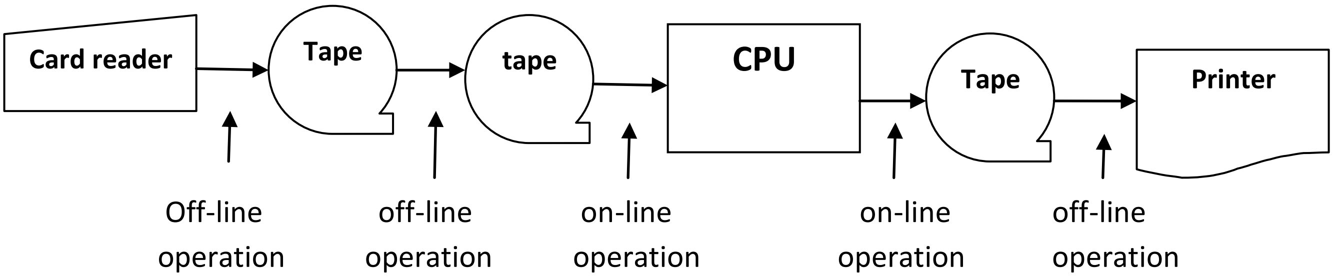
1.7 Performance Development

1.7.1 On-Line and Off-Line operations

1. **On-Line Operation:** in which they are connected to the processor.

2. **Off-Line Operation:** in which they are not connected to the central C/S.



1.7.2 Buffering

A buffer : is an area of primary storage for holding data during I/O transfers.

There are two types of buffering:

1. The single buffer. (the CPU would be idle)
2. The double buffer. (the CPU will not e idle)

Advantages: In buffering the CPU and I/O are both busy.

1.7.3 **Spooling** (Simultaneous Peripheral Operation On- Line)

1. The Spooling Operation uses a disk as a very large buffer for reading and for storing output files.
2. Rather than the cards being read from the card reader directly into memory, the cards are read directly from the card reader onto the disk, the operating system satisfies its requests by reading from the disk, similarly when the job requests the printer to output a line.

Advantages:

Spooling can keep both CPU and the I/O devices working as much higher rates.

1.7.4 Multiprogramming

In multiprogramming system, there are more than one job in memory, when a job may have to wait for any reason such as an i/o, the o/s simply switches to and executes another job.

In a non-multiprogramming system (uni-programming), the CPU would sit idle. (**inefficient**)

Advantages:

1. Maximize CPU utilization. (The CPU will never be idle)

2. High and efficient CPU utilization.

CHAPTER 1 Review

Batch systems	Interactive systems
Lack of interaction between the user and the job.	**Users can interactively run, terminate, and modify their jobs.**
Each user prepares his job off-line.	**Each user types the commands on-line one by one and getting response immediately.**
Suitable for large jobs.	**Suitable for small and medium jobs.**
There is a delay between the job submission and the job completion.	**There is no delay.**
The CPU is often idle.	**The CPU is often busy.**

Program	Process (job)
It is a passive entity.	**It is an active entity.**
Stored on disk (File).	**Stored in memory.**
It is a sequence of instructions (static).	**It is a sequence of actions (dynamic).**

CHAPTER 1 QUESTIONS

1. What is an Operating System ?

2. What are the **purpose** (**goals**) of an operating system?

3. In what system the CPU is often **idle** ? explain the reason.

4. In what kind of processing we can keep both the CPU and the I/O devices working at higher rates ? explain.

5. In what kind of system the **CPU will never be idle**? Explain how this system work, with drawing.

6. In what kind of system the **CPU** and **I/O** are both busy.

(e.g. what is the system that allows overlap operation with processing.)

7. Give three another names for Time-Sharing system.

8. What are the differences between **Batch**-**System** and **Time**-**Sharing System**?

9. What are the differences between **Parallel Systems** and **Distributed System** ?

10. What is **Real**- **Time System**? What are its applications?

11. What is the differences between **On**-**Line** and **Off**-**Line** operation? Give examples.

12. What is **Throughput**?

13. What is the difference between each two of the following:(وزاري ٢٠١٣)

1. Parallel system and Distributed system.

2. Time-sharing system(Multi-tasking) and Multi-programming.

14. What is the reason for building parallel system?

15. <u>Circle</u> the correct answer to the following questions

1. **distributed system is a collection of processers that:**

   A. Share memory & clock **B**. Do not share memory & clock

   C. Share memory nor clock **D**. Share clock nor memory

2. Programs that do not require interaction or programs with long execution time may served well by:

A. Batch System

B. Real time System

C. Time-Sharing System

D. Parallel System

CHAPTER 2

2. Computer System Operations

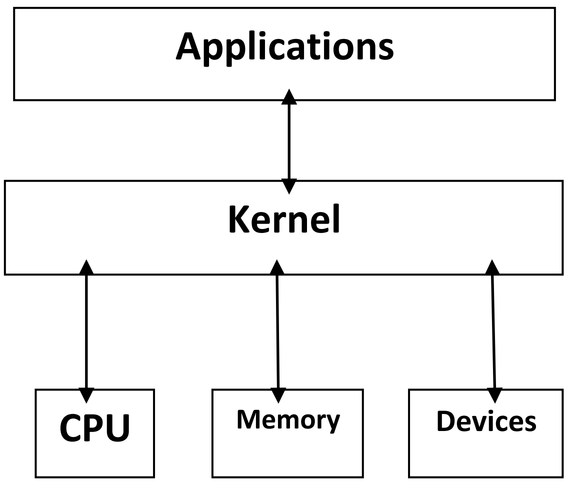
2.1 Bootstrap program: It is an initial program to run, its loaded at power-up or reboot. (stored in **ROM** or **EPROM** known as **firmware**)

2.2 Bootstrap program functions:

1- Initialize all aspects of the system from CPU registers to device controllers to memory contents.

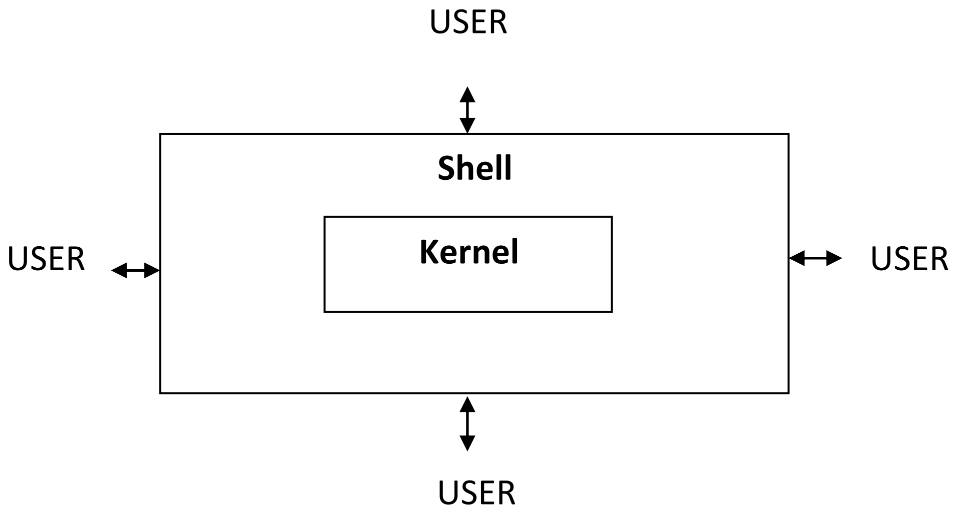
2- Locate and Loads **operating system kernel** into memory and starts execution.

- **Kernel**: it is the lowest level of any O/S and it is running at all times on the computer.



A kernel connects the application software to the hardware of a computer

Shell: Is a software that provides an access to operating system services.



2.3 I/O Structure

A **device driver** : The operating system "calls" the driver, and the driver "drives" the device. So, the driver is the link between the operating system and the device.

- Each I/O device is connected to the C/S through its **device driver**.

A device controller: Is an interface between a device and a device driver. There is always a device controller and a device driver for each device to communicate with OS.

2.4 DMA Structure

1- Used for high-speed I/O devices.

2- Transfers **blocks** of data from buffer storage directly to main memory without CPU intervention.

3- Only one interrupt is generated per **block**, rather than one interrupt per **byte**.

2.5 Storage structure

1- Main memory: large storage media that the CPU can access directly.

2- **Secondary Storage**: Extension of main memory that provides:

1- large, and 2- **nonvolatile** storage capacity.

We want the **programs** and **data** to reside in main memory **permanently**, this is not possible for the following reasons:

1. **Main memory** is usually too small to store all needed programs and data permanently.
2. **Main memory** is a volatile storage device that losses it's contents when power is turn off.

Therefore most computer systems provide **secondary storage** as an extension of main memory, these secondary storage can be able to hold large quantities of data **permanently**(non volatile storage).

Example: Magnetic disk, flash RAMS.

3- **Caching**: faster storage system.

Operating Systems Concepts

Chapter 3

Operating system components

Prepared by Assist . Prof .

Imad Matti

AL-mamoon University College / Computer Science Department

2018

CHAPTER 3

3. O/S System Components

3.1 Process Management

- **Process**: is a program in execution.
- **Process**: is a unit of work within the system.
- **Process**: is an active entity.
- **Program**: is a passive entity.

-Process components:

A process includes: 1- Code to execute. 2- Data to manipulate.

The O/S is responsible for the following activities in connection with process management:

1- Creation and deletion of both user and system processes.

2- Suspension and resumption processes.

3- Providing mechanisms for process synchronization.

4- Providing mechanisms for process communication.

5- Providing mechanisms for process deadlocks handling.

3.2 Memory Management

The O/S is responsible for the following activities in connection with memory management:

1- Keep track of which parts of memory are currently being used and By whom.

2- Decide which processes are to be loaded into memory when space becomes available.

3- Allocate and de-allocate memory space as needed.

3.3 File Management

- **File**: is a collection of related information defined by its creator.
- **Files** represent programs and data.

The O/S is responsible for the following activities in connection with file management:

1- Creation and deletion files.

2- Creation and deletion directories.

3- Mapping files onto secondary storage.

4- backup files on stable (nonvolatile) storage media.

3.4 I/O System Management

I/O Subsystem consists of:

a- Memory management include buffering, catching and spooling.

b- General device driver interface.

c- Driver for hardware devices.

3.5 Secondary Storage Management.

The O/S is responsible for the following activities in connection with secondary storage management:

a- Free Space management.
b- Storage Allocation.
c- Disk scheduling.

3.6 Networking

Collection of processes, each process has its local memory and clock, the processors communicates with one another through communication lines, such as high speed buses or telephone lines.

3.7 Protection

Controlling the access of program, processes.

3.8 Command Interpreter System.

Interface between the user and the O/S.

System Structure

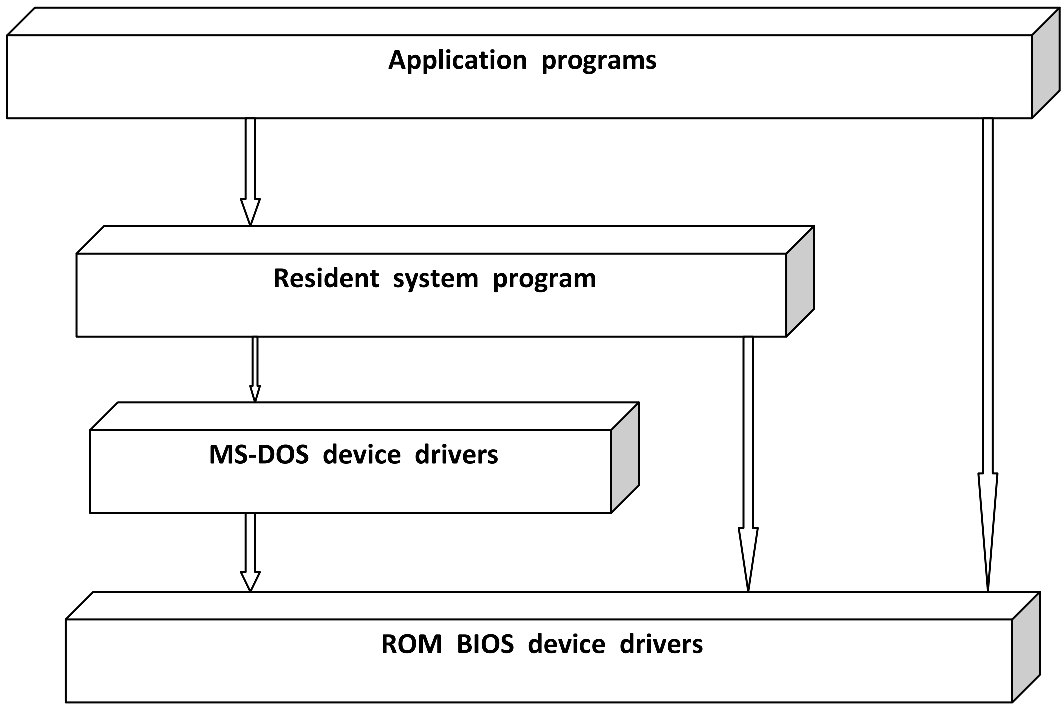
There are two approaches for the O/S structure:

2.1 Simple Structure

Small, simple, and limited systems.

The interfaces and levels of functionality are not well separated.

Example: MS-DOS.



2.2 Layered Approach

Consists of breaking the O/S into number of Layers (levels), each built on top of lower layers. The bottom layer (layer 0) is the H/W, the highest (layer N) is the user interface.

Layer 5:	user program
Layer 4:	buffering for input and output devices
Layer 3:	operator-console device driver
Layer 2:	memory management
Layer 1:	CPU scheduling
Layer 0:	Hardware

The layer structure

Advantages:

a- modularity: The layers are selected such that each uses functions (operations) and services of only lower-level layers.

b- simplifies debugging and system verification: The first layer can be debugged without any concern for the rest of the system.

CHAPTER 3 QUESTIONS

1. State five activities of File- management. (وزاري ٢٠١٦)

2. What is the difference between process and program?

3. state three activities the OS is responsible in connection with process management.

4. state three activities the OS is responsible in connection with process management.

5. state three activities the OS is responsible in connection with secondary storage management.

Operating Systems Concepts

Chapter 4

Process management

Prepared by Assist . Prof .

Imad Matti

AL-mamoon University College / Computer Science Department

2018

CHAPTER 4

4. Process Management

A computer system consists of a collection of processes:

1- O/S processes: execute system code.

2- User processes: execute user code.

4.1 Process Concept

A **Process**: is a program in execution.

Or **process**: is a unit of work.

Or **process**: is an active entity.

Program	Process
It is a passive entity	It is an active entity.
Stored in disk (i.e. file).	Stored in memory.
It is a sequence of instructions (static).	Sequence of actions (dynamic).

4.2 Process States

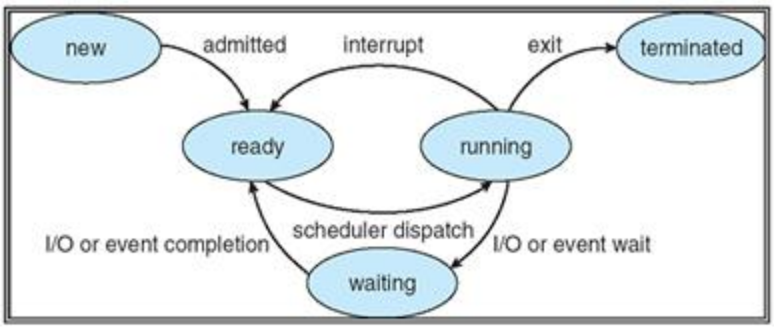
1- **New**: The process is being created.

2- **Ready**: The process is waiting to be assigned to a processor.

3- **Running**: instructions are being executed.

4- **Waiting**: The process is waiting for some event to occur.

5- **Terminated**: The process has finished execution.



4.3 PROCESS CONTROL BLOCK (PCB)

PCB: is a data structure containing all the necessary information for representing a process in the system.

It contains many pieces of information such as:

1- **Process Identifier**: ID number that identifies the process.

2- **Process state**: new, ready, running, waiting, or terminated.

3- **Program counter**: contains the address of the next instruction to be executed.

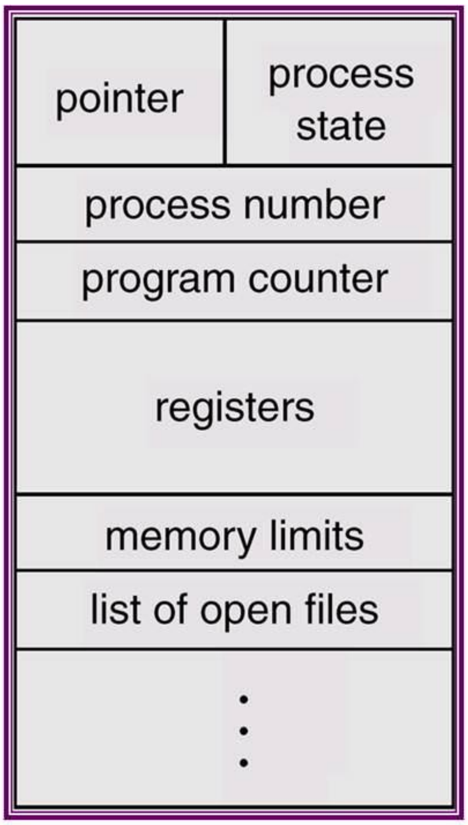
4- **CPU registers**: index registers, stack pointers, and general purpose registers.

5- **CPU scheduling**: process priority, and any other scheduling parameters.

6- **Memory management information**: value of base and limit registers.

7- **Accounting information**: amount of CPU and real time.

8- **I/O status information**: list of I/O devices, list of open files.



4.4 Process Scheduling

4.4.1 Scheduling: is a task by which the operating system decides to introduce new processes into the system.

4.4.2 Scheduling aims:

Maximize → CPU utilization, throughput.

Minimize → Response time, waiting time, and turnaround time.

4.4.3 Scheduling Criteria: (definitions)

a. CPU utilization: the percentage of the time CPU doing useful work to the total elapsed time.

b. Throughput: is the total number of processes that complete their execution per unit of work.

c. Turnaround time: is the total time between submission of a process and its completion.

d. Waiting time: is the time the process remains in the ready queue.

e. Response time: is the time from the submission of a request until the first response is produced.

f. Balance: keep all parts of the system busy.

Processes can be described as :

1- **I/O-bound process**: spends more time doing I/O than computations.

2- **CPU-bound process**: spends more time doing computations.

4.5 Scheduling Levels

There are three levels (terms) of scheduling:(there are three types of schedulers)

4.5.1 **Long -Term Scheduler**: (or **job scheduler**) selects which processes should be brought from secondary storage devices (e.g. disk) into memory for execution.

L.T.S control the degree of multi programming.

L.T.S select a good process mix of I/O-bound process and CPU-bound process.

4.5.2 **Short-Term Scheduler**: (or **CPU scheduler**) selects which ready processes should be executed next and allocates CPU to it.

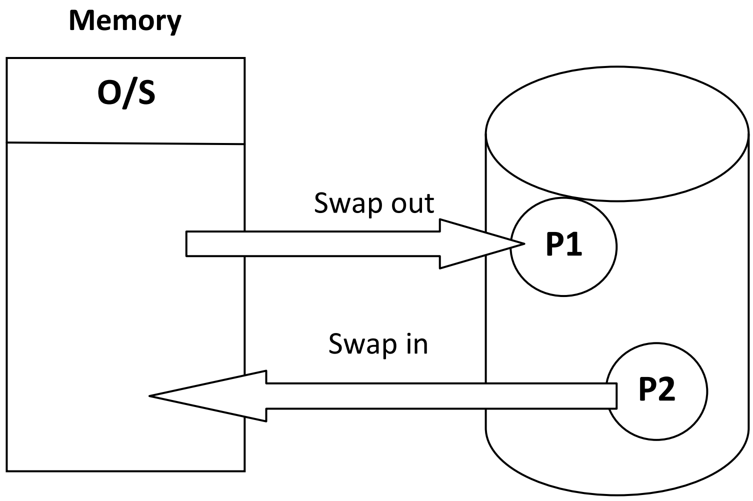
S.T.S select a good process mix of I/O-bound process and CPU-bound process.

4.5.3 **Medium Term Scheduler**: it removes processes from the memory, it reduces the degree of multiprogramming.

4.6 **Context Switch**: Switching the CPU to another process by saving the state of the old process and loading the saved state for the new process.

Disadvantages: **Context-switch time is Pure overhead** , because the system does no useful work while switching.

4.6.1 Swapping: removing a process from memory for some reason and later it can be reloaded into memory.



A process can be swapped out of memory to a backing store and then brought back into memory for continued execution.

Chapter 4 Questions

Q1: Fill in the blanks the following statements with **MAXIMIZE** or **MINIMIZE**:

1-The objective of the O/S is to ---------------------------- response time.
2- The objective of the O/S is to --------------------------- waiting time.
3 The objective of the O/S is to --------------------------- throughput.
4- the objective of the O/S is to ---------------------------- turnaround time.
5- the objective of multi programming is to ------------- CPU utilization.

Chapter 5

5. Interrupt Processing

5.1 Interrupt: is a signal to the processor indicating that an event was occurred and needs immediate attention.

→ The interrupt is generated by the H/W of O/S.

→ Each **interrupt** has its own **interrupt service routine**.

(interrupt handler routine)

→ An operating system is interrupt driven.

5.2 Interrupt vector: contains the addresses of all service routines.

5.3 There are two types of interrupts:

5.3.1 Hardware generated interrupt.

Examples: 1. Timer.
2. Pressing a key on the keyboard.
3. Moving the mouse.

5.3.2 Software generated interrupt (trap or exception).

Examples: 1. Division by zero.
2. System calls.
3. Access to a bad memory address.

5.4 Interrupt Classes (types)

There are six interrupt classes:

5.4.1 SVC (Supervisor call) interrupts

- I/O request.
- obtaining more storage.

5.4.2 I/O interrupts

- I/O operation completes.
- I/O error.

5.4.3 External interrupts

5.4.4 Restart interrupts

- pressing the console restart bottom.

5.4.5 Program Check Interrupts

- Divide by zero.
- Arithmetic overflow.

5.4.6 Machine check interrupts

5.5 Preemptive and non Preemptive Scheduling

- **Preemptive scheduling**: the CPU can be taken away from the process.

- **Non preemptive scheduling**: the CPU cannot be taken away from the process.

5.6 Scheduling Algorithms

5.6.1 First- Come, First- Served (FCFS)

The process that requests the CPU first is allocated the CPU first.

Advantages: Simple and easy to understand.

Disadvantages: 1- The average waiting time is quite long.
2- Not useful for time-sharing system.

- FCFS is not optimal because the average waiting time is quite long.

- FCFS is not useful for time-sharing system because once the CPU is allocated to a process, that process keeps the CPU until it releases the CPU.

NOTE: **FCFS** is **Non-Preemptive** scheduling Algorithm.

Example 1: consider the following set of processes that arrive at time 0, with the length of the CPU burst time given in milliseconds:

Process	Burst time
P1	24
P2	3
P3	3

1. Draw Gantt Chart.
2. Find average waiting time.
3. Find average turnaround time.

The Gantt Chart is:

P1	P2	P3

0 24 27 30

Waiting time = Start of execution time – arrival time.

Waiting time for P1 = (0 – 0) = 0 ms
Waiting time for P2 = (24 – 0) = 24 ms
Waiting time for P3 = (27 – 0) = 27 ms
Average waiting time: (0 + 24 + 27) / 3 = 17 ms

Turnaround time = waiting time + burst time.

Turnaround time for P1 = 0 + 24 = 24 ms

Turnaround time for P1 = 24 + 3 = 27 ms

Turnaround time for P1 = 27 + 3 = 30 ms

Average turnaround time = (24 + 27 + 30)/3 = 27 ms

Example 2: Consider the following snapshot (table):

Process	Arrival time	Execution time
P0	0	5
P1	1	3
P2	2	8
P3	3	6

The Gantt Chart:

P0	P1	P2	P3

0 5 8 16 22

Waiting time for P0 = (0 – 0) = 0 ms

Waiting time for P1 = (5 – 1) = 4 ms

Waiting time for P2 = (8 – 2) = 6 ms

Waiting time for P3 = (16 – 3) = 13 ms

Average waiting time = (0 + 4 + 6 + 13)/4 = 5.55 ms

Example 3: Suppose 4 processes P1, P2, P3, P4 arrive to C/S at time (1, 2, 3, 4) ms respectively, their execution time (burst time) are (5, 3, 8, 6) ms respectively. Use FCFS scheduling, draw Gantt Chart and find average waiting time.

Gantt Chart:

idle	P1	P2	P3	P4

0 1 6 9 17 23

Waiting time for P1 = (1 – 1) = 0 ms

Waiting time for P2 = (6 – 2) = 4 ms

Waiting time for P3 = (9 – 3) = 6 ms

Waiting time for P4 = (17 – 4) = 13 ms

Average waiting time = (0 + 4 + 6 + 13)/4 = 5.55 ms

Example 4: Suppose 4 processes P1, P2, P3, P4 arrive to C/S at time (6, 8, 5, 2) ms respectively, their execution time (burst time) are (7, 5, 6, 2) ms respectively. Use FCFS scheduling, draw Gantt Chart and find average waiting time.

Gantt Chart:

Idle	idle	P4	idle	P3	P1	P2

0 1 2 4 5 11 18 23

Waiting time for P4 = (2 – 2) = 0 ms
Waiting time for P3 = (5 – 5) = 0 ms
Waiting time for P1 = (11 – 6) = 5 ms
Waiting time for P2 = (18 – 8) = 10 ms
Average waiting time = (0 + 0 + 5 + 10)/4 = 3.75 ms