Introduction

Don’t give up! Work on your projects since the late penalty is very small.

Your projects will be regraded at the end of the semester. If you need earlier, send an email, we will try to regrade earlier.

The course/assignments are hard, but it is a challenge you can deal with and learn with it.

Next couple weeks, the lectures are going to be very light. In the remaining time, we will have Q&A sessions.

We will talk about OS and code optimization.
Next week, Dr. Gary Leavens will talk about program analysis.
We will cover a bit about software security, and we will see an example of stack smashing.

Operating Systems

For software people, the lowest level is the hardware.
As compiler people, we look at the architecture level, the ASI, the interface we deal with.
Understanding the lowest level helps us to optimize our code and other things.

What is firmware?
It is a kind of a software that is put onto a chip on your hardware. You cannot easily change it.
BIOS / UEFI
BIOS is old fashioned way how your first program runs when you power up your pc.
Whole chain of processes gets chained with the help of BIOS to start up the operating system.

Kernel space is abstracted view of a hardware.
Today, in protected mode, the kernel has more access to your hardware. There are restrictions to user space.
You just say I want to run this program and kernel decides when and how your program runs.
In your processor, you are running one program is running at a time. Kernel is switching between processes very fast, which brings the illusion that you run multiple programs at the same time.

System software: compilers, linkers, loaders.. which help you to build and run your programs

Kernel abstracts away hardware differences.
Think of electric vs gas-powered car. If you sit in any of these cars, you will have the very same interface: wheel, pedals etc.
Kernels does the same: it allows you to handle with the same interface by abstracting away the hardware differences.
For example, you deal with USB flash and SATA using the same interface although they have differences in hardware and standards.

Kernel gives us the illusion that we have files. In does not really exist anywhere in the physical world. It is just the interpretation of byte values.

When you start a program, it does not immediately gets executed. Your operating system puts it into a queue and forces a fair scheduling for fair share of resources.

What is an operating system?

a) Kernel?
b) Kernel + system software?
c) Kernel + system software + GUI?
d) Kernel + system software + GUI + fortnite?
e) Something else?

Back then, Microsoft said that browsers are a part of operating system.

Command-line is not linux. None of the software you run with this command-line is not part of the linux operating system.

Richard Stallman
Look at some of the videos of him. See stallman.org
He wanted to make of his own free version of Unix
Open source software vs free software

Don’t be afraid to look at the source code even if you don’t understand everything
You will get farther than you would expect

The initiator of the linux kernel: torvals
Initially, 50k lines of code (1991)
GNU Public License (GPL)

Linux comes with lots of other software packaged with linux kernel.
Linux is free and open source

The word ‘boot’ comes from ‘bootstrapping’
BIOS loads and runs 512 bytes of HDD, which is a very limited space. We cannot fit, for example, linux in this space. Therefore, we put another bootloader which will run some other code.
If you have a computer with BIOS, you can put code in first 512 bytes of your HDD and run your own operating system

Booting is how we have our initial program running

`gcc -S -c` : compile to assembly code (-c). You can read the output of it thanks to -S.

We have different set of instructions for different hardware because of the hardware design
For example, ARM follows RISC approach.

If we don’t use -S option, we will have an object file instead of a readable set of lines in ASCII characters.
We can use **`objdump`** to read what is inside in this object file

The main components of your operating system are: file system, virtual memory and scheduler

Go through the link at the end of the slides.