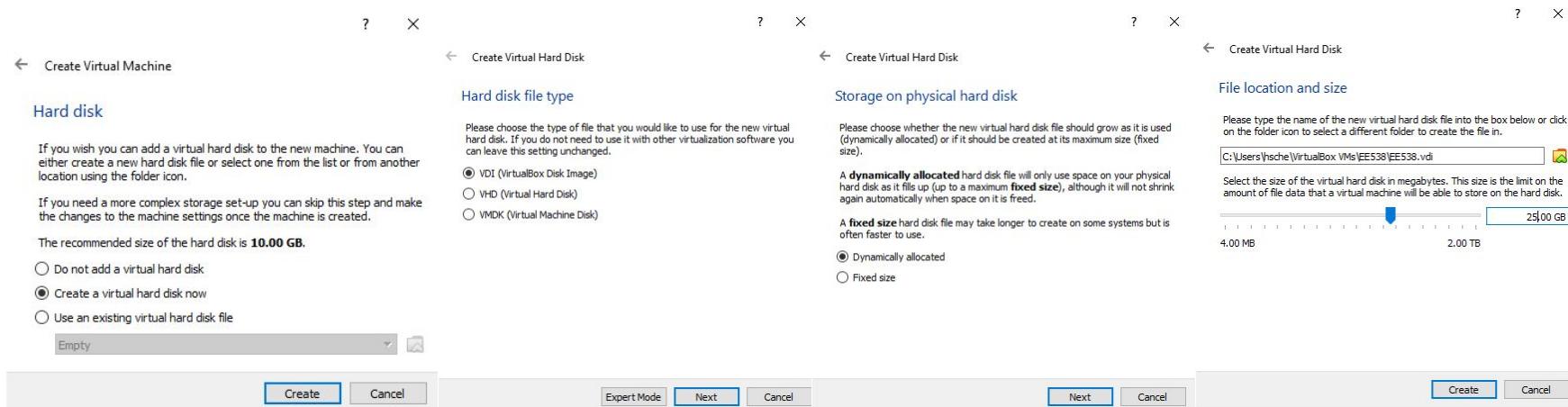
Memory should not exceed

1/3 of the total amount of memory of
your host machine!

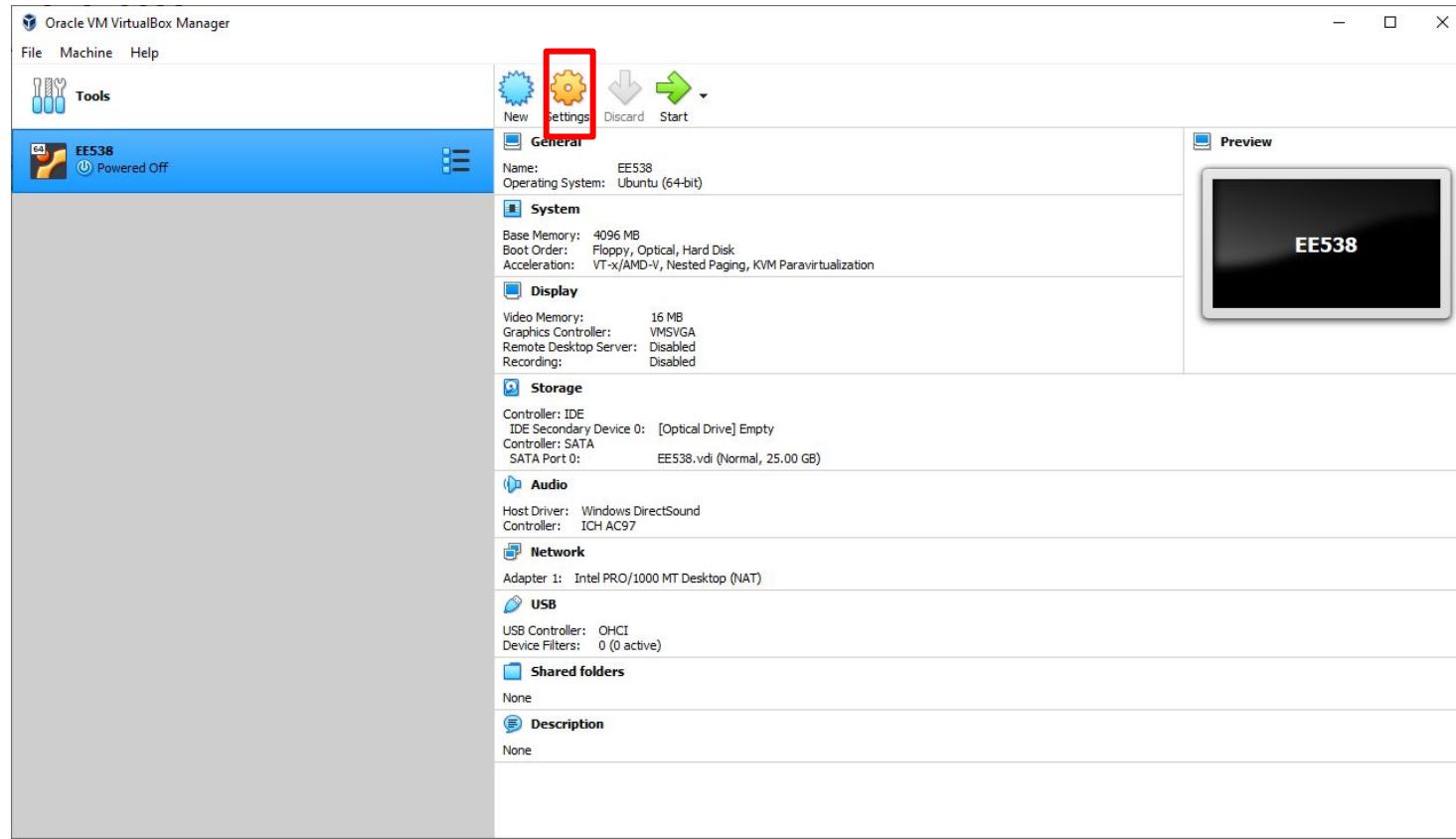


Install Ubuntu on VM

Step 6 Disk size: 25 GB



Install Ubuntu on VM



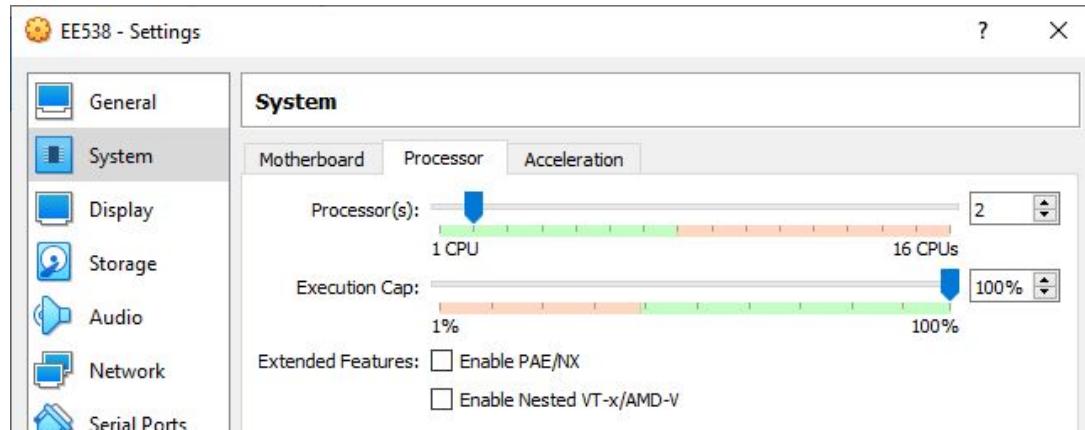
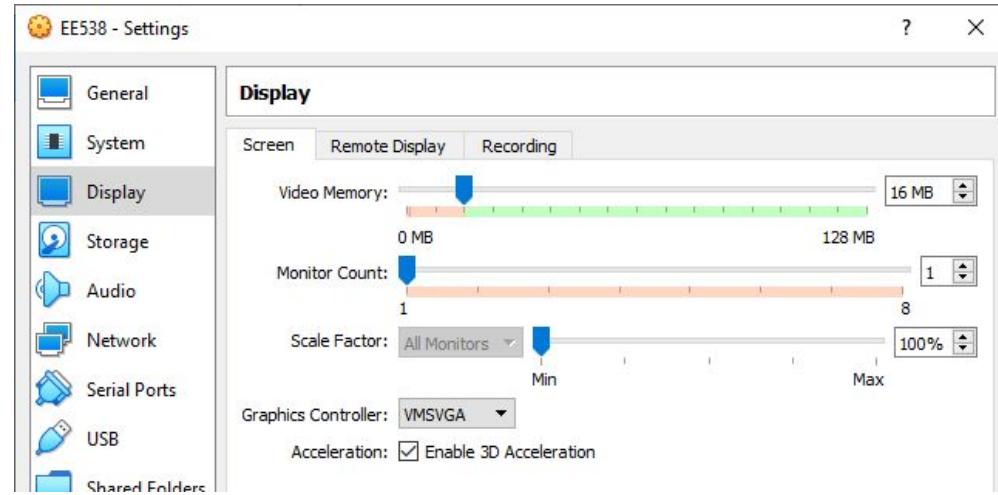
Install Ubuntu on VM

Step 7

Enable 3D acceleration

Make sure the VM

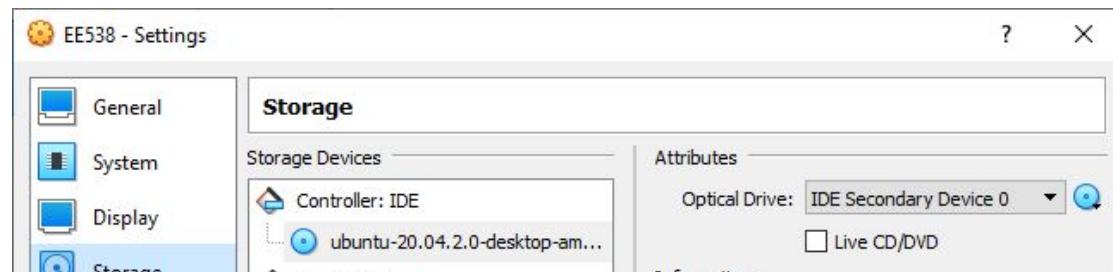
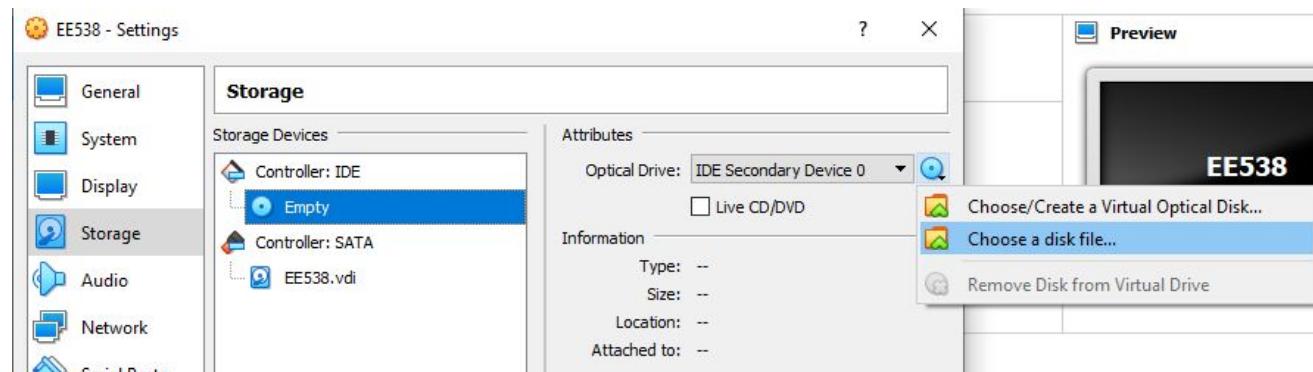
has at least 2 CPU cores allocated.



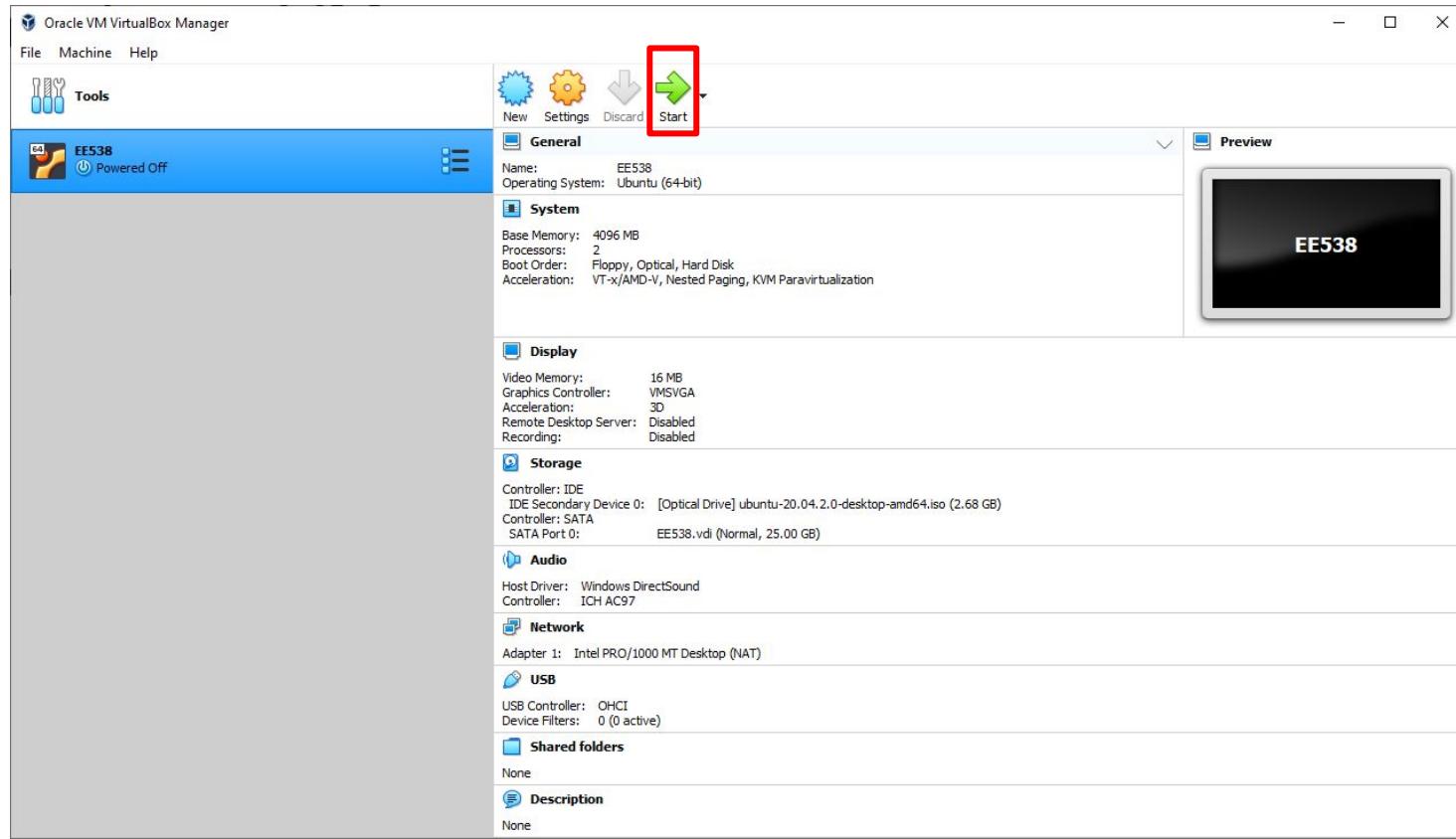
Install Ubuntu on VM

Step 8

Choose Ubuntu disk file
that just downloaded



Install Ubuntu on VM



Aug 25 09:23

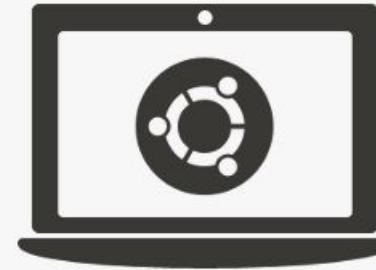


Install



Welcome

- Asturianu
- Bahasa Indonesia
- Bosanski
- Català
- Čeština
- Cymraeg
- Dansk
- Deutsch
- Eesti
- English
- Español
- Esperanto
- Euskara
- Français
- Gaeilge
- Galego
- Hrvatski
- Íslenska

[Try Ubuntu](#)[Install Ubuntu](#)

You can try Ubuntu without making any changes to your computer, directly from this CD.

Or if you're ready, you can install Ubuntu alongside (or instead of) your current operating system. This shouldn't take too long.

You may wish to read the [release notes](#).

Install



Keyboard layout

Choose your keyboard layout:

English (Angola)

English (Nigeria)

English (South Africa)

English (UK)

English (US)

Esperanto

Estonian

Faroese

Filipino

English (US)

English (US) - Cherokee

English (US) - English (Colemak)

English (US) - English (Dvorak)

English (US) - English (Dvorak, alt. intl.)

English (US) - English (Dvorak, intl., with dead keys)

English (US) - English (Dvorak, left-handed)

English (US) - English (Dvorak, right-handed)

Detect Keyboard Layout

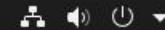
Quit

Back

Continue



Aug 25 09:24



Install



Updates and other software

What apps would you like to install to start with?

 Normal installation

Web browser, utilities, office software, games, and media players.

 Minimal installation

Web browser and basic utilities.

Other options

 Download updates while installing Ubuntu

This saves time after installation.

 Install third-party software for graphics and Wi-Fi hardware and additional media formats

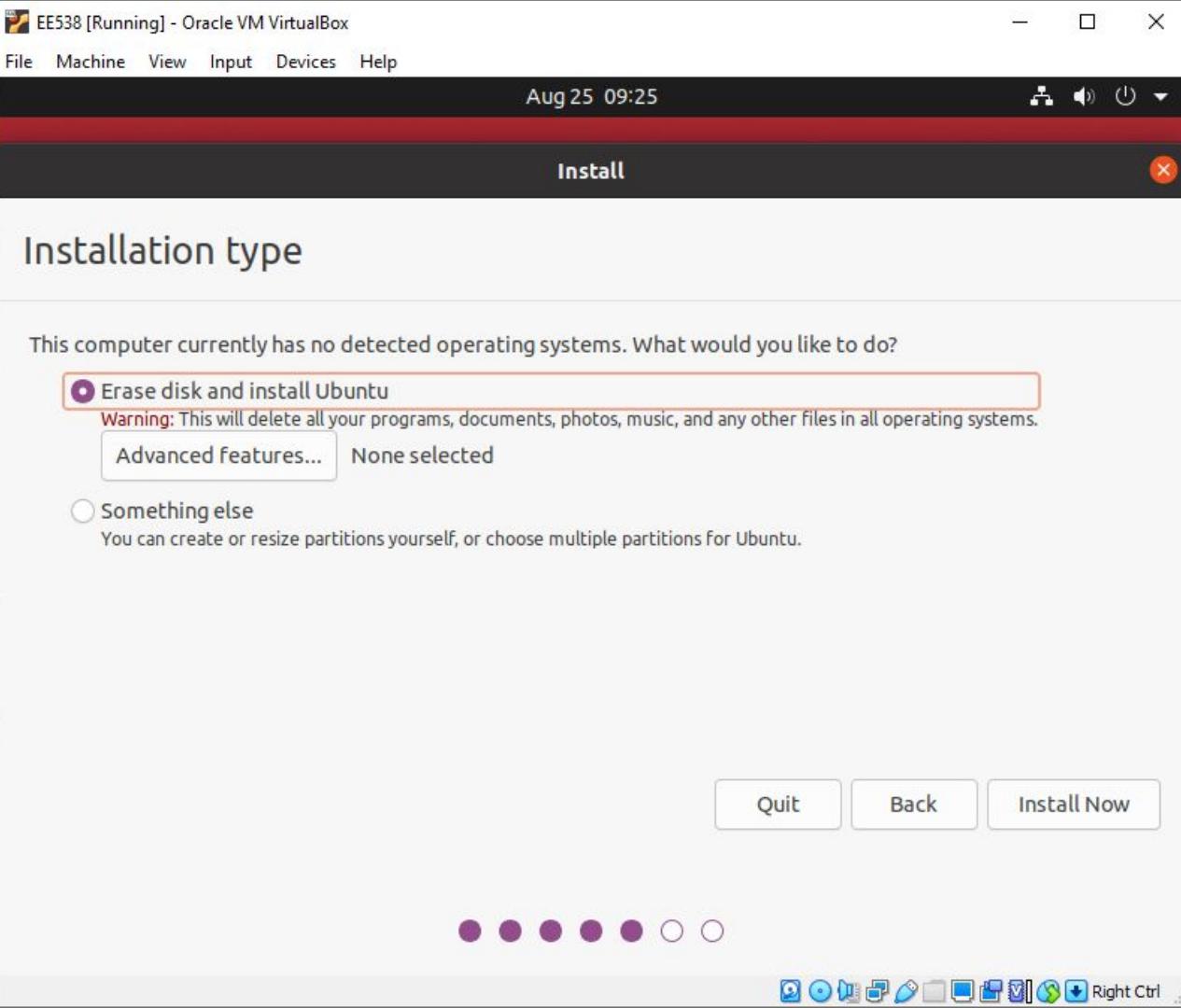
This software is subject to license terms included with its documentation. Some is proprietary.

Quit

Back

Continue





Aug 25 09:25



Install

Installation type

This computer currently has no detected operating systems. What would you like to do?

Write the changes to disks?

If you continue, the changes listed below will be written to the disks. Otherwise, you will be able to make further changes manually.

The partition tables of the following devices are changed:
SCSI3 (0,0,0) (sda)

The following partitions are going to be formatted:
partition #1 of SCSI3 (0,0,0) (sda) as
partition #5 of SCSI3 (0,0,0) (sda) as ext4

Go Back**Continue****Back****Install Now**

Aug 25 09:25



Install

Where are you?

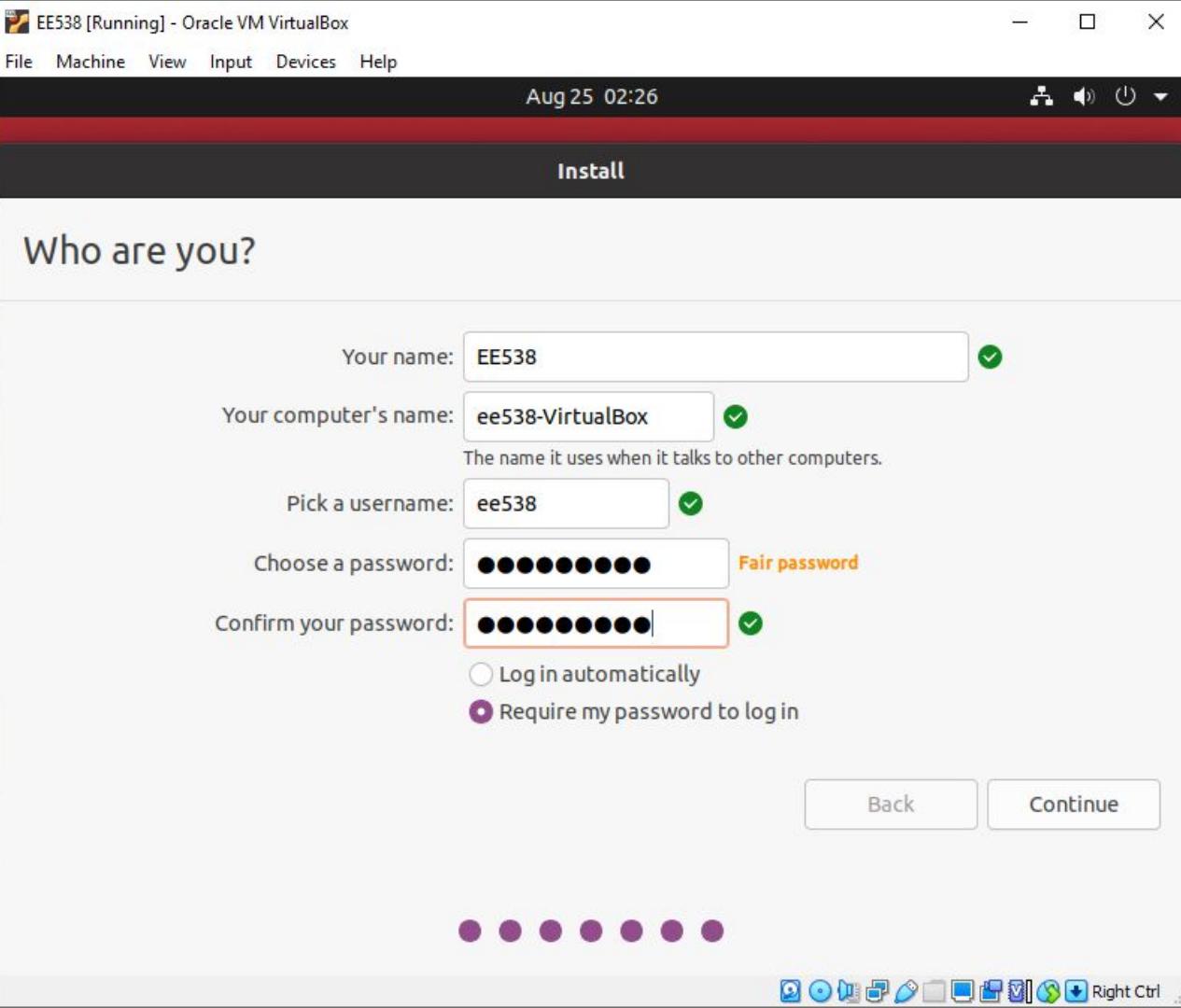


Los Angeles

Back

Continue





Install

Welcome to Ubuntu

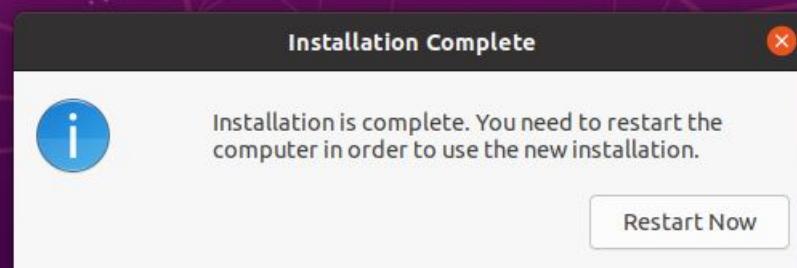
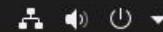
Fast and full of new features, the latest version of Ubuntu makes computing easier than ever. Here are just a few cool new things to look out for...



▶ Retrieving file 102 of 159

Skip

Aug 25 02:35



Aug 25 02:36



You are done!



EE538

Not listed?

ubuntu®

Programming Practice - Find Average

- Steps:
 - 1. Declare a function

double FindAverage(std::vector<int> &inputs) in src/lib/cplib.h

- 2. Define and implement FindAverage in src/lib/cplib.cc
 - 3. Use your FindAverage in src/main/main.cc
 - 4. Try ***bazel run src/main:main***
- Tests:
 - 5. Add a test case in tests/cplib_test.cc
 - 6. Try ***bazel test tests:cplib_test***

Programming Practice - Find Average

```
class CPPLib {  
public:  
    // A very first question: Find Average in an vector  
    double FindAverage(std::vector< int > &inputs);  
  
};
```

Programming Practice - Find Average

```
double CPPLib::FindAverage(std::vector< int> &inputs) {  
    if (inputs.size() == 0) {  
        return -1;  
    }  
    double result = 0;  
    for (auto n : inputs) {  
        result += n;  
    }  
    return result / (inputs.size());  
}
```

Programming Assignment - Find Average

```
TEST(CPPLibTest, FindAverage) {  
  
    CPPLib cpplib;  
  
    std::vector<int> inputs = {1, 2, 3, 4, 5};  
  
    EXPECT_EQ(cpplib.FindAverage(inputs), 3);  
  
}
```