What we will cover

- What is Software?
- System Software
- Application Software

What is Software?

- First, it is important to understand that modern computers are “general purpose machines.”
- In other words, they are flexible enough to be made to perform very different tasks, depending on what purpose the computer will serve.
What is Software?

- The term “software” refers to any “instructions” that are loaded into the computer to perform a given task (i.e. “programs”).
- It also refers to data that is produced as a result of these tasks (files, etc.).
- In short it does NOT refer to the unchangeable attributes of the physical machine.

Software vs. hardware

- You may correctly remember that hardware too is “hard-wired” to contain and interpret instructions.
  - ROM chip – bootstrap loader
  - Instruction set in the Microprocessor
  - Mathematical and logic operations in the ALU
- So keep in mind that “instructions” and “software” are not always the same thing.
- In this context, Software can generally be understood as any external instructions that are loaded into the machine.

How is Software delivered

- Since software is separate from the machine itself, it must be somehow delivered so we can load it into the machine.
- It can be delivered via various forms of storage media or may even be downloadable.
- Just keep in mind that it is always originally separate from the machine. I mention this because it is possible to buy computers with software “pre-loaded” – but it is not an integral part of the machine.
How is Software Created?

- You can probably surmise that like anything else a computer works with, software must also be in binary form.
- But, software must also be written by people, who do not “speak” binary.
- We will understand more of the software writing process (programming) later. But for now lets assume that software can be written in a form understandable by people.
- So how does it get into a form the computer can understand??

Compiling and Interpreting

- There are two primary ways we can get software into a binary form.
  - **Compiling**: An entire human-readable program is converted to machine-readable code by running it through a special piece of software called a “compiler”. Then it can be run on a computer in its entirety.
  - **Interpreting**: A human-readable program is interpreted line by line, as it is running, by another piece of software called an “interpreter.”

Compiled Software

- **Advantages**: Since it is converted to binary before it is run, compiled software runs faster.
- **Disadvantages**: Since different computers use different instruction sets, a compiled piece of software can only be run on the type of computer that the compiler was written for.
  - Because...the compiler must know the format of the instructions that are contained in the appropriate instruction set hardware.
- **Example**: .exe files on PC-type machines. These cannot be run on Macintosh, etc. The program would need to be re-written (in many cases) and compiled in a separate version using a Macintosh compiler.
Interpreted Software

- **Advantages:** Since it is interpreted line-by-line as it is running, this software can be run on many different types of machines – The caveat being that an appropriate interpreter must be available on the machine. The programmer only needs to write one version of the software.

- **Disadvantages:** Since each line must be converted on-the-fly to a machine-readable form, this software runs slower.

- **Examples:** Web pages/JavaScript, Java programs. Web browsers and Java interpreters are some of the most common interpreters used on many types of computers today.

Review – What Is Software?

- Know what software is, and how it is distinguishable from hardware.

- Understand the various ways that software is converted from human-readable form to machine-readable

Type of Software

- While there are many “types” of software, they can be broken down into two main classes. These are:
  - System Software
  - Application Software
System vs. Application Software

- **System Software** only exists in the context of a computer, and has no human-task equivalent.
  - Examples: Operating systems, utilities, device drivers, interpreters, compilers, etc.
- **Application Software** allows us to perform a real human task, which can also be accomplished without a computer.
  - Example: Accounting, producing written documents, organizing and filing information, business tasks (ordering/inventory/stocking/etc), art, photography, and many others.

System Software – overview

- **Device Drivers** allow a computer to “understand” and interact with a device hooked to it, like a printer, scanner, network card, sound card, etc.
- **Utilities** let us maintain the computer and improve its performance (virus scanners, disk defragmenters, etc.).
- **Operating Systems (OS)** allow the computer to function as an interactive general purpose machine. The OS is the most important type of system software, so let’s spend some more time on understanding what an OS does.

The Operating System

- The operating system performs many different tasks that involve the actual operation of the machine.
  - Provides a user interface (a way for people to interact with the machine)
  - Runs other software
  - Manages devices attached to the machine
  - Manages processor resources
  - Manages storage resources
  - Manages memory resources
The OS: User Interface

- The User Interface consists of various ways to allow the user to input information, run programs and receive output.
- Most machines now use a Graphical User Interface (GUI) allowing use to “click” on various “icons” to run programs, save files, etc.
- We take this for granted now but remember that this was not always the case. The User Interface is very complex and takes a lot of computer processing power to run.

The OS: Running Other Software

- As you probably have guessed, the OS is the first piece of software that the bootstrap loader (remember that?) will look for and run.
- By itself, of course, it is not of much use.
- Its real power is that it allows us to run other programs, and in modern operating systems we commonly run several different programs simultaneously.
- It is the operating system that makes this possible, because it is a “layer” between the software we are running, and the machine itself.

The OS: Managing devices

- The OS uses the appropriate device drivers to interact with other peripherals attached to the machine.
- So for instance if you are printing something, the OS must know if the printer is already in use and also how to properly communicate with the printer so the item prints properly.
- Every device (input, output, storage) is managed by the OS. This way, each individual piece of software we run does NOT need to know the specifics of the devices. This is another way the OS functions as a “layer” between the software we are running, and the machine itself.
The OS: Managing Processor resources

- As you know it is possible for several different programs to run at the same time.
- The microprocessor must keep track not only of the instructions for each program, but must also take care not to “mix them up” with instructions from other programs!
- The OS works with the processor to make sure programs can run independently without “stepping” on each other.

The OS: Managing Storage

- Remember our discussion about storage hardware.
- It is the OS that keeps track of the table of contents, decides what addresses to write/read the files, constantly evaluates the most efficient use of space while keeping fragmentation to a minimum, etc.
- It also decides how the storage media is “broken down” into different addressable “chunks.”
- I did not explore these complexities in our simple storage hardware example. Different systems use different methods to manage storage.

The OS: Managing Memory

- The OS manages RAM as well.
- It must know what areas of memory are being used by which data/programs/etc. and make sure that memory already being used is not written over by some other process.
- It also makes sure memory no longer is use is “released” so it may be used by other processes.
- It also manages the “Virtual memory” as discussed in our Hardware lecture.
System Software Review

- Know what System Software is
- Know and understand the various functions of the operating system.

Application Software – overview

- Unlike System Software, Application Software allows us to perform a human task that would also be possible to perform without a computer.
- In this way, application software provides a way to simulate/calculate/represent a real-life task.

Application Software Design

- Remember our previous discussion about digital representation.
- The analysis process we discussed then is crucial to the development of effective application software.
- Remember, not only does application software need to know what instructions to perform, it also needs to know how the data we are representing in our simulated human task is designed and structured.
Application Software and the OS

- Application Software is written to run not only on a specific machine, but more importantly via a specific OS "layer."
- Application Software therefore does not "talk" directly with the machine, but rather the OS serves as the "intermediary."
- This is why for example an old version of an Application might not run properly on a newer OS, even though the two versions of the OS were written for the same machine.

Types of Application Software

- There are as many "types" of application software as can be imagined, but they can generally be broken down into several main types (these are not "official" types and of course overlap in cases):
  - **Personal Productivity** - Software that makes common tasks like writing, organizing information, budgeting, etc. easier to do.
  - **Problem Solving** - various types calculators, measurement converters, equation solvers, statistical analysis, etc.
  - **Educational** - Tutorials, interactive learning, etc.
  - **Hobby** - personal interests like music, graphics, gaming, sports, etc.
  - **Business Productivity** - Very complex and sophisticated packages used for running an entire business.
- Think of some other types and examples.

Common Applications

- Today, there are integrated packages available that contain many different types of personal productivity applications. Four very common types, frequently packaged together, are:
  - **Word Processing / Document Production**
  - **Spreadsheets**
  - **Database Management Systems**
  - **Presentation software**
- Can you think of any specific products?
Document Production

- Document Production applications allow us to produce a wide variety of "written word" documents that may be printed, displayed, or distributed. Some examples include:
  - Text Editors
  - Word Processors
  - Desktop Publishing packages
  - Web Page designers (AKA “HTML Editors”)

Spreadsheets

- Spreadsheets allow us to create various types of systems that perform calculations on ranges of data.
- Data is arranged into columns and rows; each single element called a cell.
- Spreadsheets can be "programmed" using formulas to perform various calculations as needed.
- Can be used for
  - Ledger sheets (accounting)
  - Checkbook registers
  - Statistical analysis
  - Graphing changes in data over time

Databases

- Databases are used to classify, organize, and locate data that is stored in a structured format.
- Database systems are at the core of many common applications, for example:
  - Student Records systems
  - Library Systems
  - Inventory/Purchasing systems
- We will learn more about databases later in this course.
Presentation Software

- Presentation Software has largely replaced older audio-visual technologies (slide/overhead/opaque projectors, file projectors, tape players, etc.)
- Used to present information to groups of people, for example in the following contexts:
  - Education/Classroom
  - Training
  - Conferences
  - Speeches

Application Software Review

- Know what Application Software is, and how it is different from System Software
- Know and understand some examples of application software.