Introduction

Course Overview

Course is about what lies under the command line prompts or the GUI Linux/Windows/MAC.

We will discuss:

- How are operating systems organized?
  - Process scheduling.
  - Process coordination.
  - Interprocess communication.
  - Memory management.
  - File systems and I/O.
  - Other topics" (time permitting).

Course Overview

Evaluation.

- 2 exams.
- Programming projects.
- Programming Assignments (labs).
- Quizzes.

There is a heavy programming component based on C language.

FAQ

- Why not use Java? Other universities do.
  - Java is NOT suitable for operating systems.
  - Operating systems are written in C and assembly.
- How proficient do I need to be in C?
  - Pointers, casting.
- How heavy is workload?
  - The assignments do not necessarily involve a lot of coding.
  - However, you will have to read and understand code.

What is an Operating System?

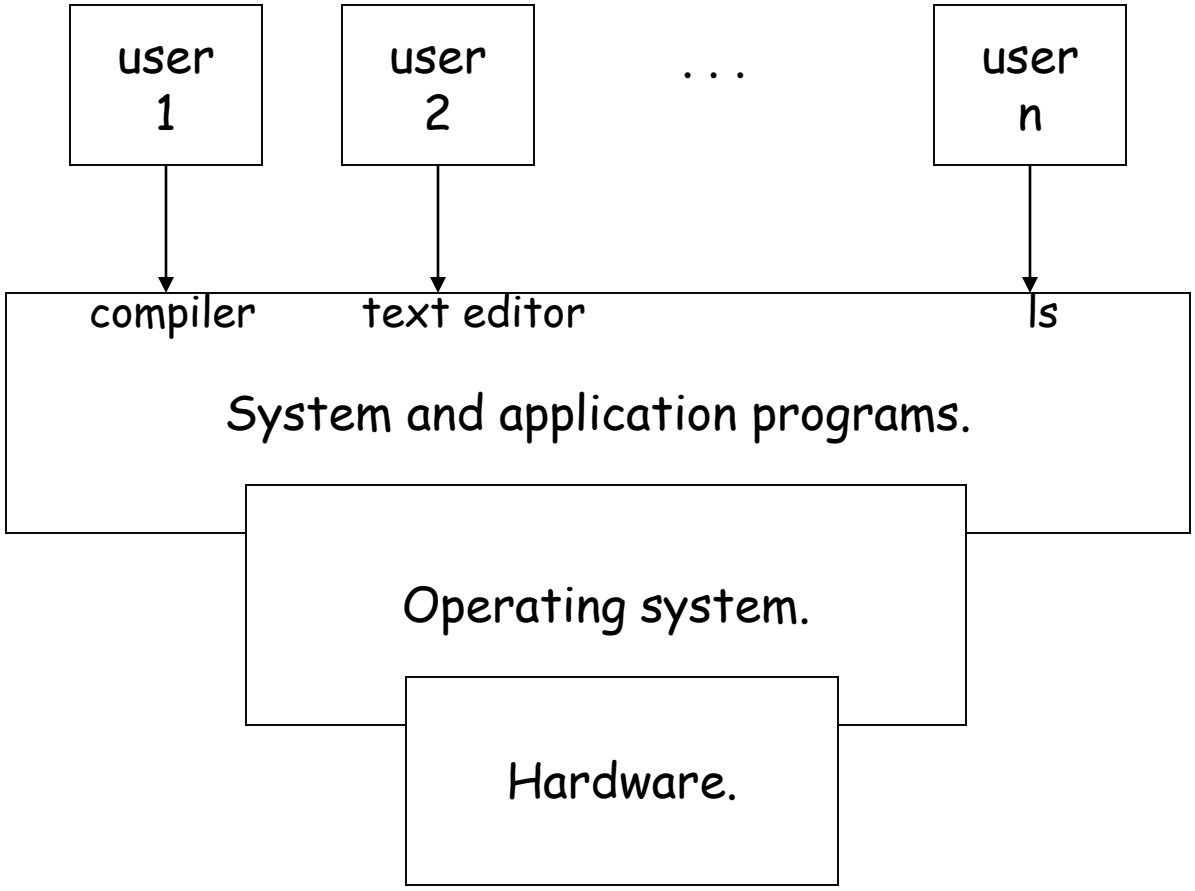
An operating system is an intermediary between a computer user and the hardware.

Make the hardware convenient to use.

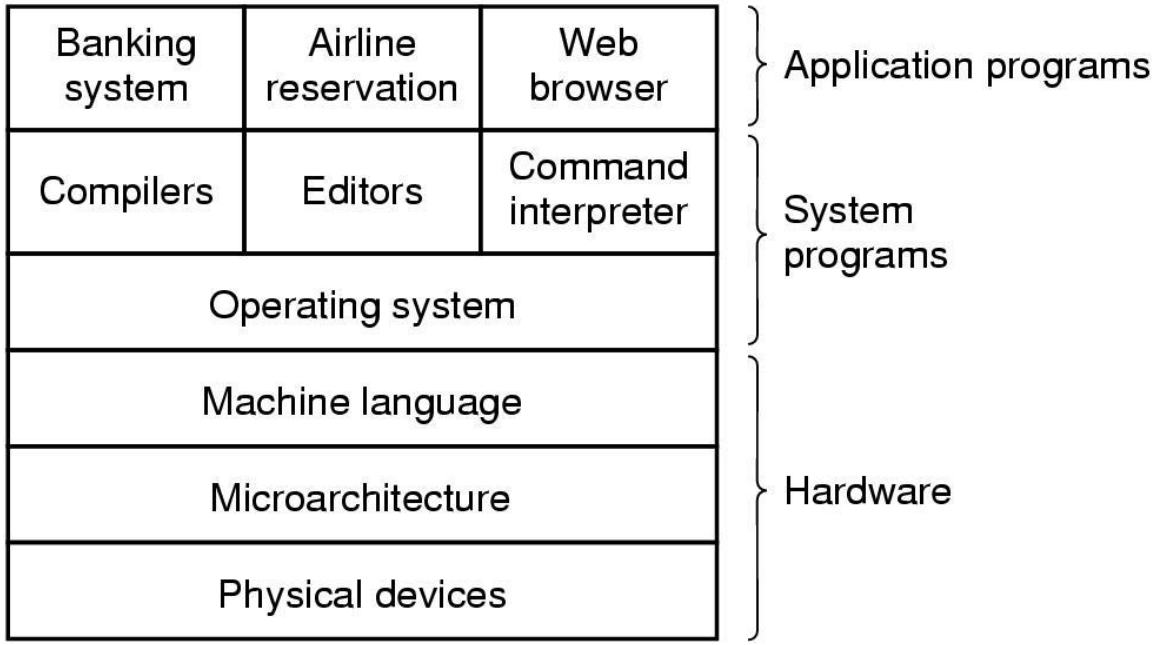
Manages system resources.

Use the hardware in an efficient manner.

System Diagram



What is an Operating System?



System programs:

- Generally, execute in user mode.
- Command interpreter (shell), compilers, editors, …

What is an Operating System?

The software layer between user applications and hardware.

Provides an abstraction for hardware by an interface to the hardware.

Hides the messy details which must be performed.

What is an Operating System?

Manages the hardware resources.

- Each program gets time with the resource.
- Each program gets space on the resource.

May have potentially conflicting goals:

- Use hardware efficiently (e.g., maximize throughput).
- Give maximum performance to each user (e.g., minimize response time).

Types of Systems

Batch

- Submit large number of jobs at one time.
- System decides what to run and when.

Time Sharing

- Multiple users connected to single machine.
- Few processors, many terminals.

Single User Interactive

- One user, one machine.
- Traditional personal computer.

Types of Systems

Parallel

- Traditional multiprocessor system.
- Higher throughput and better fault tolerance.

Distributed

- Networked computers.

Real Time

- Very strict response time requirements of hardware and/or software.

Single Tasking System

Only one program can perform at a time.

Simple to implement.

- Only one process attempting to use resources.

Few security risks.

Poor utilization of the CPU and other resources.

Example: MS-DOS.

Multitasking System

Very complex.

Serious security issues.

- How to protect one program from another sharing the same memory.

Much higher utilization of system resources.

Example: Ubuntu, Windows 10.

Hardware Basics

OS and hardware closely tied together.

Many useful hardware features have been invented to help the OS deliver its services.

Basic hardware resources:

- CPU
- Memory
- Disk
- I/O

CPU

CPU controls everything in the system.

- If work needs to be done, CPU gets involved.

Most precious resource.

- This is what you are paying for.
- User wants to get high utilization (useful work).

Only one process on a CPU at a time.

Hundreds of millions of instructions / sec.

- CPUs are getting faster all the time.

Memory

Limited in capacity.

- Never enough memory.

Temporary (volatile) storage.

Electronic storage.

- Fast, random access.

Any program to run on the CPU must be in memory.

Storage (SSD/Disk)

Virtually infinite capacity.

Permanent storage.

Orders of magnitude slower than memory.

- Mechanical device (Hardrive).
- Millions of CPU instructions can execute in the time it takes to access a single piece of data on disk.

All data is accessed in blocks.

- 4096 bytes, for instance.

I/O

- Disk/SSD are actually part of the I/O subsystem.
  - They are of special interest to the OS.
- Many other I/O devices.
  - Printers, monitor, keyboard, etc.
- Most I/O devices are painfully slow.
- Need to find ways to hide I/O latency.
  - Multiprogramming.

Protection and Security

OS **must** protect itself from users.

- Reserved memory only accessible by OS
- hardware is enforced.

OS **may** protect users from one another.

- Hardware enforced again.

Protection and Security

- Dual-Mode Operation.
  - User mode.
    - Limited set of hardware instructions and memory available.
    - Mode all user programs run in.
  - Supervisory mode.
    - All hardware instructions and memory are available.
    - Mode the OS runs in.
- Never let user run in supervisory mode.

Why Study Operating Systems?

I probably will not write one from scratch.

Haven't OS developers figured out everything out already? What more is there to do?

Why Study Operating Systems?

I have been writing Java (or C or Python) programs and I didn't need to know anything about how the OS works.

All I need to know are the commands I should submit to the OS to get my program to run or to store my data in a file.

Why Study Operating Systems?

Understanding how operating systems manage CPU, memory, and I/O can help you be a better programmer.

Example:

- Structuring a loop.
- Web servers.

OS and Loops

Consider the following code segments:

```
int data[128][128];
for (j = 0; j <128; j++)
  for (i = 0; i < 128; i++)
   data[i,j] = 0;
```

```
int data[128][128];
for (i = 0; i <128; i++)
  for (j = 0; j < 128; j++)
   data[i,j] = 0;
```

Does it matter which you use?

1000 d[0][0], d[0][1], d[0][2]
1003 d[1][0], d[1][1], d[1][2]
1006 d[2][0], d[2][1], d[2][2]

OS and Loops

Reasoning for answer NO? The code segments execute the same number of instructions.

The correct answer: Yes, it does matter.

- One will give you much faster results.
- The reason has to do with the way the OS manages its memory.

OS and Request Processing

Application: Withdraw money from a bank account.

Two requests for withdrawal from the same account comes to a bank from two different ATMS.

A thread for each request is created.

- A thread is a unit of execution.
- The same program (on the next page is executed).

OS and Request Processing

```
int withdraw(account,  amount)
{
   balance = get_balance(account);
   balance = balance – amount;
   put_balance(account, balance);
   return balance;
}
```

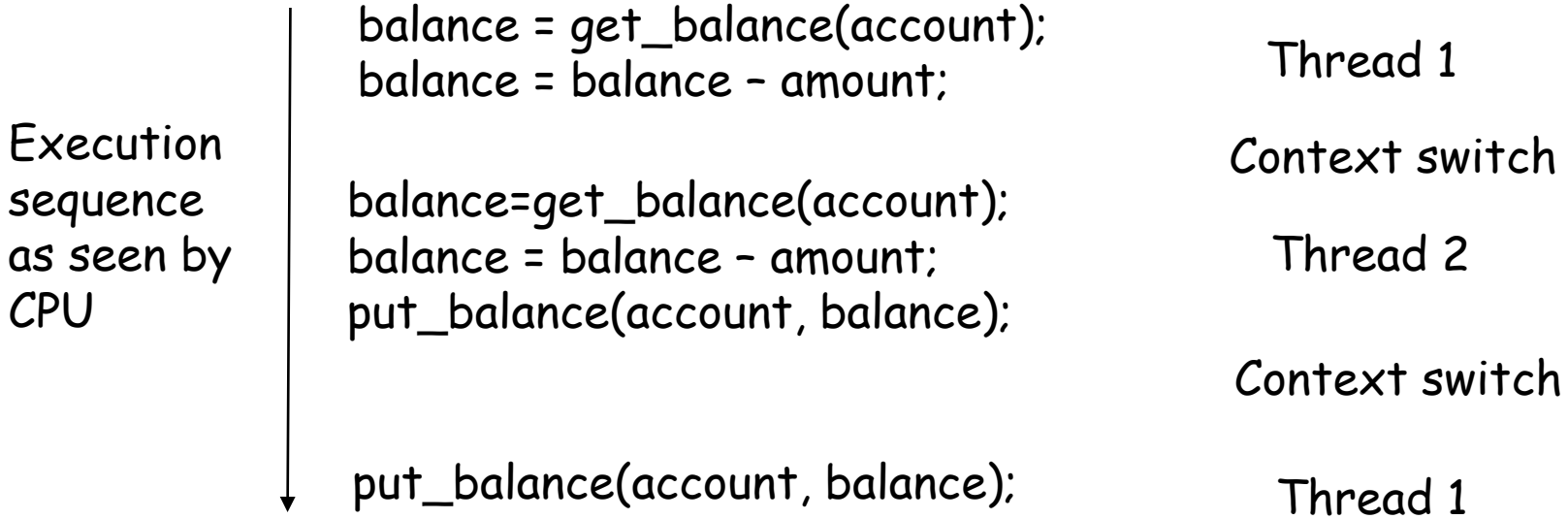
What happens if both requests request that $1000 be withdrawn from the same account?

Assume that the account currently has $1000.

OS and Request Processing

Both threads will read a balance of $1000.

Both threads will allow for $1000 to be withdrawn.



Why Study Operating Systems?

You may object to this example:

- Why are two requests taking turns using the CPU?
- A request comes in, should get executed before the next request.

If you did this, it would very slow.

Why? Consider an interactive program, for example. Well, you need to wait until you learn more about operating systems.

Why Study Operating Systems?

The example just presented is an example of concurrency.

Concurrency leads to interesting programming challenges that were first addressed in operating systems.

Why Study Operating Systems?

Also ...

The OS code is really large (Windows 10 is 50 million lines+, MAC OS is 85 million lines+).

Thus, the study of OS design is a study of the design of large software systems.

Understanding operating systems gives you un understanding about system security as well.

Why Study Operating Systems?

Ok – I'm still never going to write one from scratch.

Probably true.

But … Operating systems gives you insight into other areas of computer science.

- Data structures, concurrency, synchronization, resource management, distributed systems, networks, etc.

Why Study Operating Systems?

Aren't operating systems well-understood?

There isn't much left to think about.

Response: Absolutely NOT.

There are many challenges left e.g.,

- Parallel computing/programming with multicore.
- Mobile embedded devices.
- Real-time systems.

Challenge: Parallel Computing and Clusters

Many scientific applications want to use many CPUs at once.

- Need OS and language primitives for dividing program into parallel activities.
- Need OS primitives for fast communication.

Challenge: Mobile and Embedded Devices

Tiny computers everywhere – ubiquitous computing.

- Processor cost low enough to embed in many devices.
  - Tablets, cell phones, etc.

Very different hardware capabilities.

- Slow processors.
- Small amount of memory.
- Various wireless networks.

Why Study Operating Systems?

Ok, once we address these challenges then there isn't much.

All I can say is that people have been making statements like this since the 1950's.

They are wrong.

Hence, I'll end this section with a brief history.

Operating System Timeline

First generation: 1945 – 1955.

- Vacuum tubes.
- Plug boards.

Second generation: 1955 – 1965.

- Transistors.
- Batch systems.

Third generation: 1965 – 1980.

- Integrated circuits.
- Multiprogramming. Virtual Machine (VM/370): 1972.

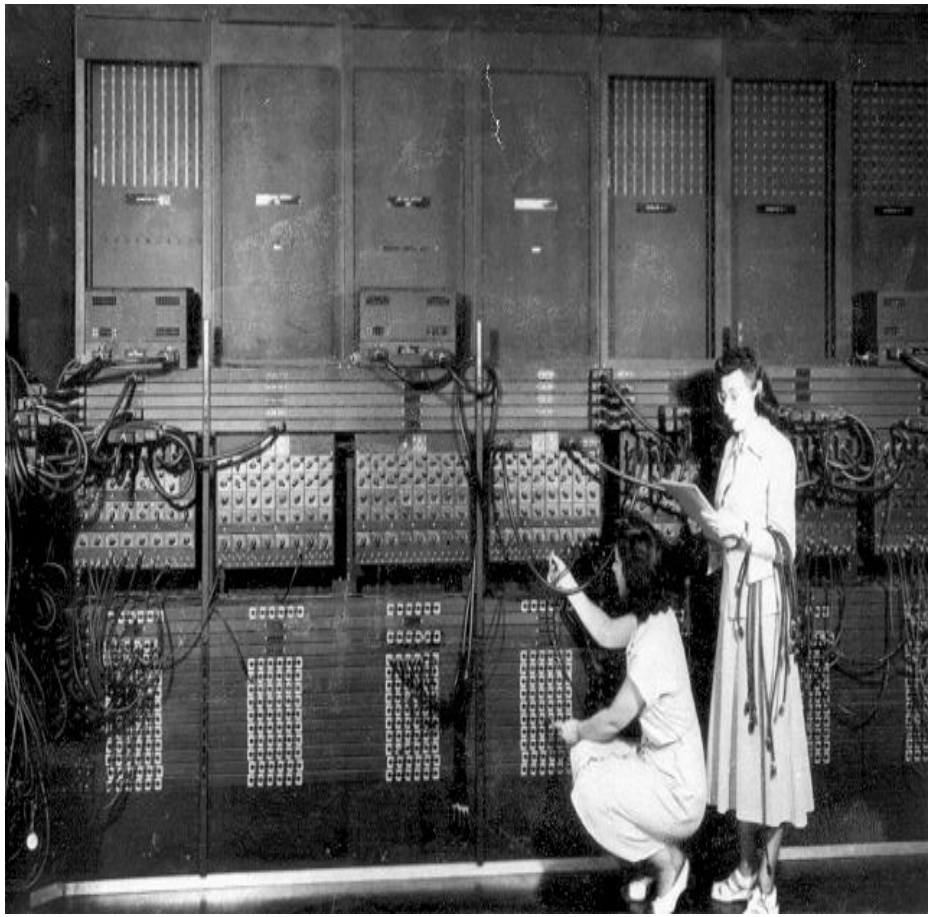
Fourth generation: 1980 – present.

- Large scale integration.
- Personal computers. Windows 10: 2015.

Next generation:

- Systems connected by high-speed networks.
- Wide area resource management.
- Internet of things.
- Quantum computing integration.

First Generation (1945-1955): Direct Input



Eniac, 1945.

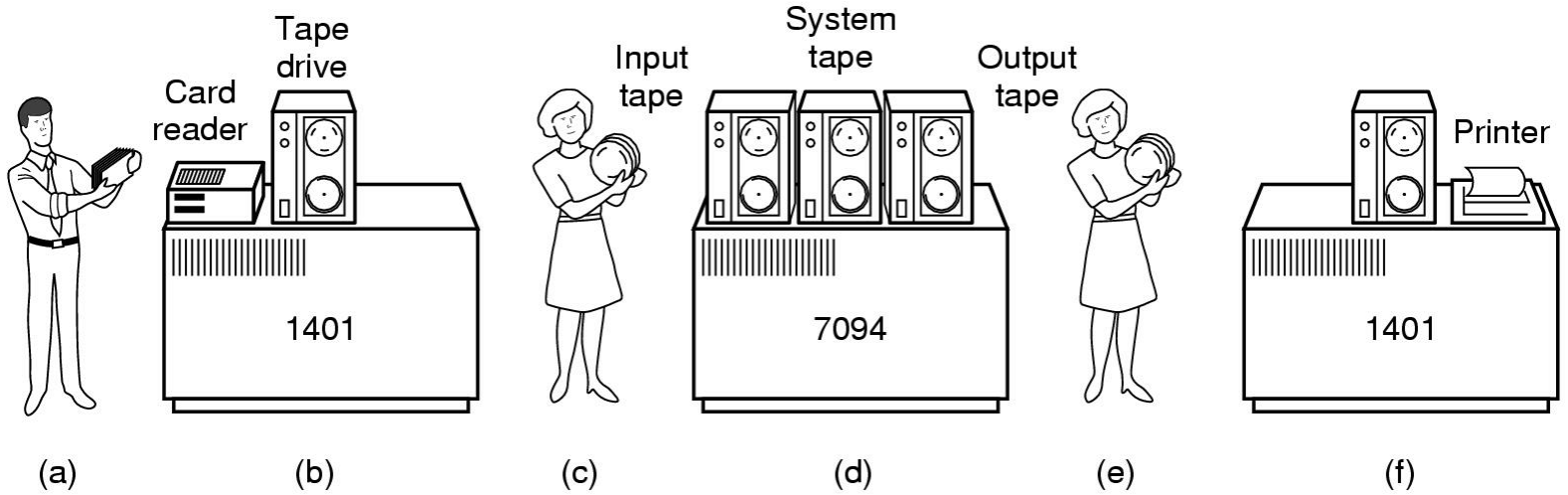
- Run one job at a time.
  - Enter it into the computer (might require rewiring).
  - Run it.
  - Record the results.
- Programming languages were unheard of.
- Assembly languages were not known.
- No reason for an OS.

A Famous Quote

"I think there is a world market for maybe five computers."

Thomas Watson, Chairman of IBM – 1943.

Second generation: batch systems



Bring cards to 1401.

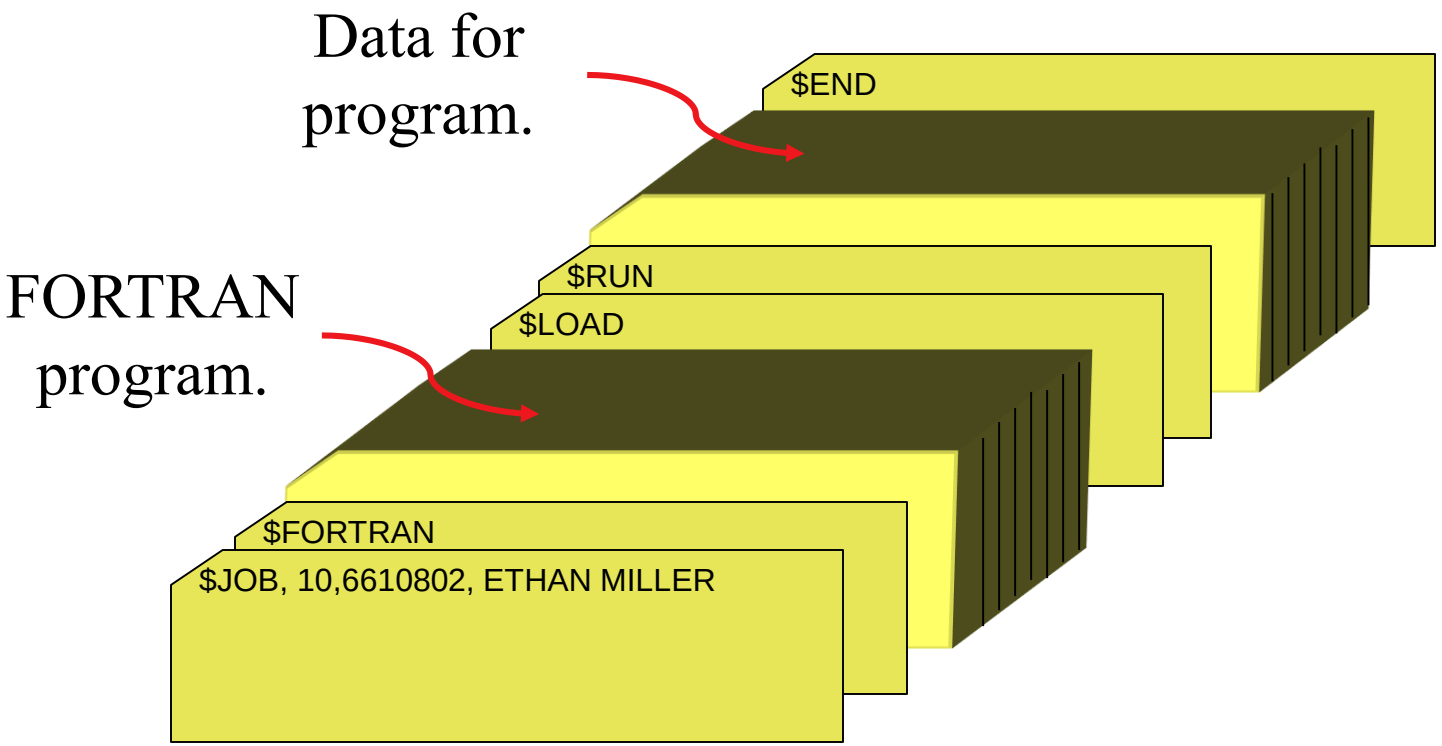
Read cards onto input tape.

Put input tape on 7094.

Perform the computation, writing results to output tape.

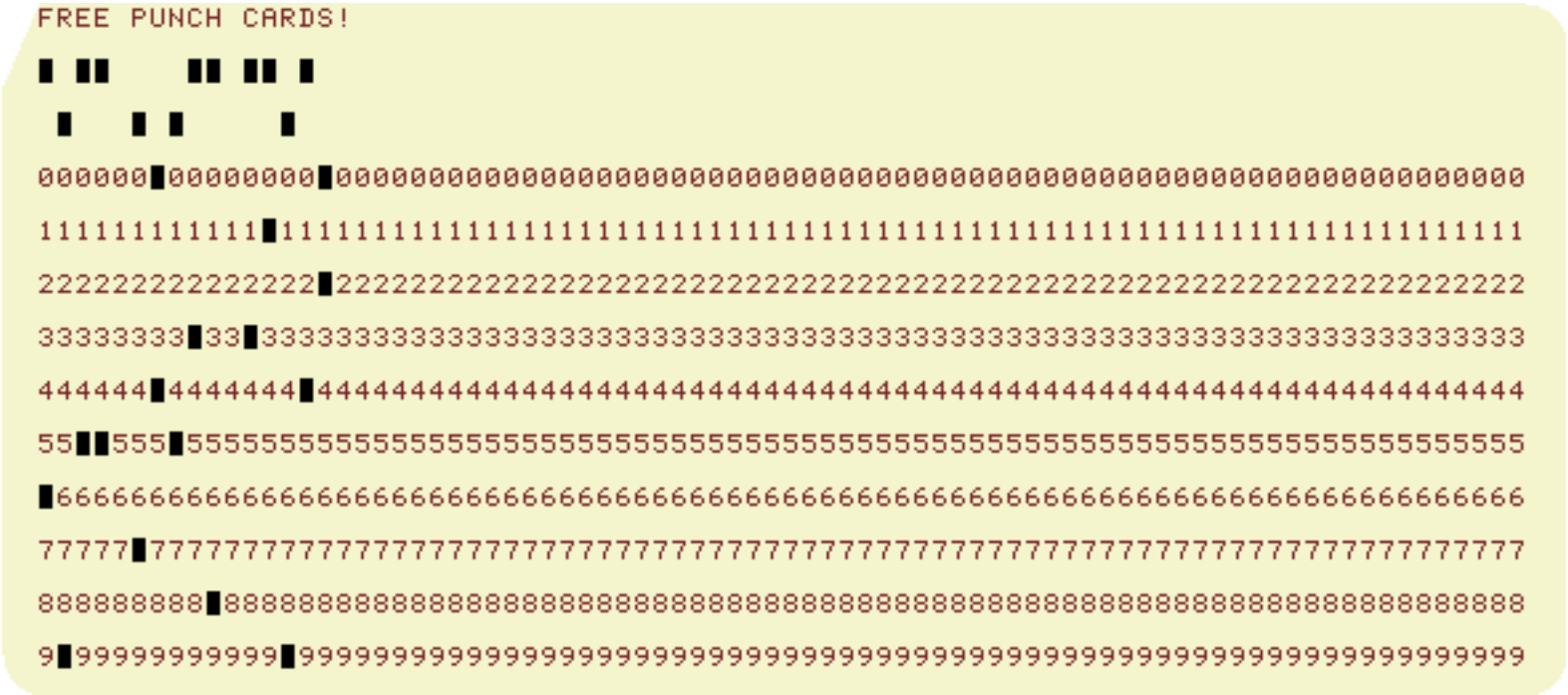
Put output tape on 1401, which prints output.

Typical 2nd generation job



Typical 2nd generation input

Punch card.



Second Generation (1955-1965): Batch Systems

IBM 7094/1401.

Programs were written on paper in either FORTRAN or assembly.

This then was put on punched cards.

The card deck was taken down to the input room and handed to one of the operators.

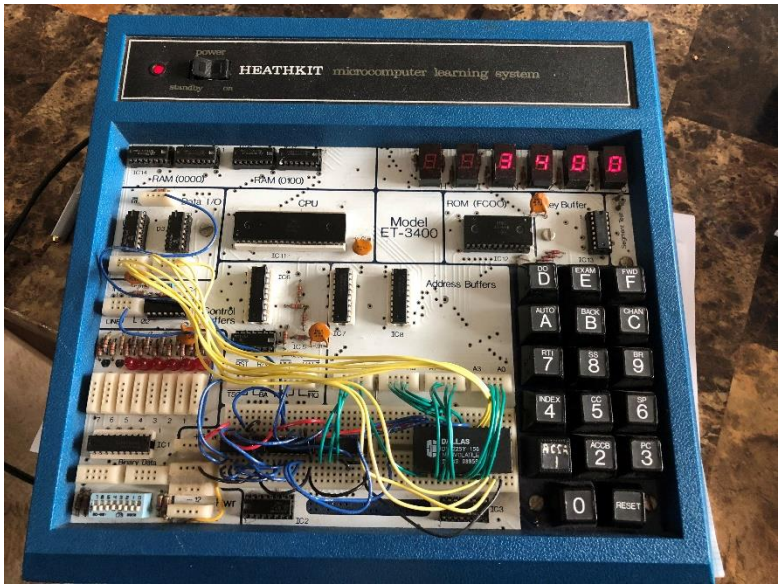
Programmer would come back later for results.

Programs on Punched Card

Z(1) = Y + X(1) //Fortran statement.

Minicomputer and I/O devices

Perkins-Elmer terminal



Heathkit ET-3400
CPU: MC6800
RAM: 256 bytes
ROM: 1K

Minicomputer and I/O devices

Data General Eclipse

Data General Hard disk

Spooling

Eventually tape drives were replaced with disks.

Disks enabled simultaneous peripheral operation on-line (spooling).

- Computer overlapped entering punched cards to disk of one job with execution of another.
- Still only one job active at any given time.
- CPU often underutilized.
  - Example: What if the job needs data from the disk?

OS/360

A first example of an OS for this generation is IBM's OS/360.

Considered one a landmark operating systems.

Other Famous Quotes

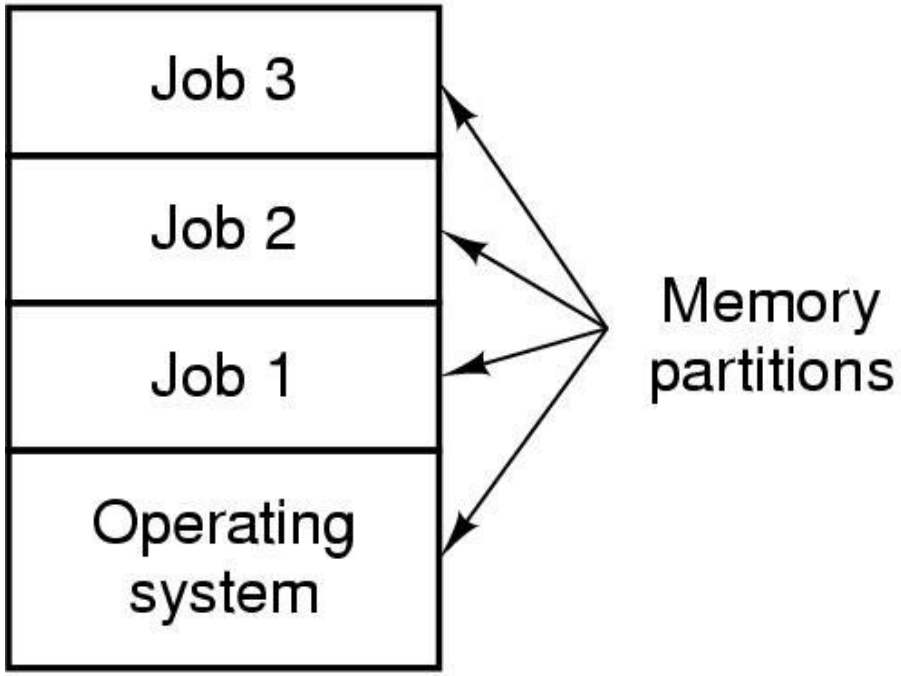
"I have traveled the length and breadth of this country and talked with the best people, and I can assure you that data processing is a fad that won't last out the year."

The editor in charge of business books for Prentice Hall – 1957.

" But what is it good for?"

IBM Engineering talking about the transistor.

Third Generation: Multiprogramming (1965-1980)



- Multiple jobs in memory.
  - Protected from one another.
- Operating system protected from each job as well.
- Overlap I/O of one job with computing from another job.

Third Generation: Multiprogramming (1965-1980)

Multiprogramming allowed several jobs to be active at one time.

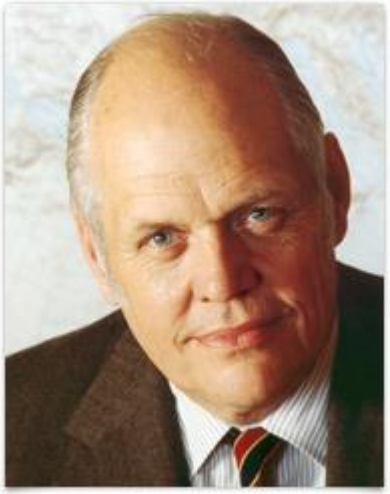
Computer use got much cheaper and easier.

These developments allowed for interactive use.

Yet Another Quote

"There is no reason anyone would want a computer in their home."

Ken Olson, president, chairman and founder of Digital Equipment Corp. – 1977.



Fourth Generation (1980-) Personal Computers

Personal computing changed the computing industry.

Intel came out with the 8080 in 1974.

Lots of companies produced complete systems.

The Control Program for Microcomputers (CP/M) from Digital Research was used.

Altair 8080, 1975:

256 bytes of memory.
Enter data through one of toggle switches on the front.
Results were indicated by flashing lights on the front panel.

Fourth Generation (1980-) Personal Computers

Motorola produced an 8-bit microprocessor, the 6800.

Group of Motorola engineers left to form a new company to manufacture the MOS 6502 after Motorola rejected it.

6502 CPU was used in early systems including Apple I.

- Steve Wozniak and Steve Jobs.
- 1 MHz CPU, 4 KB of RAM, 256 B of ROM.

Apple Introduces the First Low Cost Microcomputer System with a Video Terminal and 8K Bytes of RAM on a Single PC Card.

Apple 1

The Apple Computer. A truly complete microcomputer system on a single PC board. Based on the MOS Technology 6502 microprocessor, the Apple also has a built-in video terminal and sockets for 8K bytes of on-board RAM memory. With the addition of a keyboard and video monitor, you'll have an extremely powerful computer system that can be used for anything from developing programs to playing games or running BASIC.

Combining the computer, video terminal and dynamic memory on a single board has resulted in a large reduction in chip count, which means more reliability and lowered cost. Since the Apple comes fully assembled, tested & burned-in and has a complete power supply on-board, initial set-up is essentially "hassle free" and you can be running within minutes. At $666.66 (including 4K bytes RAM!) it opens many new possibilities for users and systems manufacturers.

You Don't Need an Expensive Teletype.

Using the built-in video terminal and keyboard interface, you avoid all the expense, noise and maintenance associated with a teletype. And the Apple video terminal is six times faster than a teletype, which means more throughput and less waiting. The Apple connects directly to a video monitor (or home TV with an inexpensive RF modulator) and displays 960 easy to read characters in 24 rows of 40 characters per line with automatic scrolling. The video display section contains its own 1K bytes of memory, so all the RAM memory is available for user programs. And the Keyboard Interface lets you use almost any ASCII-encoded keyboard.

The Apple Computer makes it possible for many people with limited budgets to step up to a video terminal as an I/O device for their computer.

No More Switches, No More Lights.

Compared to switches and LED's, a video terminal can display vast amounts of information simultaneously. The Apple video terminal can display the contents of 192 memory locations at once on the screen. And the firmware in PROMS enables you to enter, display and debug programs (all in hex) from the keyboard, rendering a front panel unnecessary. The firmware also allows your programs to print characters on the display, and since you'll be looking at letters and numbers instead of just LED's, the door is open to all kinds of alphanumeric software (i.e., Games and BASIC).

8K Bytes RAM in 16 Chips!

The Apple Computer uses the new 16-pin 4K dynamic memory chips. They are faster and take ¼ the space and power of even the low power 2102's (the memory chip that everyone else uses). That means 8K bytes in sixteen chips. It also means no more 28 amp power supplies.

The system is fully expandable to 65K via an edge connector which carries both the address and data busses, power supplies and all timing signals. All dynamic memory refreshing for both on and off-board memory is done automatically. Also, the Apple Computer can be upgraded to use the 16K chips when they become available. That's 32K bytes on-board RAM in 16 IC's—the equivalent of 256 2102's!

A Little Cassette Board That Works!

Unlike many other cassette boards on the marketplace, ours works every time. It plugs directly into the upright connector on the main board and stands only 2" tall. And since it is very fast (1500 bits per second), you can read or write 4K bytes in about 20 seconds. All timing is done in software, which results in crystal-controlled accuracy and uniformity from unit to unit.

Unlike some other cassette interfaces which require an expensive tape recorder, the Apple Cassette Interface works reliably with almost any audio-grade cassette recorder.

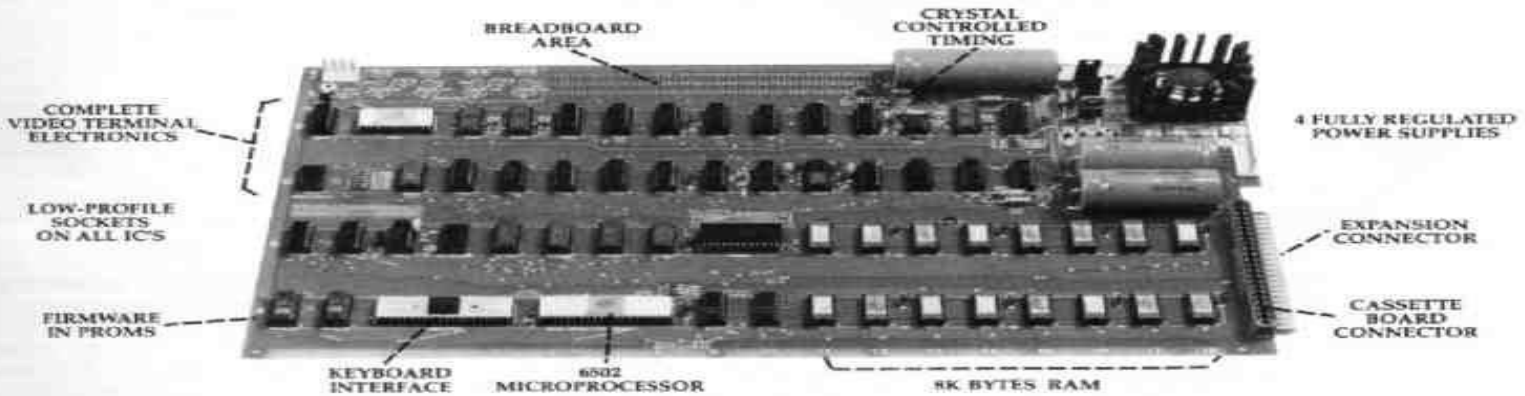
Software:

A tape of **APPLE BASIC** is included free with the Cassette Interface. Apple Basic features immediate error messages and fast execution, and lets you program in a higher level language immediately and without added cost. Also available **now** are a dis-assembler and many games, with many software packages, (including a macro assembler) in the works. And since our philosophy is to provide software for our machines free or at minimal cost, you won't be continually paying for access to this growing software library.

The Apple Computer is in stock at almost all major computer stores. (If your local computer store doesn't carry our products, encourage them or write us direct). **Dealer inquiries invited.**

Byte into an Apple $666.66*

* includes 4K bytes RAM



Fourth Generation (1980-) Personal Computers

Now came the 16-bit systems with Intel's 8086.

IBM designed the IBM PC around the 8088 (an 8086 on the inside with an 8-bit external data path).

IBM needed an OS for their PCs; CP/M behind schedule.

Who did they turn to?

IBM PC, 1981.

- Retailed at $2880.
- 64 kilobytes of RAM.
- Single-sided 160K 5.25 floppy drive.

Fourth Generation (1980-) Personal Computers

Bill Gates suggested IBM that they should look at CP/M (one of the most successful OS for microcomputers at that time, by Gary Kildall).

The biggest mistake of all:

Kildall refused to sign a non-disclosure agreement.

Fourth Generation (1980-) Personal Computers

IBM went back to Bill Gates.
Gates offered an OS called **DOS**.
DOS came from a company called Microsoft.
Microsoft hired Bill Gates to improve it.
The new OS was renamed MS-DOS.

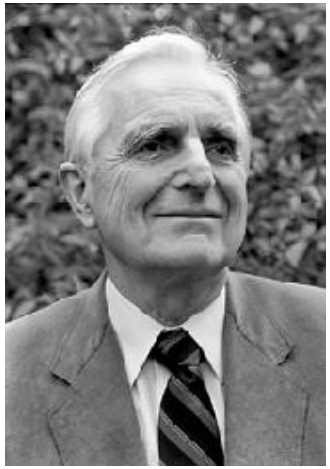
Famous Quote

- "We don't see Windows as a long-term graphical interface for the masses."
  - A Lotus Software Development official, while demonstrating a new DOS version – 1989.

Fourth Generation (1980-) Personal Computers

Up to this point all operating systems were command line.

Doug Englehart at Stanford invented the Graphical User Interface (GUI).



Fourth Generation (1980-) Personal Computers

Steve Jobs saw the possibility of a user-friendly PC.

This led to the Apple Macintosh in 1984.

Steve Jobs was also the co-founder of Pixar which has created very successful animated films: Toy Story ; A Bug's Life; Toy Story 2; Finding Nemo; Monsters.

Fourth Generation (1980-) Personal Computers

Used Motorola's 16-bit 68000.

64 KB of ROM.

Of course, it had the first GUI.

Apple only started using Intel processors in 2006.

What about UNIX?

Let's go back to the 60's.

MULTICS was the first large timesharing system developed jointly between MIT, General Electric (computing division eventually sold to Honeywell) and Bell Labs.

MULTICS introduced many good ideas.

But… OS was written in a language called PL/1.

Not a lot of these got sold but they were very popular with those who bought.

Last one was put out of commission in 2000.

It was owned by the Canadian Department of National Defense.

MULTICS

What about UNIX?

One of the computer scientists at Bell Labs who worked on MULTICS was Ken Thompson.

He found a small PDP-7 minicomputer that no one was using.

He decided to write a stripped-down, one-user version of MULTICS in the C programming language.

This became UNIX.

This was open source which led to other versions: System V (AT&T) and BSD (Berkeley Software Distribution).

What about MINIX?

Eventually AT&T realized that UNIX was commercially viable.

Unix, Version 7's license prohibited the source code from being studied in courses.

A computer scientist, Andrew Tanenbaum, was involved.

He created a new OS (using the C programming language) from scratch that would be compatible with UNIX but completely different on the inside.

This was MINIX or mini-Unix; released in 1987.

Better structured then UNIX.

MINIX-2 released in 1997.

MINIX-3 released in 2006.

MINIX-3

Minix-3: about 30,000 lines.

Other operating systems source is:

http://en.wikipedia.org/wiki/Lines_of_code

https://en.wikipedia.org/wiki/Comparison_of_operating_systems

Vista: 50 million.

Mac OS X 10.4: 86 million.

Linux kernel 2.6.0: 5.2 million.

LINUX

After MINIX was released a USENET newsgroup (think of this as a chatroom), *comp.os.minix* was formed.

Quickly had 40,000 subscribers who wanted to add stuff.

One was a Finnish student named Linus Torvalds.

LINUX

Torvalds wanted to add features which led to other things.

Eventually this led to his own OS called Linux (August 1991).

Linux is a notable success of the open-source movement.

Next Generation?

What do you think will constitute the next generation of operating systems?

Do you think we will be able to interact with systems in a more "natural way"?

Do you foresee eye gaze, brain signals as possible interactions ways too?

Summary

We have discussed what is an operating system.

We have shown examples of why you should want to know more.

We have looked at a brief history of operating systems.

Now it is time to learn more about the insides of an operating system.