CMPE 220

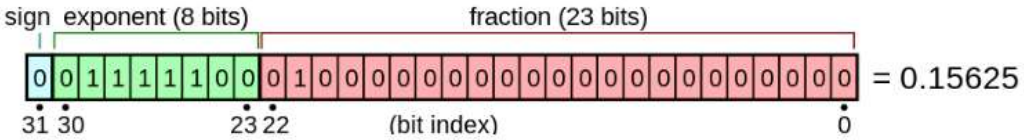
Class 28 – Review for Final

Development Tools

Early History

- 1940s: Machine code (binary)
- 1940s-1050s: Assembly Language
- 1950s: First high level language compilers (FORTRAN & COBOL)

- Early computers: Binary Coded Decimal (BCD) Arithmetic
  - Advantages – why is is still supported
- 1960s: Binary integer arithmetic
  - IBM/360 – 1965
- Floating Point Arithmetic – IEEE standard



San José State University

Character Sets

- 7-bit ASCII (American Standard Code for Information Interchange)
- 8-bit Extended ASCII (multiple variants)
- EBCDIC (Extended Binary Coded Decimal Interchange Code)
- Unicode – 1987
  - Over 160,000 characters (glyphs)
  - UTF-8 (variable length)
  - UTF-16

Machine Architectures

- Addressing Modes
  - Direct
  - Offset
  - Indexed
  - Indirect
- **Complex Instruction Set Computer (CISC)**
- **Reduced Instruction Set Computer (RISC)**
- **Microprogrammed Computers**
- **Instruction Pipelining**

RISC versus CISC

Reduced Instruction Set Computer

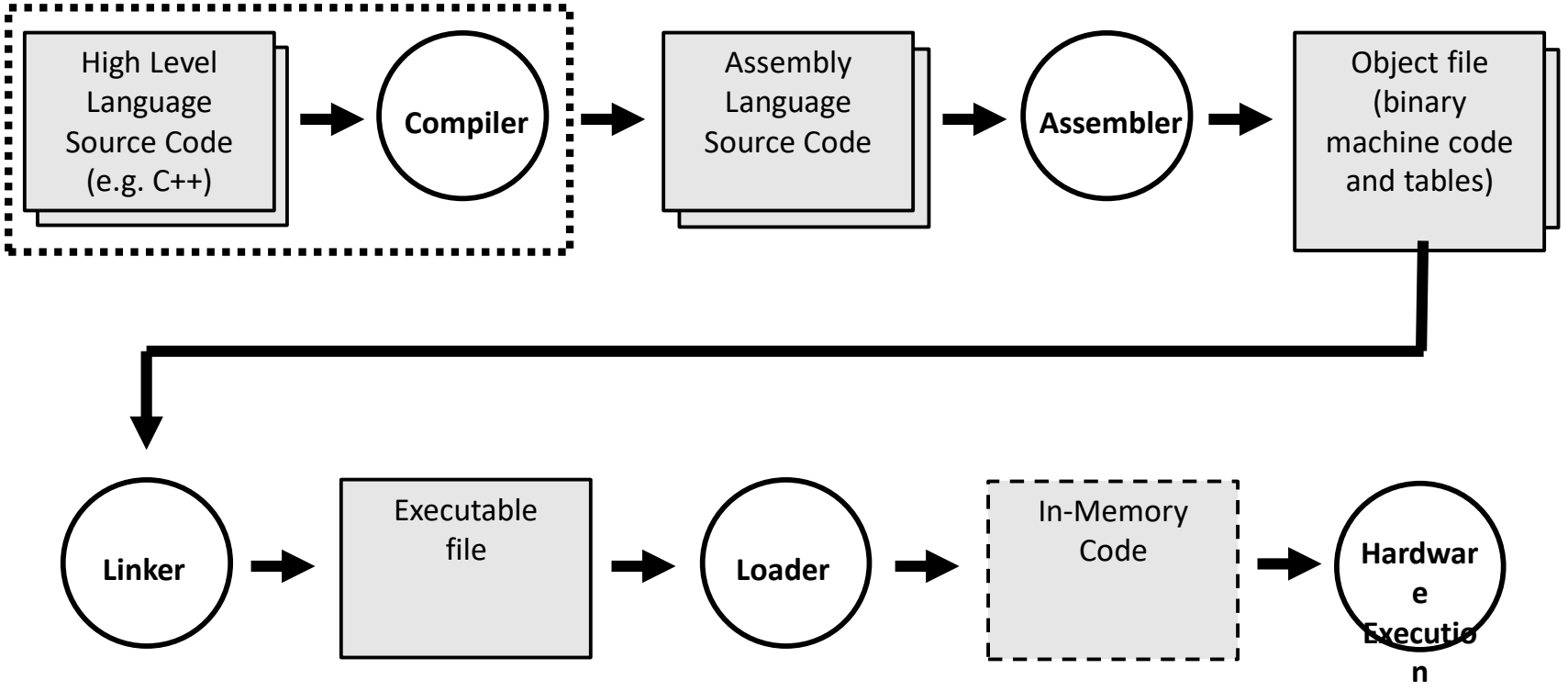
- One clock-cycle per instruction
- Effective *pipelining*
- Fewer addressing modes
- Requires more instructions per program (more RAM)
- Lower gate count
- Lower energy use

Complex Instruction Set Computer

- Multiple / variable clock-cycles per instruction
- More addressing modes
- Requires fewer instructions per program (less RAM)
- Higher gate count – more chip real estate
- Higher energy use

Development Tools

Building Software



Development Tools

- Assemblers
- Macro-Processors
- Compilers
  - Type of optimization
  - Optimization techniques
- Interpreters
- Linkers
  - Dynamic Linking – advantages
- Loaders
  - Absolute loaders
  - Relocating Loaders

San José State University

Development Tools

- Shell Scripting
- Make
- Integrated Development Environments
- Smart editors
- Debuggers
  - Breakpoints
- Version control
  - Check in / check out
  - Form & Merge

Development Tools

- Frameworks
  - Advantages
  - Model View Controller (MVC) architecture
  - Model-View-ViewModel (MVVM) architecture
  - Inversion of Control & Callbacks
  - Event driven programming

Assemblers (nomenclature)

- **Assembler:** converts assembly language to binary machine code
  - Two-Pass assembler
- **Macro Assembler:** allows the programmer to define new instructions that the assembler "expands" into the actual instruction set
  This does not create new machine instructions
- **High Level Assembler:** an assembler that includes certain high-level statements – such as IF/THEN/ELSE statements or loops – that don't correspond direction to machine instructions
- **Cross Assembler:** an assembler that runs on one machine, but generates binary machine code for a different machine
- **Micro Assembler:** converts microassembly source code into *microcode*… the low level code that implements the machine instruction set

Compiler Architecture

- Scanner (aka tokenizer, lexical analyzer)
  - FSM implementation
- Parser (aka syntax analyzer)
  - Formal Grammars (BNF)
  - Parse Trees
- Code Generator (aka semantic processor)

Software Development Life Cycle (SDLC)

- A software development life cycle (SDLC) model is a conceptual framework describing all activities in a software development project from planning to maintenance.
- Major Models
  - None
  - Waterfall (traditional)
  - Spiral
  - Agile / Scrum
  - DevOps

Types of Software Development

	Applications	Web Applications	Software Development Tools	Servers (FTP, Web, Database, Email)	Operating Systems	Embedded Systems
Knowledge of application	YES	YES				YES
Knowledge of hardware			YES		YES	YES
Knowledge of standards		MAYBE	YES	YES	MAYBE	
Performance Sensitive		YES		YES	YES	YES
Memory Sensitive					YES	YES
Process Management		MAYBE		YES	YES	

Software Development Processes

	Applications	Web Applications	Software Development Tools	Servers (FTP, Web, DB, Email)	Operating Systems	Embedded Systems
Formal Specifications	VARIES	VARIES	YES	YES	YES	VARIES
Languages	ANY	PHP, JavaScript, Ruby, Python	ANY	C, C++, JAVA	C, C++, JAVA, Assembly	Assembly, C, C++, Python, Ada, Rust, Go
Release cycles	RAPID	VERY RAPID	SLOW	MODERATE	SLOW	SLOW
Testing (Risk)	VARIES	VARIES	EXTENSIVE	EXTENSIVE	EXTENSIVE	VARIES

Software Development Processes: Fit

- The best "fit" for developing a particular type of software is a matter of opinion, and based on specific circumstances. The table below is intended as a basis for discussion.

	Applications	Web Applications	Software Development Tools	Servers (FTP, Web, DB, Email)	Operating Systems	Embedded Systems
Waterfall	VARIES	NO	YES	YES	YES	YES
Spiral	VARIES	NO	YES	YES	YES	YES
Agile	VARIES	YES	NO	YES	NO	NO
DevOps	VARIES	YES	NO	YES	YES	NO

Operating Systems

What Does a Modern Operating System Do?

1. Process Management
   - Fork & Exec
   - Scheduling and Dispatching
   - Interprocess Communications
2. Input / Output (I/O) Management
   - I/O blocks
   - Interrupts
3. Memory Management
   - MMUs & Address mapping
   - Virtual Memory & Swapping
4. File System Management
   - Disk fragmentation
5. System Functions and Kernel Mode
6. User Interaction – (maybe)
7. Networking

Types of Operating Systems

- Batch / Single-Process (early computers – 1940s-50s)
- Multi-Programming Systems (“modern” computers – last 1960s)
- Multi-Processor Systems
- Distributed Operating Systems
- Network Operating Systems

Types of Operating Systems

- Embedded Systems
  - Limited Memory
  - No Disks
- Real-Time Systems
  - Predictability more important than speed
- Cloud Systems
  - Network OS meets Virtualization
- Mobile Systems
  - Android versus iOS
  - Security issues
  - No Disks (but mass storage)
  - Novel I/O

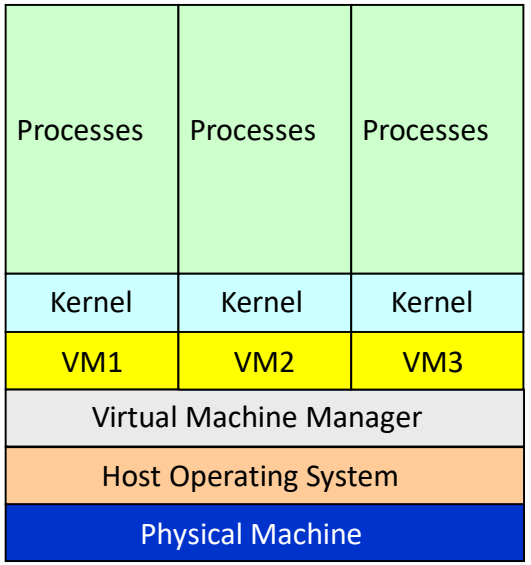
San José State University

Unix / Linux / POSIX

- Importance
- Evolution
- Standards

Virtual Machines

- Abstract the hardware of a single computer system.
- Create several different execution environments.
- Each environment can have its own operating system.
- Each environment believes it has the entire physical system.



Security

- Breach of confidentiality
  - Unauthorized reading of data
- Breach of integrity
  - Unauthorized modification of data
- Breach of availability
  - Unauthorized destruction of data
- Theft of service
  - Unauthorized use of resources
- Denial of service (DOS)
  - Prevention of legitimate use

Attack Modalities

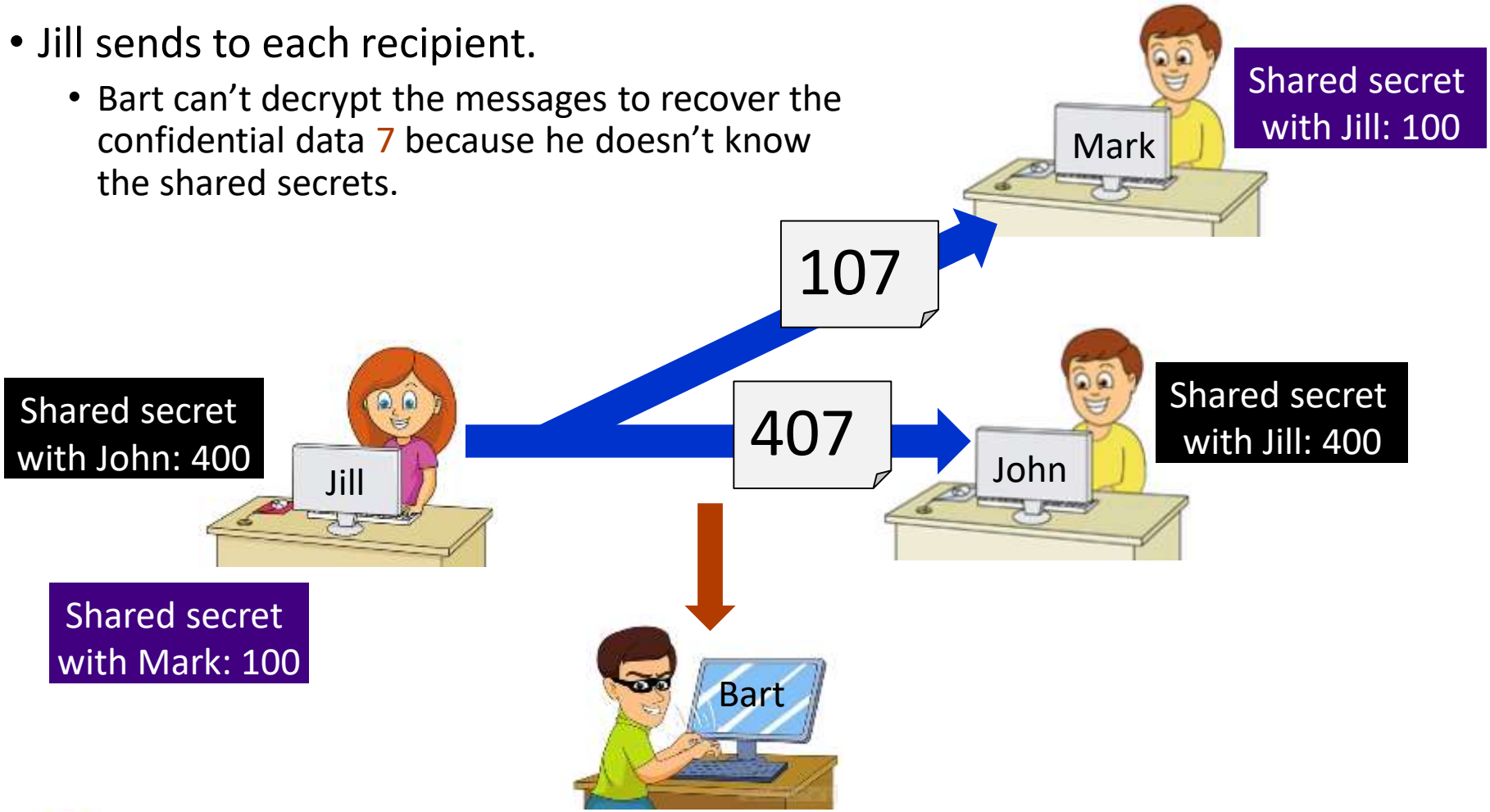
- Masquerading
- Replay Attack
- Session Hijacking
- Man-in-the-Middle
- Trojan Horse
- Virus
- Trapdoors
- Logic Bombs
- Code Insertion
- People Threats!

Mitigation of Security Risks

- Use secure protocols
- Require strong passwords / long passwords
- Require two-factor authentication
- Restrict admin access to the server
- Restrict physical access to the server
- Set file permissions to the least access (POLA)
- Keep all system software up to date
- Block unnecessary ports
- Run regular file checksums
- Run anti-virus software
- Make regular backups

One-Way Encryption & Public Key Cryptography

- Jill sends to each recipient.
  - Bart can't decrypt the messages to recover the confidential data 7 because he doesn't know the shared secrets.



Optimization

- Architecture & Design
  - Algorithms & Data Structures
- Compiler Optimization
  - Speed
  - Size
  - Power Consumption
- Reduce Kernel Mode Switches
- System Tuning
  - Monitoring Tools
- Hardware Configuration Tuning
- Housekeeping
  - File System Optimization
- Server / Task Allocation

System Utilities & Management

- System Management
- Network Management
- VPNs
- Maintenance
- Command Line Shells
- Windows Registry
- User Utilities

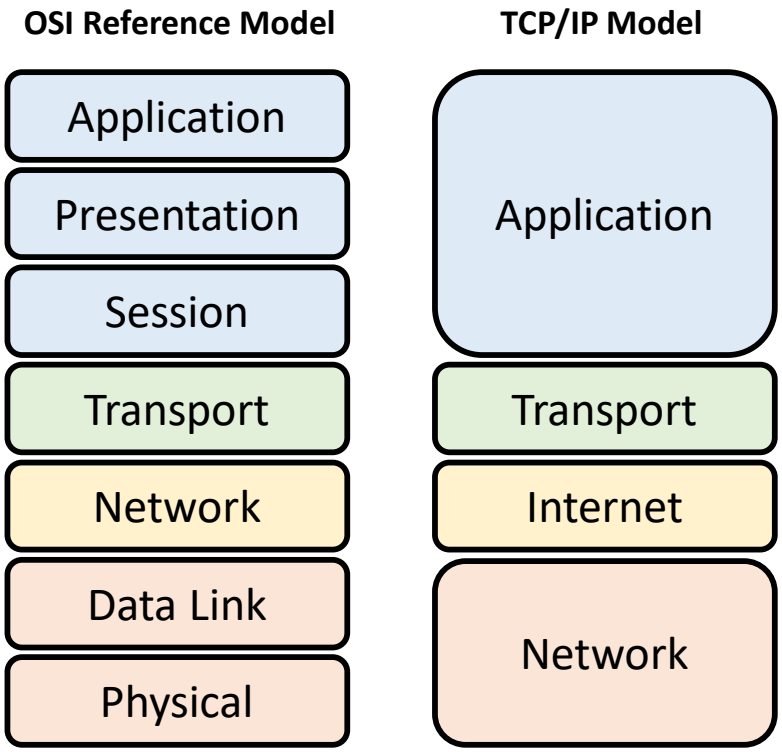
San José State University

Servers

The Client / Server Model

- A *server* is a program that runs on a computer, providing a specific service to other program(s), called *clients*
- In the POSIX world, a server is often called a *daemon* (demon)
  - a long-running background process that answers requests for services
- *The server program may launch multiple processes, as needed*
- The software that makes up a server may - or may not - have system dependencies
- Examples:
  - Database Management System
  - FTP Server
  - Mail Server
  - Web Server
  - Windowing System

OSI Versus TCP/IP Models

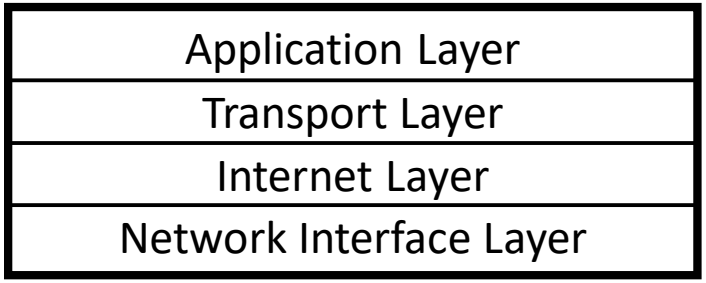


- OSI model provides a clear distinction between application, presentation, and session services.
- TCP/IP groups these as a single *Application* layer

- In the OSI model, the data link layer and physical are separate layers.
- TCP/IP groups these as a single *Network* layer

TCP/IP

- Transmission Control Protocol / Internet Protocol
- Based on research done by the Army Research Projects Agency (ARPA) in the 1960s and 1970s
- Standardized by the US Department of Defense in 1982
- Administered by the Internet Engineering Task Force (IETF) since 1989
- Four Layers:



TCP/IP Enabled the Internet

- Ubiquitous: supported by virtually every operating system
- Essentially makes every system today a *Network Operating System*
- Every device on the Internet has a unique 32-bit *IP address*, consisting of four 8-bit numbers (4.2 billion addresses):
  - 67.169.41.253
- IPv6: an extended protocol which supports 128-bit addresses, made up of eight 16-bit numbers, expressed in hexadecimal
  - FE80:CD00:0000:0CDE:1257:0000:211E:729C
  - Drafted by the Internet Engineering Task Force (IETF) in 1998
  - Finalized in 2017 – currently being deployed worldwide

Common Services

- Web Servers
  - UDP (Universal Datagram Protocol) – DNS Lookup
  - HTTP / HTTPS
  - Asynchronous JavaScript And XML (AJAX)
  - Simple Object Access Protocol (SOAP)
  - JavaScript Object Notation (JSON)
  - Representational State Transfer (REST)
- FTP Servers
- Mail Servers
  - POP, IMAP, SMTP
- Database Servers

Primary DNS Record Types

- **A Record:** The Address Mapping record (or DNS host record) stores a hostname and its corresponding IPv4 address
- **AAAA Record:** The IP Version 6 Address record also stores a hostname but points the domain to its corresponding IPv6 address
- **DNS CNAME Record:** The Canonical Name record can be used as a hostname alias that points to another domain or subdomain but not to an IP address
- **MX Record:** The Mail Exchanger record indicates an SMTP email server for the domain
- **TXT Record:** A text (TXT) record can store any type of descriptive information in text format. Often used for authentication

History & People

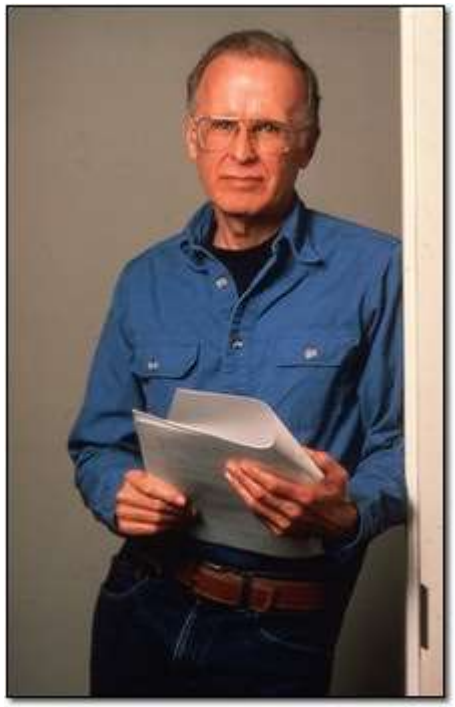
San José State University

Rear Admiral Grace Murray Hopper

- December 9, 1906 – January 1, 1992
- PhD, Mathematics, Yale University, 1934
- One of the first programmers of the **Harvard Mark I** computer, she was a pioneer in computer programming who invented one of the first *linkers*.
- She popularized the idea of machine-independent programming languages, which led to the development of **COBOL**, an early high-level programming language still in use today.

John Backus

- Pioneered parsing techniques; co-creator of Backus-Naur Form (BNF) for describing formal grammars
- The first FORTRAN (FORmula TRANslation) compiler was developed at IBM in 1957, by a team led by John Backus
- FORTRAN was the first *commercial* compiler – it was an extra cost add-on when you bought an IBM computer
- It is sometimes called the first modern compiler, because it employed formal grammar rules



Dennis Ritchie

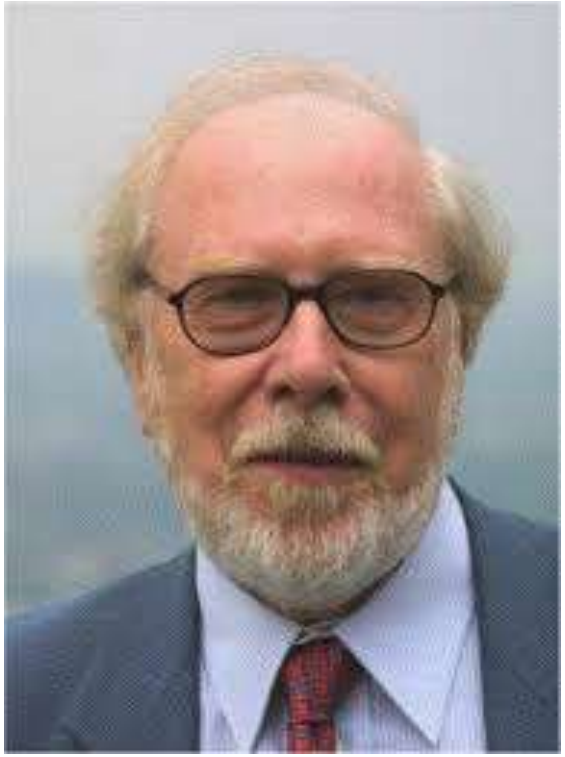
- While working at Bell Labs, invented the C programming language
- Co-invented the Unix Operating System (along with Ken Thompson)

Ken Thompson

- The first Unix shell, called *sh*, was developed by Ken Thompson at Bell Labs in 1971. Thompson, along with Dennis Ritchie, invented and popularized the **Unix** operating system. (Ritchie was also the inventor of the *C* programming language.)
- Sh was completely rewritten by Stephen Bourne in 1979, who created the Bourne Shell. A number of spinoff and replacement shells were subsequently developed, including ksh, csh, tcsh, and bash.

Nicklaus Wirth

- **P-code:** may also refer to *Pascal code* – an early portable layer
- PASCAL – 1970 – created by Swiss computer scientist Nicklaus Wirth as a highly structured instructional language; quickly replaced ALGOL as the teaching language of choice
- First P-code emitter - 1973
- UCSD P-Machine – 1977 – widely used in academia, as well as commercially



James Gosling

- **JVM (Java Virtual Machine):** a virtual machine built by James Gosling and Brendan Eich – 1995 – specifically for Java - but now widely used
  - Virtual machine instruction set
  - Manages memory and system resources

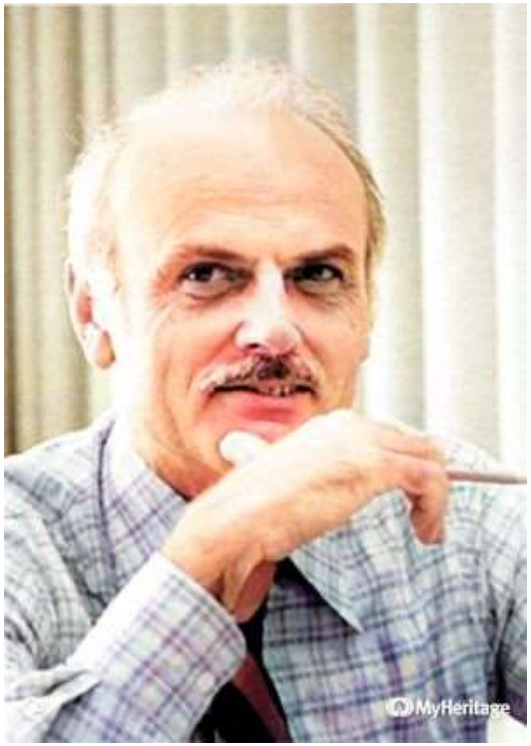
Abhay Bhushan

File Transfer Protocol (FTP)

- One of the earliest client/server applications (early 1970s)
- A graduate of the first class (1960–65) from the Indian Institute of Technology Kanpur
- Masters in EE from MIT
- Drafted RFC 114 – FTP
- Contributed to the development of the ARPAnet and email protocols

Ted Codd

- The term "relational database" was invented by Dr. Edgar (Ted) Codd at IBM in 1970.
- Codd introduced the term in his research paper "A Relational Model of Data for Large Shared Data Banks". In this paper and later papers, he defined what he meant by "relational".



David Patterson

The Rise of RISC

In 1980, *David Patterson* at UC Berkeley outlined an architecture for a *Reduced Instruction Set Computer*, and coined the term *RISC*.

- Fellow of the ACM
- Fellow of the IEEE
- Fellow of the Computer History Museum
- ACM Distinguished Service Award
- ACM-IEEE Eckert-Mauchly Award