# COMPSCI 1XC3 - Computer Science Practice and Experience: Development Basics

## Topic 1 - Shell Basics

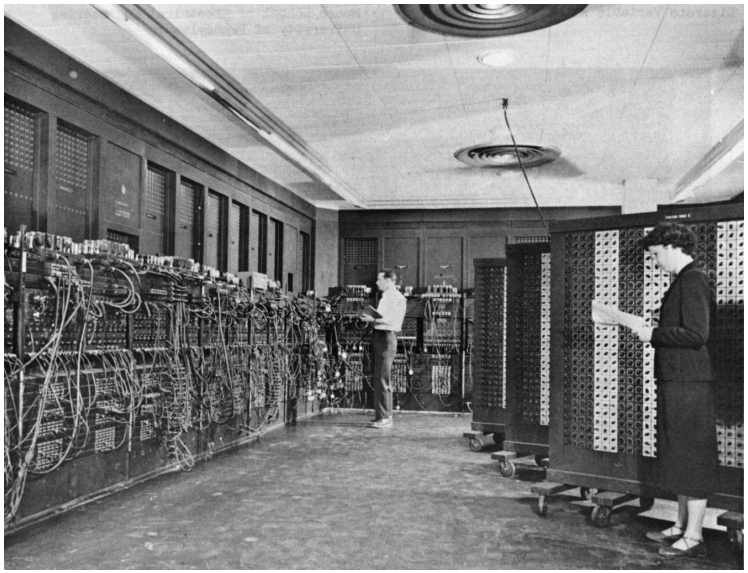
Zheng Zheng

McMaster University

Winter 2023

# The First Generation Computers

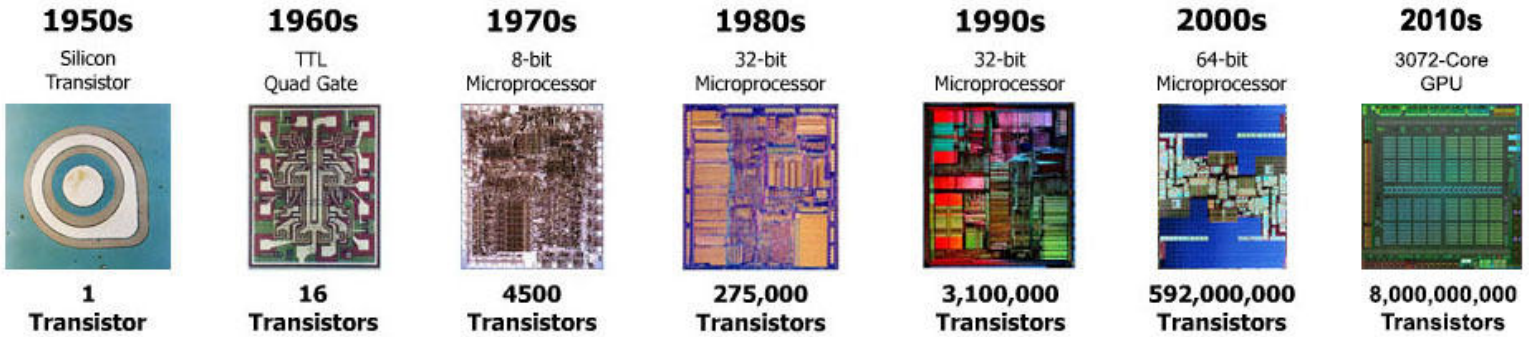
Computers (based on vacuum tubes) were very large, requiring large rooms for their housing. Programming via machine instructions, assembly language.



ENIAC (Electronic Numerical Integrator and Computer) was the first programmable, electronic, general-purpose digital computer, completed in *1945*.

# Semiconductor Electronics and Integrated Circuits (IC)

- The semiconductor transistor (late 40s) is possibly the most important invention of the $20^{th}$ century. It has smaller size, longer life and higher efficiency (100X).
- From late 1950s to 1960s, the development of IC contributed to the birth of the $3^{rd}$ generation computers.
- Moore's law is the observation that the number of transistors in a dense IC doubles about every two years.



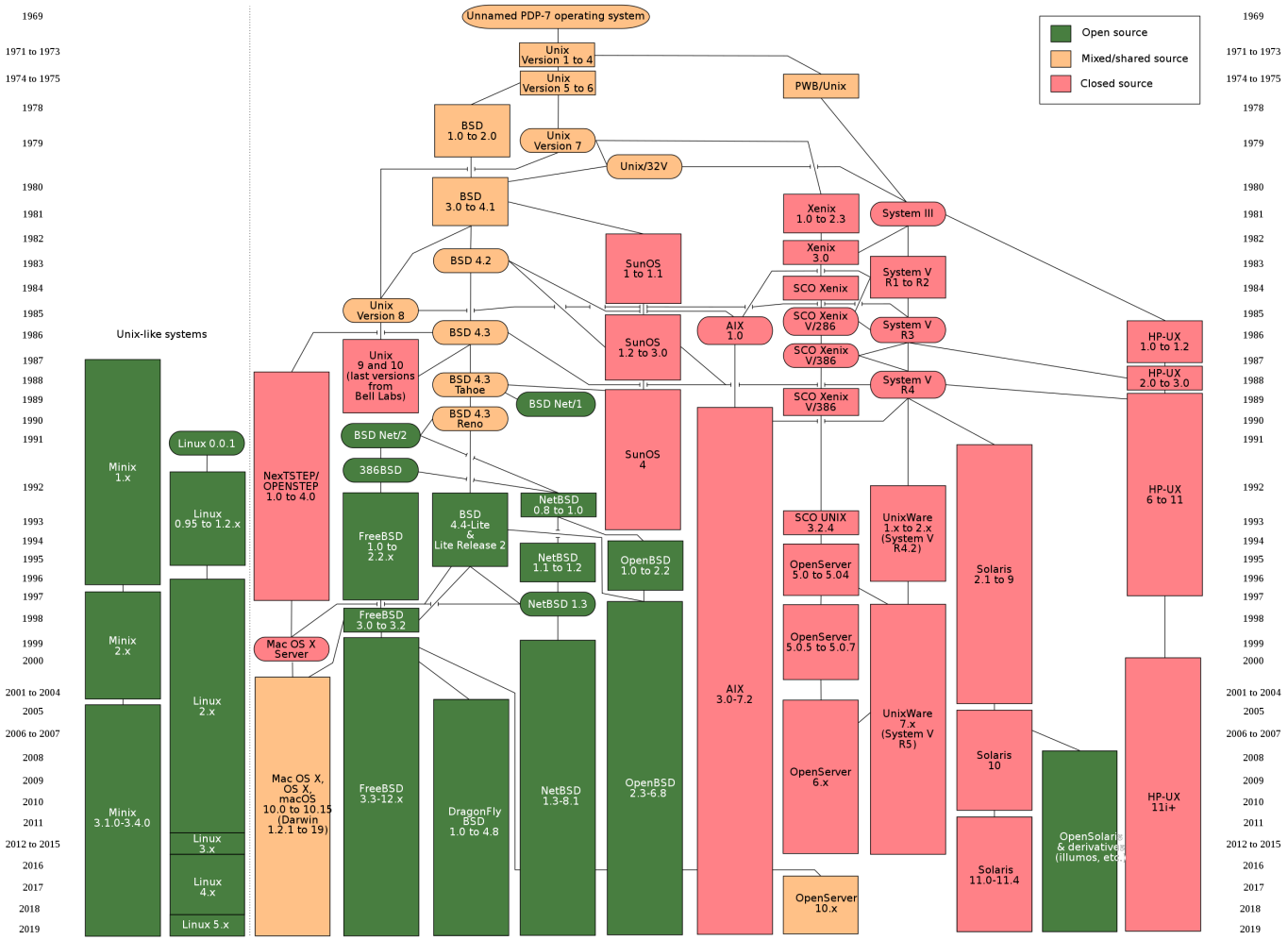
# Distinctive feature of third-generation computers

- Gradual maturity of the operating system (OS).
- Advanced Programming Languages.
  - Programs written at this time (notably using FORTRAN, COBOL and LISP) were not generally portable between machine architectures. `Incompatible due to a lack of standardization!`
  - All this changed with the development of C in 1972 by Dennis Ritchie at Bell laboratories.
    - Because C was strongly standardized, C programs could be ported across participating computer architectures with no compatibility issues.
    - C was originally developed for writing utilities for the Unix operating system.

# Unix - the Uniplexed Information and Computing Service

- Unix was originally written in assembly code, but after the development of C, the Bell Labs gang re-implemented the Unix kernel in C, and it has remained in C ever since.
- Due to it's low cost and high portability (especially to low-cost hardware), Unix was widely adopted by academic institutions, and from there, *the world!*
- Unix featured some key innovations:
  - An hierarchical file system with arbitrarily nested sub-directories
  - The universalization of almost all file formats as new-line delimited plain text.
  - A pervasive philosophy of modularity and code re-use, and the establishment of a set of cultural norms for software development practice.

# The Unix Family

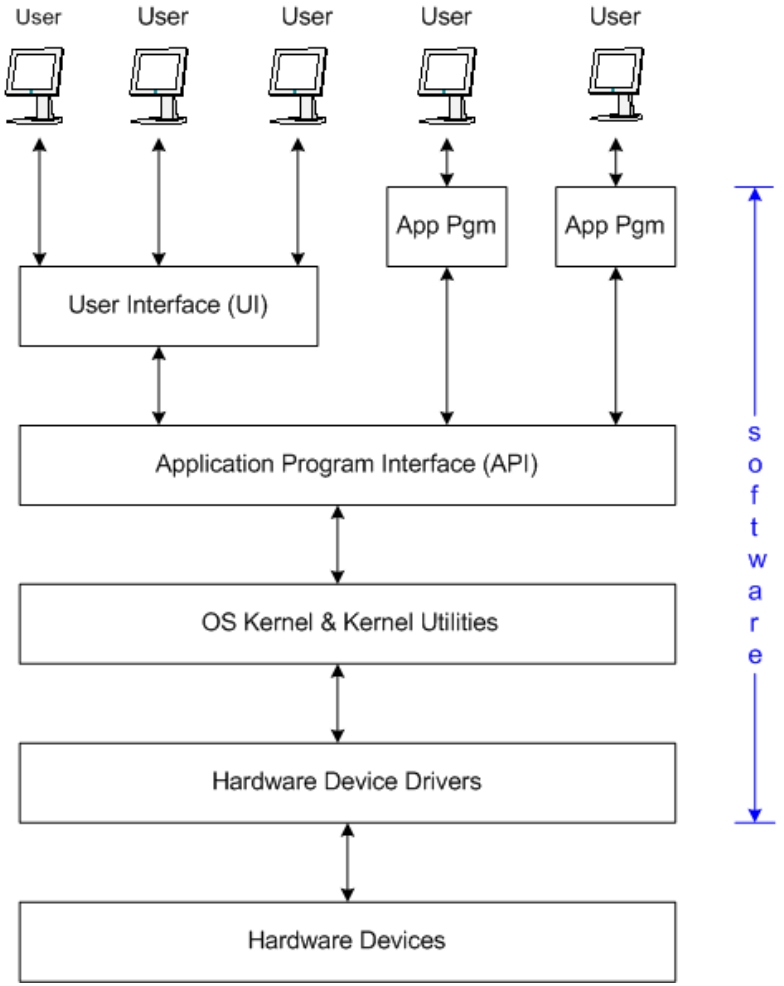


# Operating Systems

# Operating Systems In General

An **Operating System** provides a collection of services to **User Applications**, allowing them to run on a computer system's **hardware**.

- User Applications are anything from internet browsers to word processors to solitaire.
- The **API** provides system libraries and other utilities via *system calls*.
  - These are typically executed by the **Kernel**.

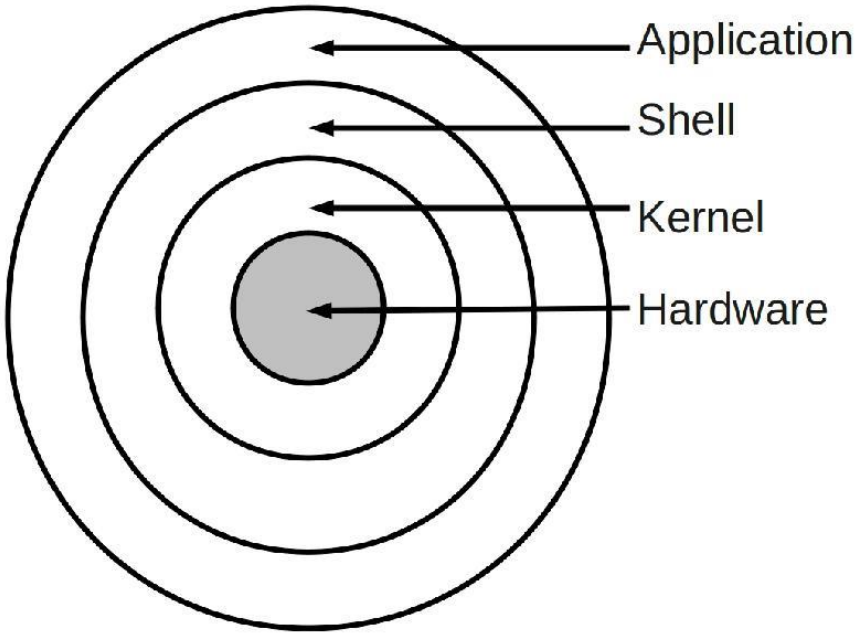


# The Kernel

Operating systems include a **kernel**, which manages:

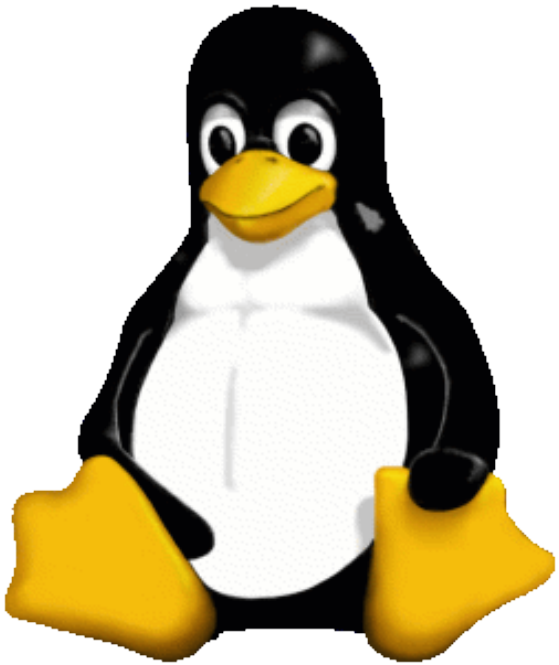
- Access to Memory (Random Access and Read Only)
- Access to the CPU
- Input / Output handling
- Access to hardware and software resources

Most operating systems use kernels written in C, because C is *fast*.



# Linux

Linux is a family of free operating systems.



- Unix was free until 1984 when AT&T divested itself of Bell Labs. Unix then became proprietary software.
- This led to the creation of The GNU Project, and the GNU General Public License in 1989, which kicked off the *open source* movement.
- The Linux Kernel was written in 1991 by Linus Torvalds at the University of Helsinki.

Today, Linux has the largest install base of any operating system, though only about 2% of personal computers run it.

# Giving it a Bash!

In Linux distributions, command line interfaces are commonly used

- Command lines have a high skill cap than Graphical User Interface (GUI).

The **Bash** shell is a very common command line interface in Unix-like environments.

- In Windows:
  - The Windows Subsystem for Linux allows Windows 10 users to access a bash prompt.
- On Macintoshes:
  - Opening up a terminal and entering the command `bash`
- In Linux:
  - Do you have to ask?

This course will require you to have ready access to a bash prompt.

**Your homework this week is to get your computer set up so that you have access to a bash prompt.**

# Accessing Linux from Older Windows Computers

We have set up a server for you to login to if the options on the previous slide don't work.

- Remote servers are accessed using a **Secure Shell Protocol (SSH)**.
- On Windows, it is common to use a secure shell client, such as **PuTTY**.
  - `https://www.chiark.greenend.org.uk/~sgtatham/putty/`

The department has set up a server for the class to use this semester. For more information on how to access it, check the *Resources* section of the course content on Avenue.

- Always remember to `logout` when you're finished!

# Bash Commands

Almost all Unix and Unix-like systems support a comprehensive set of Bash commands.

- `https://en.wikipedia.org/wiki/List_of_Unix_commands`

Bash commands are extremely versatile.

- The output of one command can be made the input of another command using **Pipes and Filters**
- Bash commands can be collected into **Scripts** and executed as units.
- Bash commands can be invoked from programs written in C or Python.

All these topics will be covered in this course.

# Directory Structure

The directory structure in Linux is **hierarchical**.

- Directories may contain files and sub-directories, forming a **tree**.

In Bash, commands are executed within the **working** or **active directory**.

- One directory in your file system is designated as *active*. This active directory may be changed using the `cd` command.

```
1 %user@system:~/Documents/Example $ cd Topic1
2 %user@system:~/Documents/Example/Topic1 $ cd ..
3 %user@system:~/Documents/Example/ $ cd ..
4 %user@system:~/Documents/ $ cd Example/Topic2
5 %user@system:~/Documents/Example/Topic2 $ cd
```

# Actual Bash Commands

| Command | Description |
|---|---|
| `cat <filename>` | display the contents of the file |
| `cd <directory>` | change the working directory |
| `cp <filename> <filename>` | copy a file |
| `ls` | List directory contents |
| `man <command>` | show a command's **man page** |
| `mkdir <directory>` | make directory |
| `ps` | list all processes |
| `pwd` | outputs current working directory |
| `rm <filename>` | removes a file |
| `rmdir <directory>` | removes a directory (if empty) |

Important Linux Commands:
https://www.howtogeek.com/412055/37-important-linux-commands-you-should-know/

# Secure Shell Protocol

```
$ ssh username@serverURLaddress.com
```

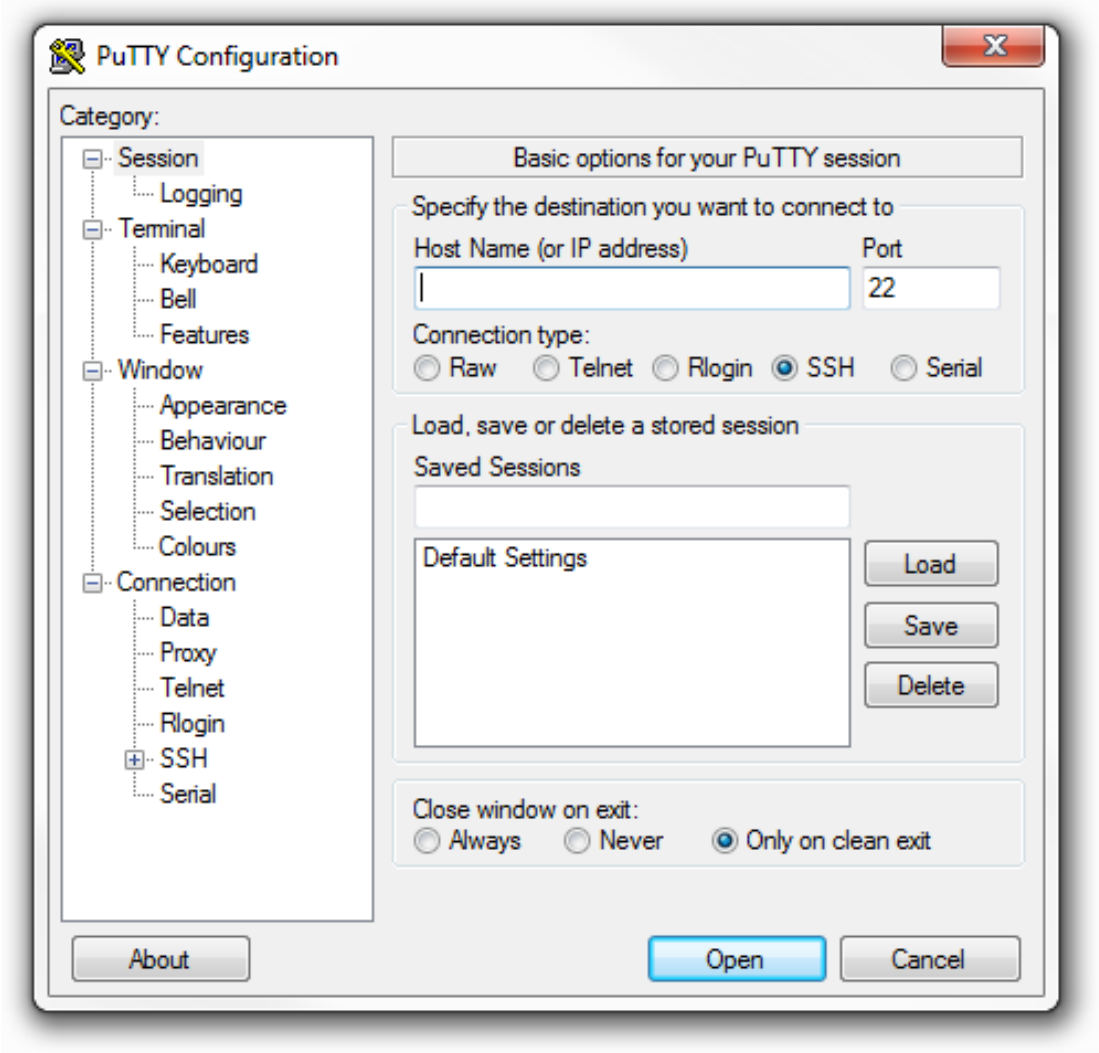
The Secure Shell (SSH) protocol is a network protocol for secure remote login over insecure networks.

- A **network protocol** is an agreed-upon format for information transmission.
- Anything but military-grade intranet should be considered insecure...
  - And even then...

In short, you (the **client**) open a shell on a remote **server**.

- This is the way that PuTTY accesses *pascal.cas.mcmaster.ca*, it just gives you a nice little GUI for entering the connection details.

# Graphical User Interface (GUI)




The point of this course is for you to gain computer skills.

The most important computer skill is knowing when and how to look things up.

When in doubt, consult the documentation!

# Network Protocols

Some common network protocols:

- Ethernet
- Internet Protocol (IP)
- Transmission Control Protocol (TCP)
- Hypertext Transfer Protocol (HTTP)
- Dynamic Host Configuration Protocol (DHCP)

Network protocols typically define the construction of **data packets**, which are transferred by the network.

- In general, data packets consist of a **header**, followed by some data.
- The header may contain different information depending on the protocol, such as the size of the packet, the source and destination of that packet, and security features like check-sums.

# Header Organization for IPv6 Data Packet

| Bits | 0-3 | 4-7 | 8-11 | 12-15 | 16-18 | 19-23 | 24-27 | 28-31 |
|---|---|---|---|---|---|---|---|---|
| 0 | Version | Traffic Class | | Flux Identification | | | | |
| 32 | Data Length | | | | Next Header | | Hop-Limit | |
| 64 | Source Address | | | | | | | |
| 128 | | | | | | | | |
| 192 | Destination Address | | | | | | | |
| 256 | | | | | | | | |

**IPv6 Header**

In general, the exact construction of data packets isn't something you need to worry about unless you're a network specialist or an Electrical Engineer.

# Acknowledge

The contents of these slides were liberally borrowed (with permission) from slides from the Summer 2021 offering of 1XC3 (by Dr. Nicholas Moore).