Introduction to Operating Systems and SQL for Data Science

Lecture 1 – Introduction to OS, processes

Lecturer: Shay Sakazi

Teaching Assistant : Shlomit Harush

Course Outline

- **Operating Systems**
  - Introduction, processes.
  - Scheduling
  - Threads, race conditions, deadlock
  - Memory management, Garbage collection[ITA]
  - File systems, i-nodes(index-nodes)

Course Outline - continue

- **Relational Databases**
  - Introduction to database management system (DBMS)
  - The relational model
  - Relational Algebra
  - SQL language
  - Normalization
  - Semantic models, Entity relation(ER)
  - Transactions
  - Analyzing Databases and Data frames using Python and SQL

Course preliminaries

- Basic Python knowledge

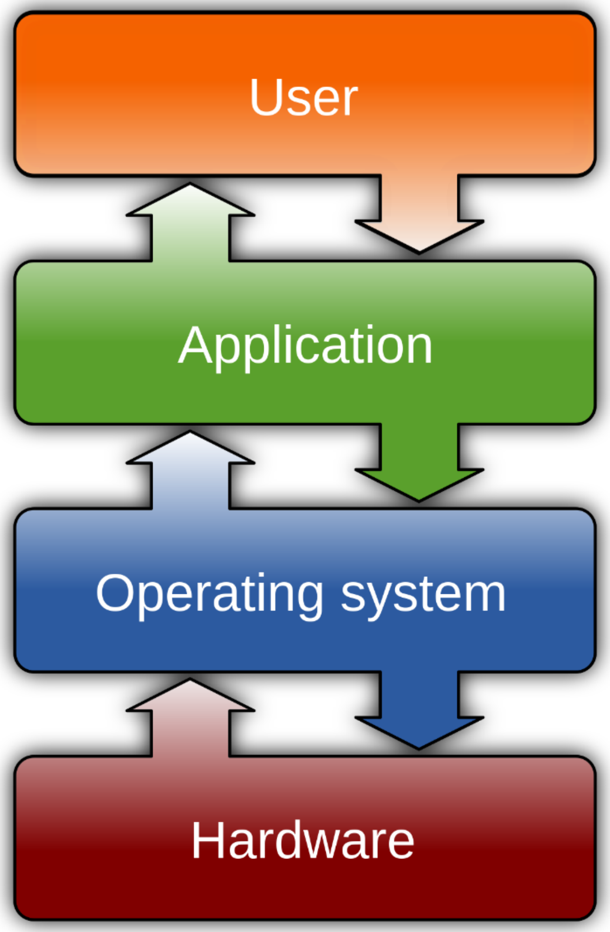
- Basic computer science concepts (Data structures, Loops, etc.)

Homework & Grades

- Exercises (Three or Four, ~two for each topic), 30%
- Final Exam, 70%
- Must pass the test (grade 60)

Operating System

An **operating system (OS)** is system software that manages computer hardware, software resources, and provides (API) common services for computer programs.



Course Material

Operating System:

1. **Modern Operating Systems (4th Ed.), By Andrew S. Tanenbaum.** Link
2. Operating System Internals and Design Principles (7th Ed.), By William Stallings. Link
3. **Elmasri R. & Navathe S. Fundamentals of Database Systems (6th Ed.), By Addison-Wesley.** Link

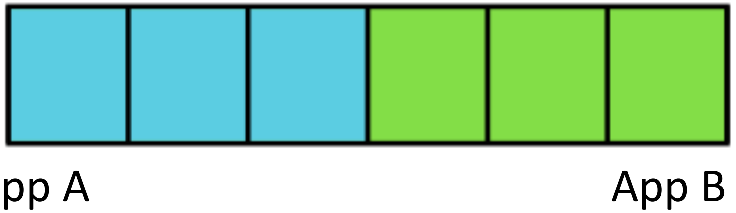
Introduction

Computer Components

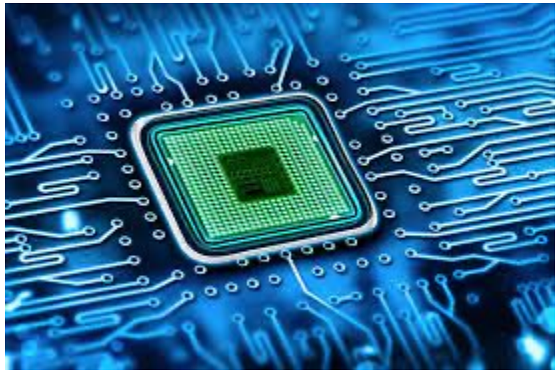
1. CPU (central processing unit) – responsible for the computation.
2. Memory – Different granularities i.e., registers, RAM, Disk.
3. I/O(Input/Output) – keyboard, mouse, screen, etc.

Computer Resources

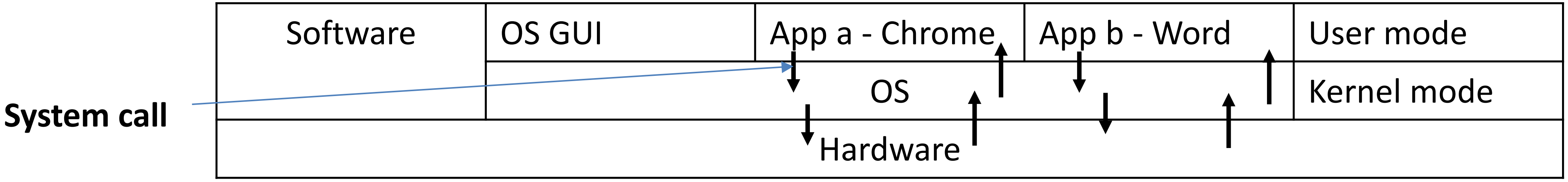
- Memory – the memory allocation is a physical space



- CPU run time – time allocation on CPU



OS-Apps relationship



System call

Software

OS GUI

App a - Chrome

App b - Word

User mode

OS

Kernel mode

Hardware

GUI – graphic user interface

A request of a program/app from the OS called system call

1. A certain app runs on user mode, and request resources from the OS
2. The OS transforms the request to kernel mode.
3. The Kernel mode has access to the Hardware.

OS-Apps relationship

- Most of the command are blocked in the user mode. Some of the OS functions are running in this mode (GUI)

- Only on kernel mode there is an access to the Hardware. Only the OS is running in this mode.

- Why we need this architecture? What could go wrong if we don’t enforce this condition?

Operating System

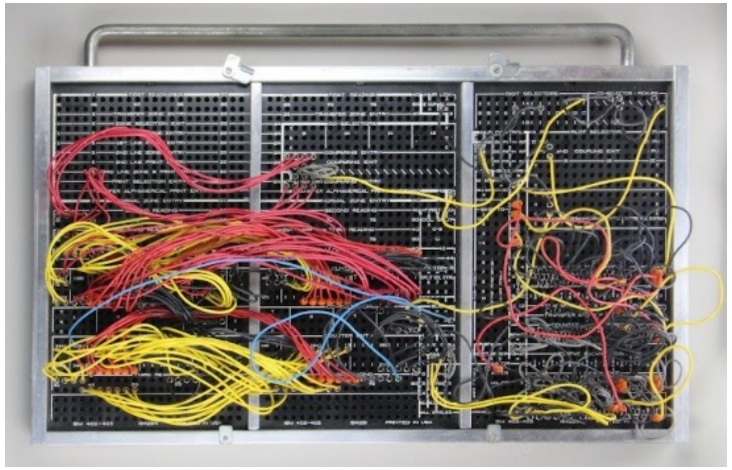
We don’t want that every app builder will understand the computer/device the app will run on.

Why?

Hence, we want to create an abstraction for the system resources.

History of OS

- **First generation (1945-1955)**
  - vacuum tubes, plug boards



- **Second generation (1955-1965)**
  - transistors, batch systems – multiple programs on disk

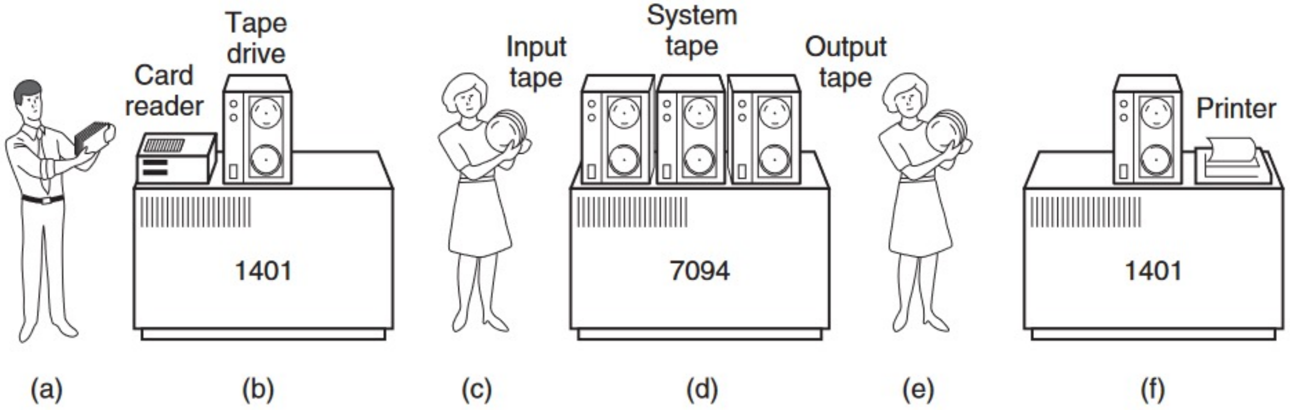


Figure 1-3. An early batch system. (a) Programmers bring cards to 1401. (b) 1401 reads batch of jobs onto tape. (c) Operator carries input tape to 7094. (d) 7094 does computing. (e) Operator carries output tape to 1401. (f) 1401 prints output.

Reichman University

History of OS - continue

- **Third generation (1965–1980)**
  - ICs and multiprogramming -*user interaction (time-sharing)*

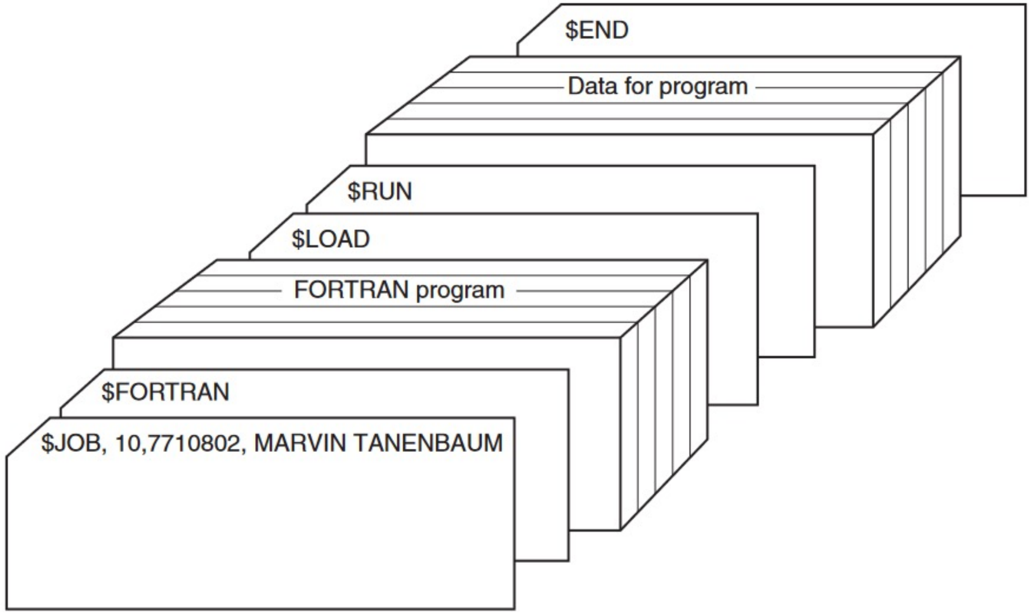


Figure 1-4. Structure of a typical FMS job.

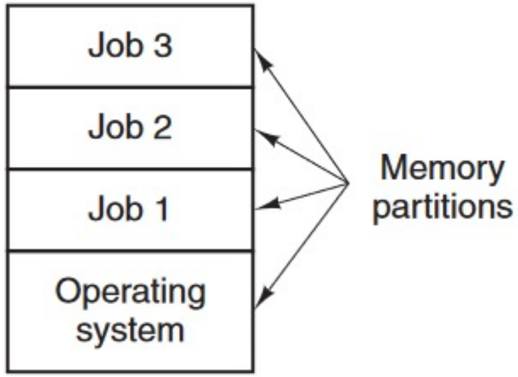
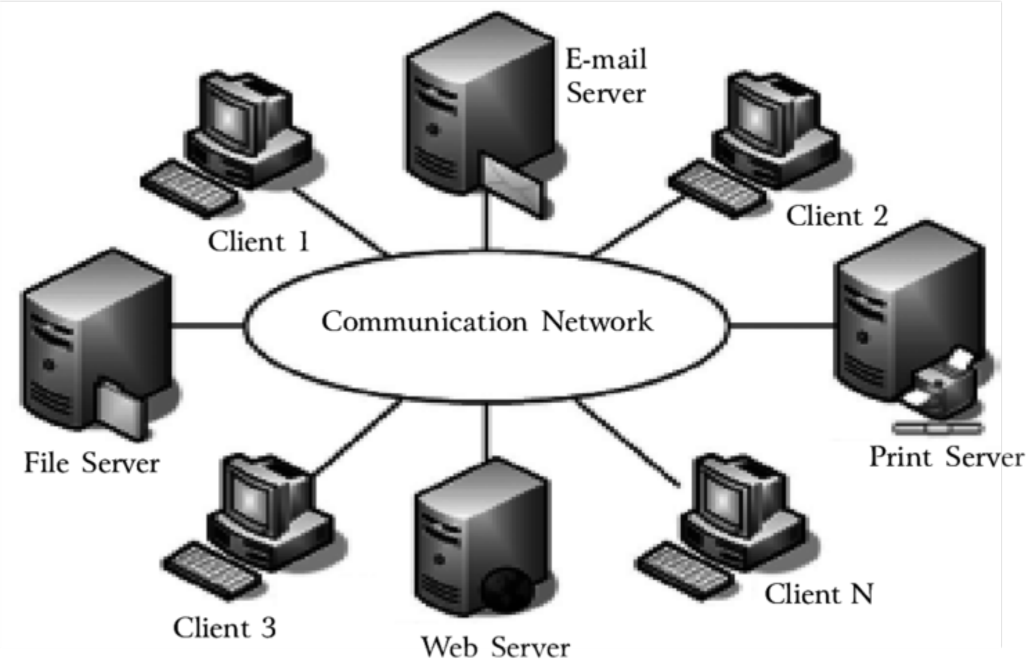


Figure 1-5. A multiprogramming system with three jobs in memory.

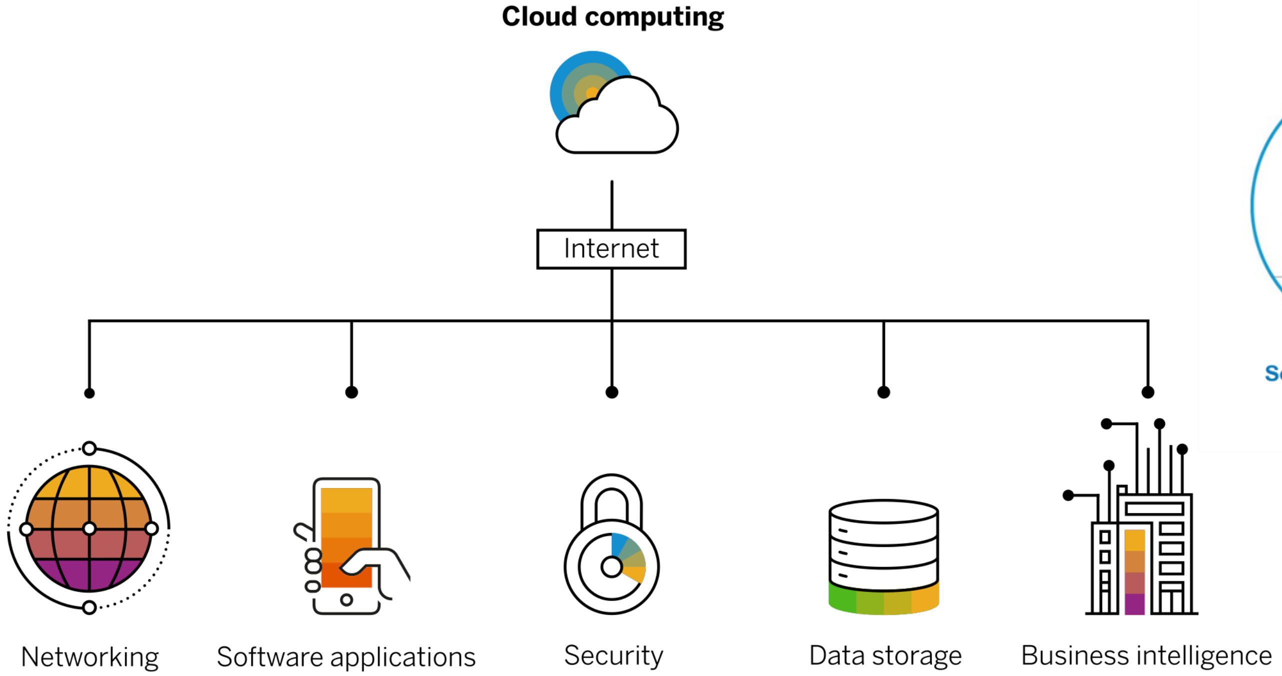
History of OS - continue

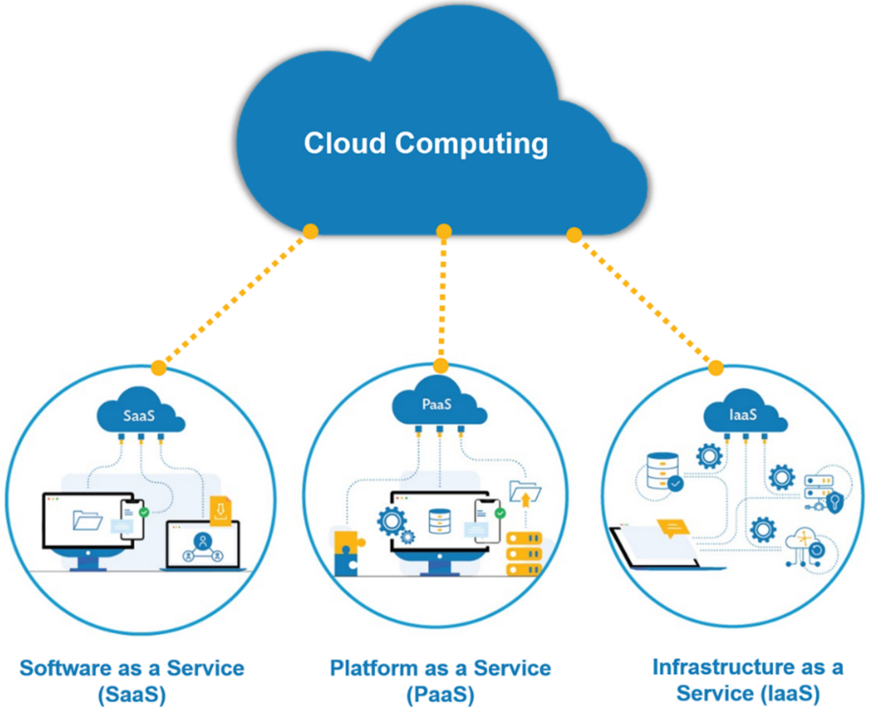
- **Fourth generation (1980–present)**
  - personal computers – *graphic user-interface*
  - Networks – *file & computing services*
  - Web-computing, *Handheld devices , Cellular phones*



History of OS - continue

- **Fifth Generation (Present)**
  - Multiprocessor, Multicore
  - Virtualization, Cloud




Cloud Computing
SaaS
PaaS
IaaS
Software as a Service (SaaS)
Platform as a Service (PaaS)
Infrastructure as a Service (IaaS)

Reichman University

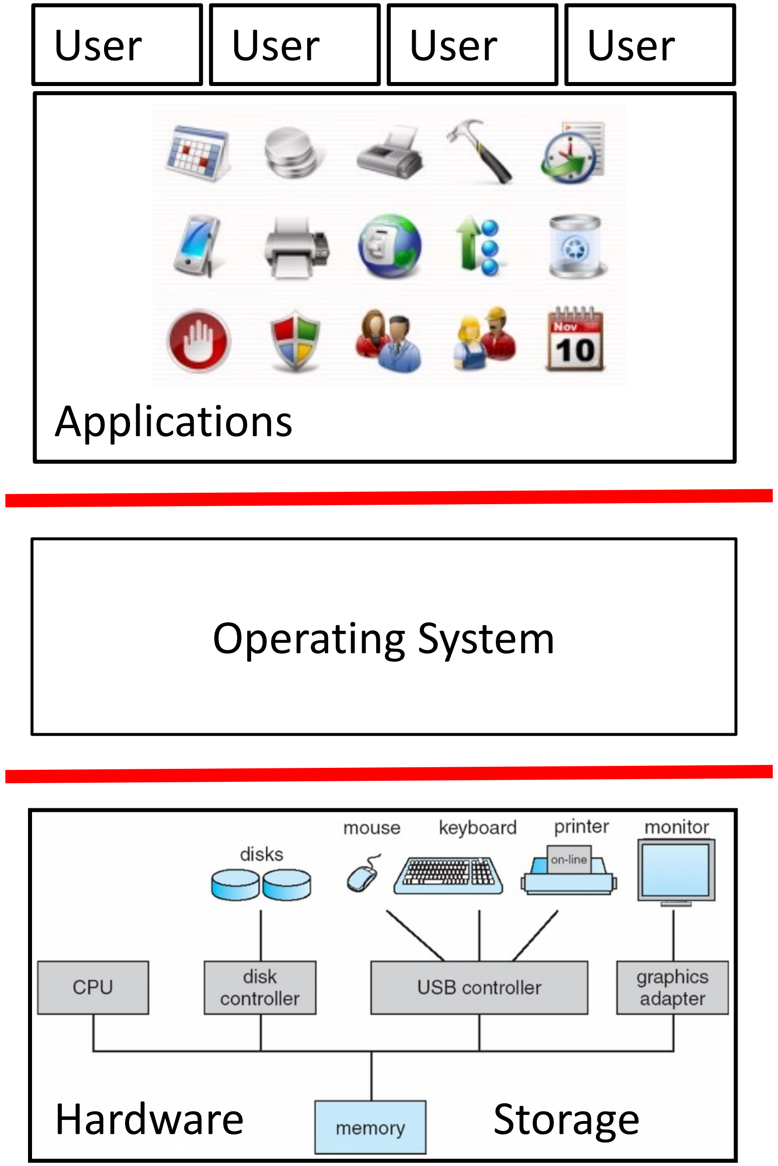
The Operating System Zoo

- Mainframe operating systems
  - VM (IBM), VMS(Compaq)
- Server operating systems
- Multiprocessor operating systems
- Personal computer operating systems
- Handheld computer operating systems

- Embedded operating systems
  - VxWorks
  - Sensor-Node opearting systems
- Real-time operating systems
  - VxWorks
- Smart card operating systems

And many more…

Simplified view of OS



User User User User

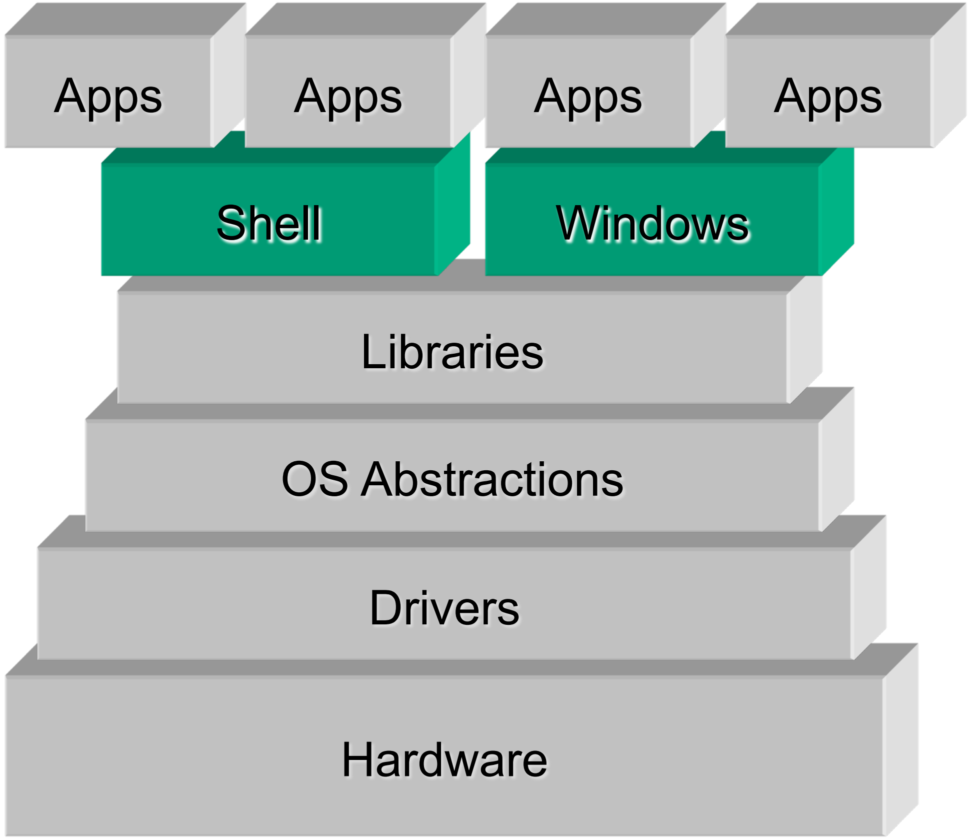
Applications

Operating System

Hardware

Storage

Its much more than that…



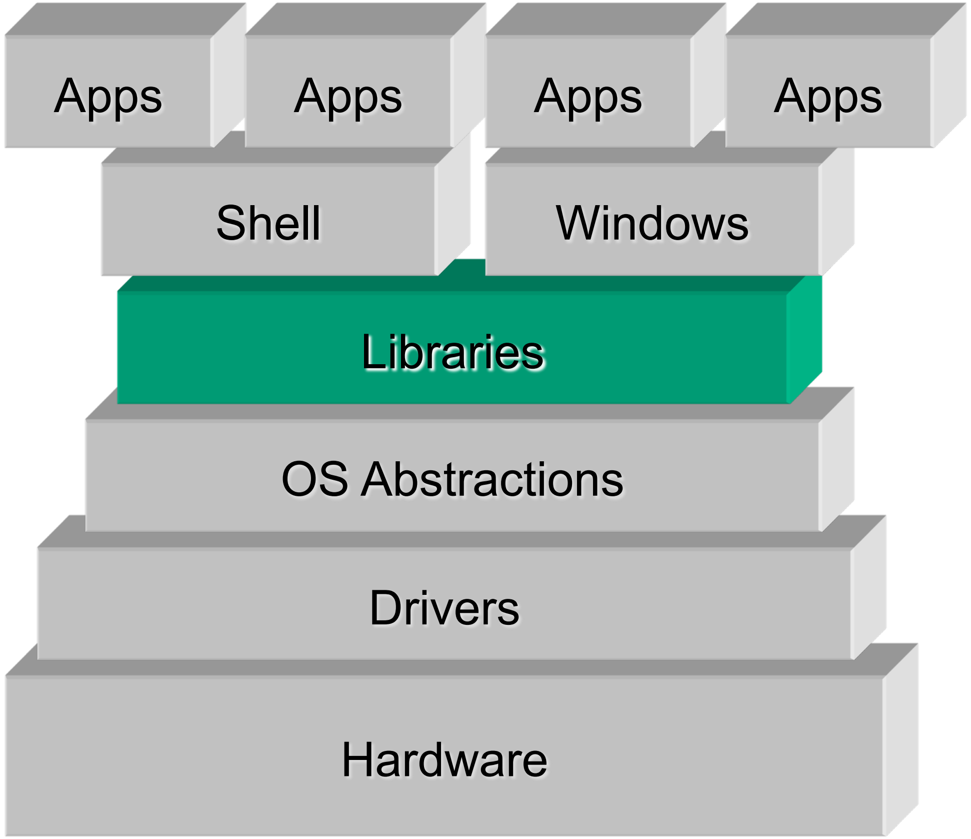
User interface

- Make OS mechanisms available to user
- psychological issues are important

Is a web browser part of an Operating System ?

Reichman University

Layers of System

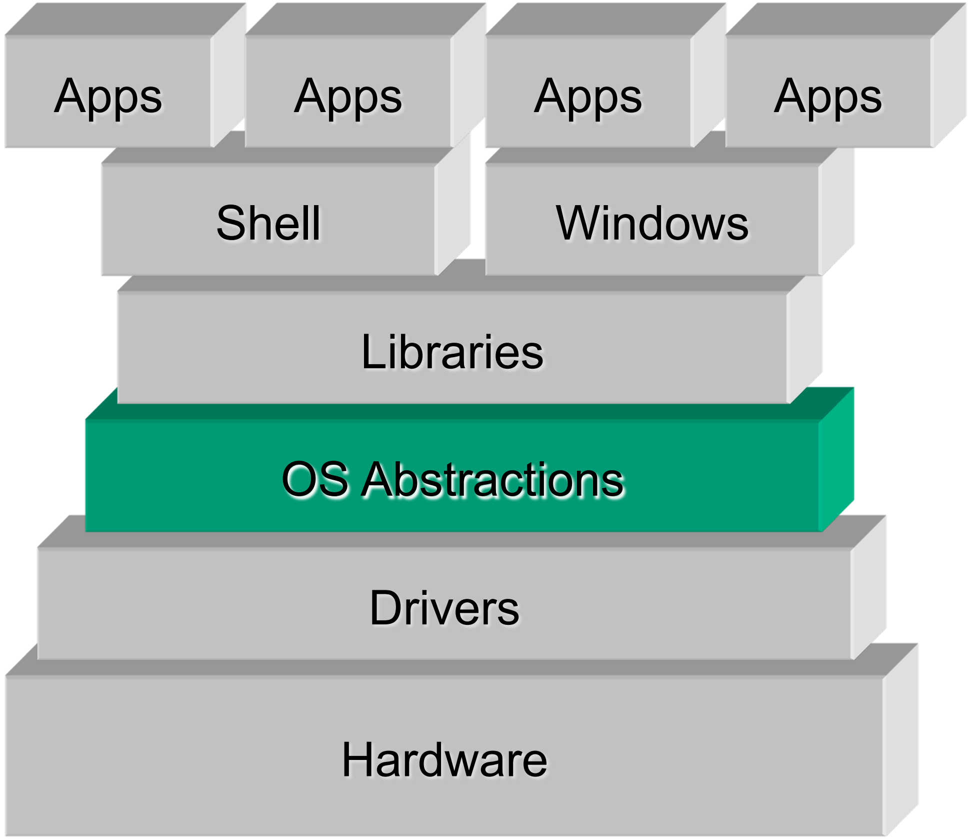


Libraries

- Usually language specific
  - java.io.*, java.net.*
  - stdio.h; stdlib.h
- Often higher level abstractions

Reichman University

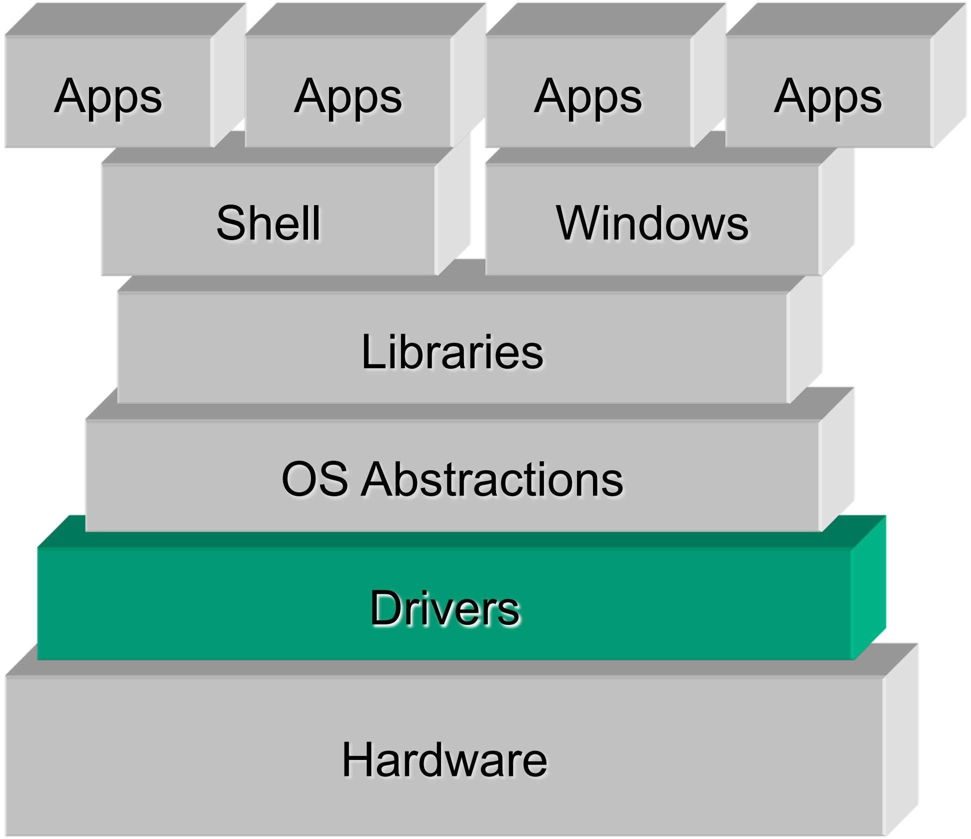
Layers of System



OS Abstractions

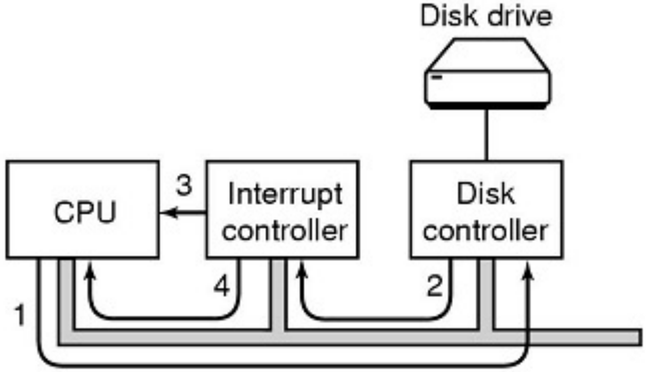
- provide lower level abstractions and mechanisms
- Storage
  - File systems
- Computation
  - processes
- Communications
  - sockets

Layers of System



Hardware drivers

- provide usable interface to hardware



Processes

Processes

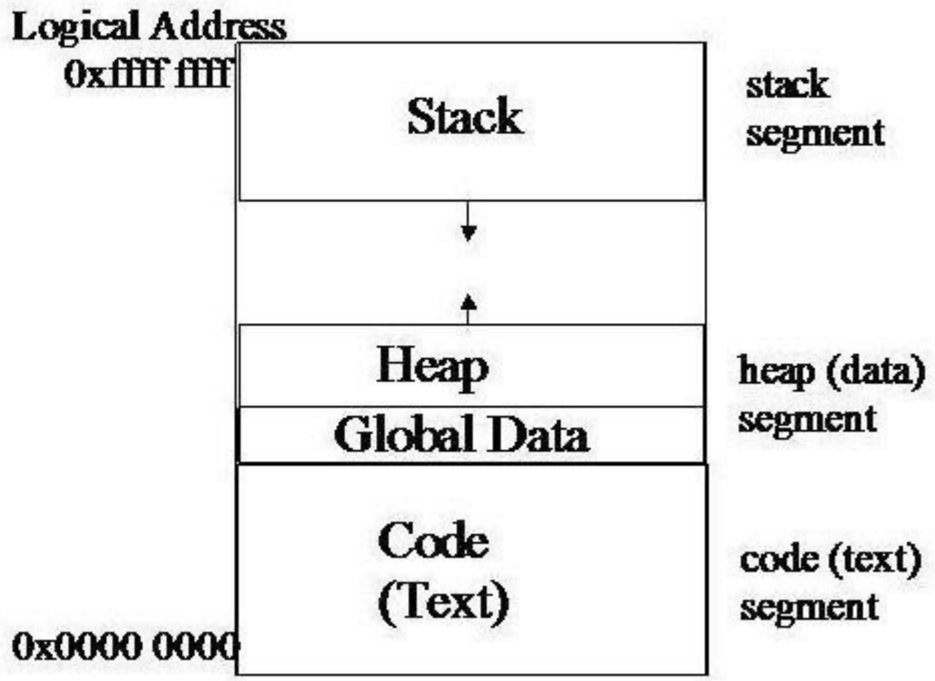
Process – A program in execution, each process has its <u>address space</u>, a list of memory locations from 0 to some maximum which the process can read and write.

Process – address space

The address space contains the executable program, the program’s data, and its stack. Also associated with each process is a set of resources, commonly including registers.

Process – address space

The stack grows downwards while the heap(data) grows upwards. There is a trade-off between them.



Process table

All the information about each process is stored in an operating system table called the process table.

- Times (when the process starts, how much CPU runtime it gets, etc.)
- Process address space
- Process ID (a unique ID per process)
- Etc..

“Concurrency”

We can have multiple processes on the same time (concurrency).

However, the CPU could run only one command at a time.

Thus, the running of multiple processes is Pseudo concurrency.

Reichman University

Process types

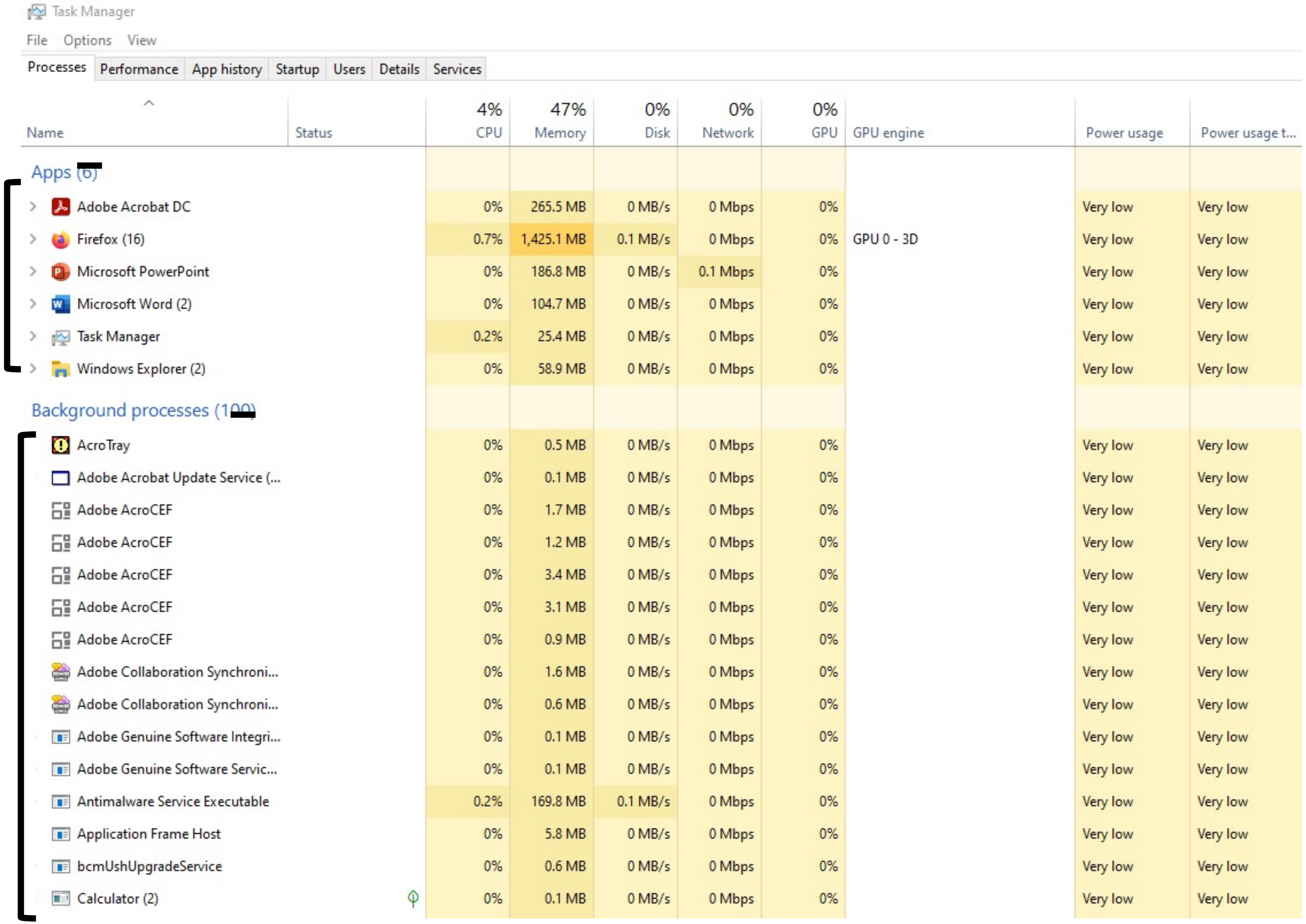
1. Foreground processes – GUI, Chrome, Word, etc.

2. Background processes – anti-virus, email checking, Teams, etc.

Process types - continue

Foreground processes

Background processes



		4%	47%	0%	0%	0%			
Name	Status	CPU	Memory	Disk	Network	GPU	GPU engine	Power usage	Power usage t...
Apps (6)									
Adobe Acrobat DC		0%	265.5 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Firefox (16)		0.7%	1,425.1 MB	0.1 MB/s	0 Mbps	0%	GPU 0 - 3D	Very low	Very low
Microsoft PowerPoint		0%	186.8 MB	0 MB/s	0.1 Mbps	0%		Very low	Very low
Microsoft Word (2)		0%	104.7 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Task Manager		0.2%	25.4 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Windows Explorer (2)		0%	58.9 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Background processes (100)									
AcroTray		0%	0.5 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe Acrobat Update Service (...		0%	0.1 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe AcroCEF		0%	1.7 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe AcroCEF		0%	1.2 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe AcroCEF		0%	3.4 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe AcroCEF		0%	3.1 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe AcroCEF		0%	0.9 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe Collaboration Synchroni...		0%	1.6 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe Collaboration Synchroni...		0%	0.6 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe Genuine Software Integri...		0%	0.1 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Adobe Genuine Software Servic...		0%	0.1 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Antimalware Service Executable		0.2%	169.8 MB	0.1 MB/s	0 Mbps	0%		Very low	Very low
Application Frame Host		0%	5.8 MB	0 MB/s	0 Mbps	0%		Very low	Very low
bcmUshUpgradeService		0%	0.6 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Calculator (2)		0%	0.1 MB	0 MB/s	0 Mbps	0%		Very low	Very low

Reichman University

Code(program) VS. Process

We can run the same code on many processes.

But each process has only one code.

$$\text{Code}_{1}\text{———}{}_{\infty}\text{Process} = \text{Class}_{1}\text{———}{}_{\infty}\text{Object}$$

Process creation

1. When we open the computer, the OS is starting (in modern OS's the OS composed from one or many processes).
2. The user is starting a process (e.g., double click on an icon using the GUI).
3. A process is creating another process.

Process creation - Windows

In C# we can create a process in the following way:

```
Process p = new Process(…)
p.start
```

Reichman University

Process creation - Linux

https://www.youtube.com/watch?v=ss1-REMJ9GA

- In Linux, the command is fork().
- It creates a copy of the process; the new process loads a new code for execution.
- The command(fork) return a value of the process ID which differentiates the new process from the original.
- In a new process the pid is -1.

Fork code example

```
for(int i=0; i<5; i++){
    fork();
    print(i);
```

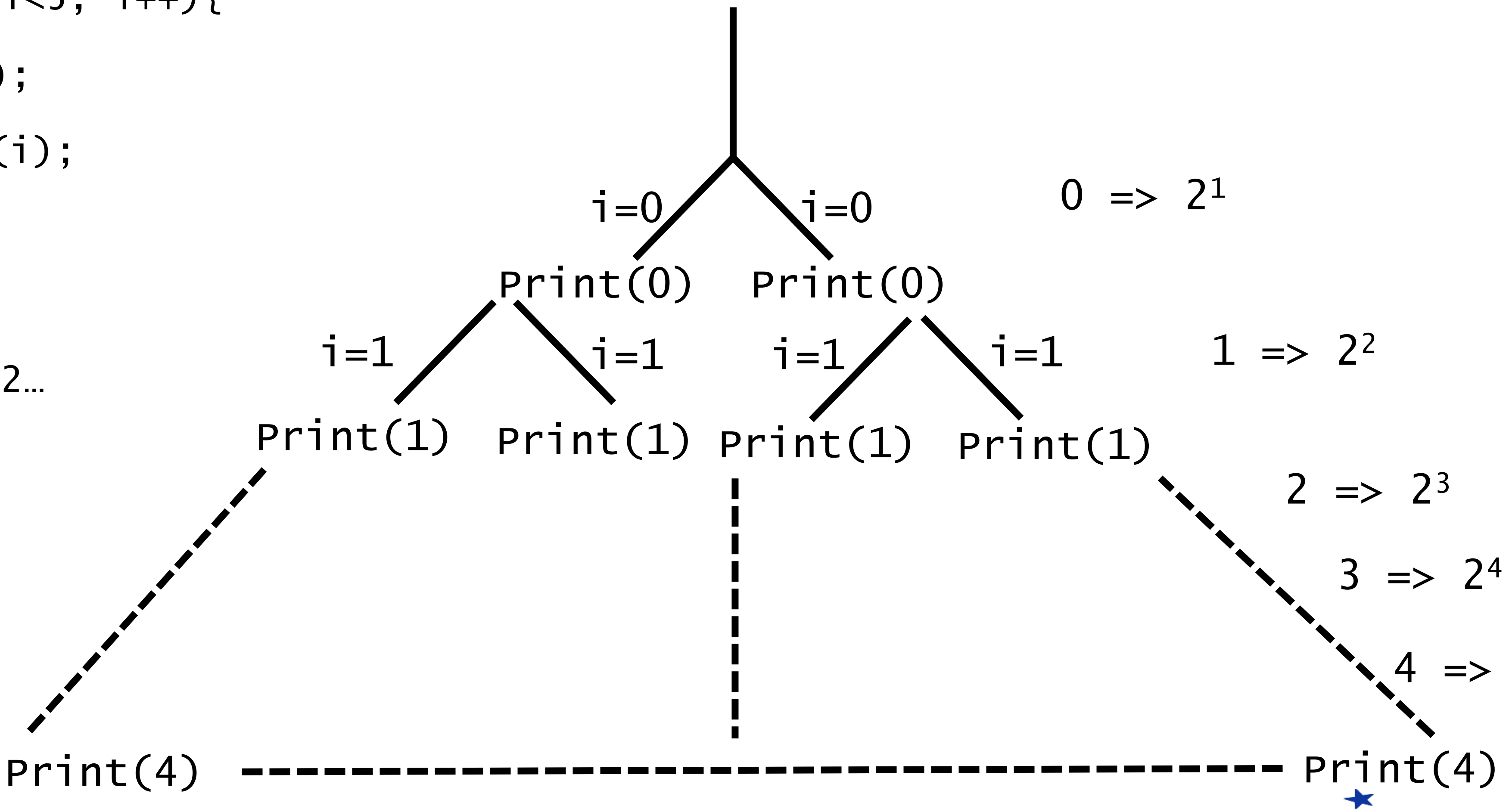
What will be printed?

Fork code example

```
for(int i=0; i<5; i++){
        fork();
        print(i);
```

Print:

0011112222222…



0 => 2^1

1 => 2^2

2 => 2^3

3 => 2^4

4 => 2^5