Evolution of Operating Systems

In the beginning

1st generation (1945-1955)
Hardware – expensive; Human – cheap

- Built using *vacuum tubes*
- Bare machine: no operating system
- No programming languages, assemblers, compilers…
- Single group of people designed, built, programmed, operated & maintained computer
- Programming in absolute machine language
  - Initially: manually use plugboards
  - Later: use punched cards for writing program
- *Very tedious & error prone*

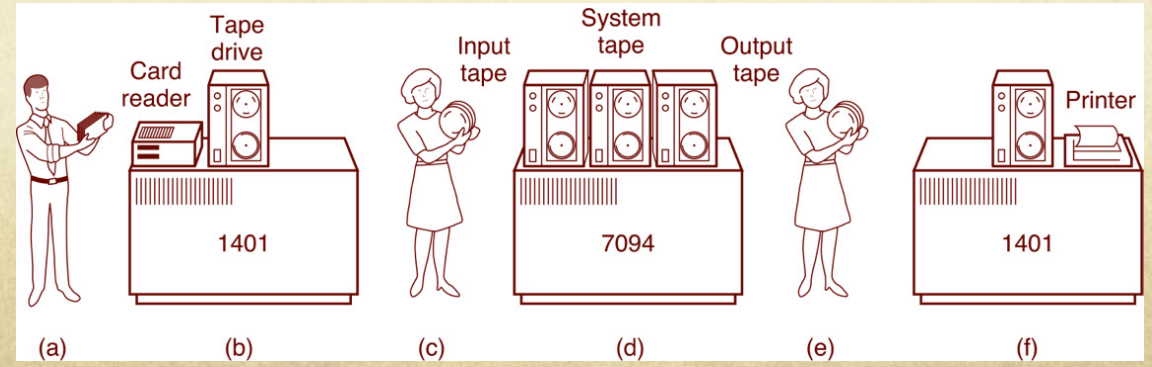
Introduction of early software

2nd generation (1955-1965)
Mainframes

- Mainframes built using *transistors*
- Separation between programmers, operators, etc.
- Introduction of programming languages, assemblers, compilers, etc.
- Early mainframes
  - Programmer submits card deck (single *job*) to operator
  - Card deck processed & output printed (w/ operator intervention)
  - Output returned to programmer by operator
  - Steps repeated for next job
- *Inefficient use of expensive resources*
  - *Low processor utilization, high setup time*

Introducing *batch* systems...

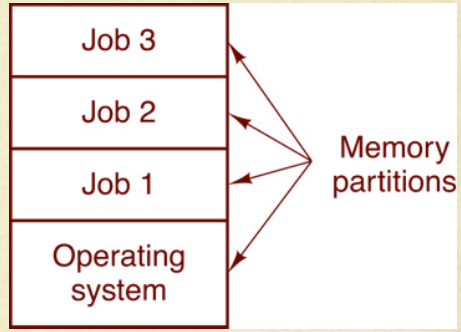
- Programmers submit card decks
  - Programs written in FORTRAN/assembly language
- Cards put on tape
  - Small, less expensive computer used (e.g., IBM 1041)
- Tape loaded by operator onto tape deck
  - More expensive computer used for actual processing (e.g., IBM 7094)
- Rudimentary OS loaded by operator (e.g., FMS/IBSYS)
  - Read next job – process job – write output to tape *[and repeat...]*
- Batch output tape processed by operator & printed on printer



More sophisticated concepts

3rd generation (1965-1980)
Faster, cheaper computers

- *Integrated circuits* came into use (e.g., in IBM 360)
- Concept of *multiprogramming* (e.g., in OS/360)
  - Partition memory into pieces for different jobs
  - → *Special hardware included to ensure protection between jobs*
  - While one job waiting for I/O, process another job



- Concept of *SPOOLING* (*Simultaneous Peripheral Operation On Line*)
  - Use disk (random access device) as large storage
    - Read as many input files as possible
    - Store output files until output devices are ready to accept them
  - Allows overlap – I/O of one job with computation of another
- *Still essentially batch processing systems*
  - *Large turnaround times for jobs → Errors in program very costly*

Timesharing

Hardware – getting cheaper; Human – getting expensive

- Provide fast interactive service to some users
- Execute large batch jobs in background
- Introduction of MULTICS (*MULTIplexed Information and Computing Service)*
- Programs queued for execution in FIFO order
- Like multiprogramming, but timer device interrupts after a quantum (timeslice)
  - Interrupted program is returned to end of FIFO
  - Next program is taken from head of FIFO

Improved operating systems

- OS/360, MULTICS
  - *Enormous & complex (over-engineered!)*
- Birth of *UNIX* (and later, *Linux*)

Personal Computers

4th generation (1980s – present)
Hardware –cheap; Human –expensive

- Single user systems, portable
- I/O devices: keyboards, mice, display screens, small printers
- Laptops and palmtops, Smart cards, Wireless devices
- May not need advanced CPU utilization/protection features
- Birth of *CP/M*, *MS-DOS*, *Mac OS X*, *Windows…*
- *Convenient for user, responsive, ubiquitous*

Real-time systems

- Correct system function depends on timeliness
- Need special OS to ensure timeliness
- Hard real-time systems –
  - Failure if response time too long
  - Secondary storage is limited
- Soft real-time systems
  - Less accurate if response time is too long
  - Useful in applications such as multimedia, virtual reality