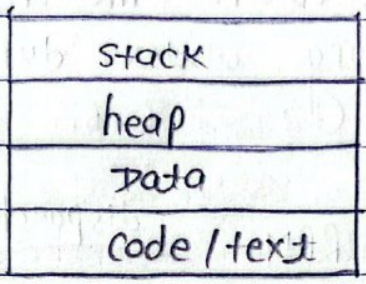
UNIT: 2

Process Management.

18 Marks

DATE : __/__/__
PAGE : _____

Process means program in execution. OS can allocate required memory space for program execution. The process have a predefined structure as shown below



In this process memory space is divided into four segments

(i) **code / text segment.**

It used to store actual program. It is fixed size segment. It size cannot be change when the process executed.

(ii) **Data segment.**

It is used to store data values related to the program like variable initialization etc.

(iii) **Heap segment.**

It allocates memory which may be processed during its run time.

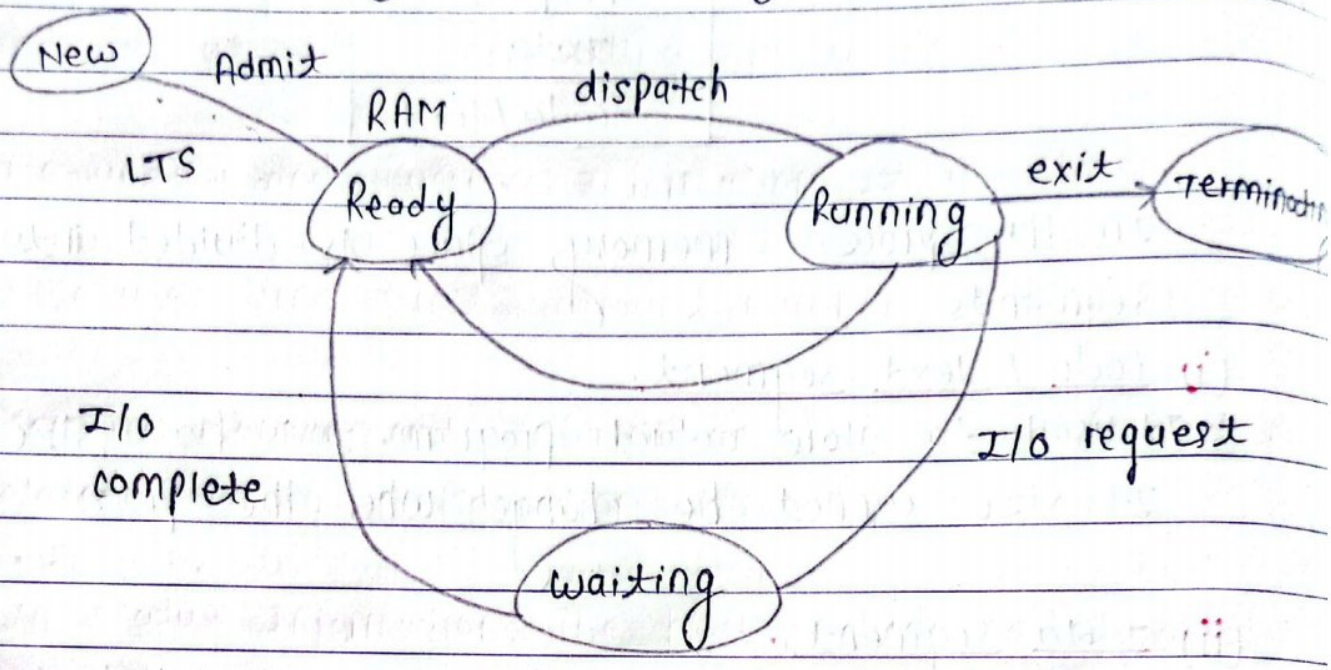
(iv) **Stack segment.**

The stack store temporary data like function parameters, return adders and local variables.

Imp

Process state:

A process may change it's state during process execution. Process state describes position of the process at particular time. A process possible to placed in following states during it's execution.



① New: The process that is just being created. The program counter block (PCB) is already being made but the program is not at yet loaded in main memory (RAM). The program remains in the new state until the long term scheduler (LTS) moves the process to the ready state (main memory).

② Ready: A process that is waiting to be executed.

③ Running: The process that is currently being executed.

④ Waiting: A process that cannot execute until some event occurs, such as the completion of an I/O operation.

DATE : __/__/__
PAGE : _____

⑤ Terminated : The process that is finished due to some reasons.

Process Control Block (PCB).

A process control block (PCB) is a data structure C that is created and manage by the OS for every process. Every process or programs that runs need a PCB. When a user requires to run a particular program, the OS construct a corresponding process control Block for that program. The process control Block is the key tool that enable the OS to support multiple process and to provide multi-processing.

Since a process is uniquely characterized by the following elements, a control process control Block contains all these elements.

process state
program counter
CPU Register
CPU scheduling information.
Accounting and business information.
Memory management information.
I/O status information.

⇒ Process state: A process can be new, ready, running, waiting, terminating

⇒ Program Counter: The program counter lets you know the address of the next instruction, which should be executed for the process.

DATE : __/__/__
PAGE : _____

→ CPU register : This component include accumulators, index and general purpose registers and information of condition code.

→ CPU scheduling information : This component include a process priority, pointers for scheduling queue and various other scheduling parameters.

→ Accounting and Business information: It include the amount of CPU and time utilities a like real time use, time limit, account numbers, job or process etc No.

→ Memory management information : Include information related memory configuration for a process

→ I/O status information: The block include a list of open files, the list of I/O device that are allocated to the process.

Threads

A process is divided into smaller task and each task is called a Thread. A thread (sometimes called a lightweight) is a single sequential execution stream within a process. It is not a itself a self program because it cannot run itis own. A thread has itis own registers, program counters, stack. A thread share address space, program code counter program code, global variables, OS resource (files, I/O devices) with other threads

The process can be split down into so many threads. for eg, In a browser, many taps

Tabs can be viewed as thread.

→ Multi-threading

→ A process is divided into number of smaller task, each task is called a thread. Number of threads within a process execute at a time is called multi-threading

→ If a program is multi-threaded even when some portion of, it is blocked the whole program is not blocked. The rest of the program continue working if multiple CPU's are available.

→ Multi-threading gives best performance. If we have only a single thread, no. of CPU's available, no performance benefits acheived.

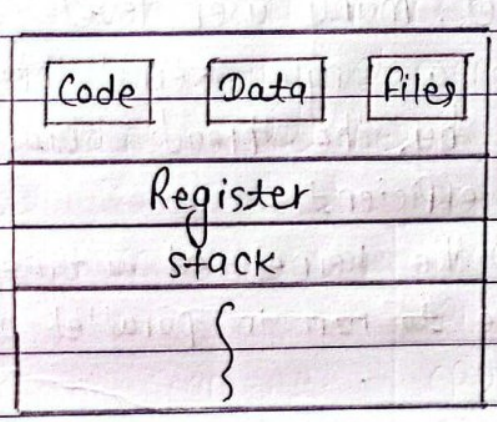


Fig single threaded process

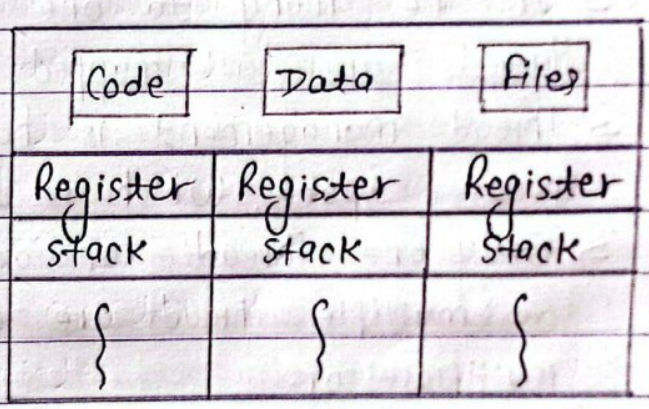


Fig mutiple threaded process.

Types of Thread

(1) User level threads.

→ User-level threads are managed by user level library.

→ User-level threads are typical fast.

→ Contexts switching fast.

→ If one user level threads perform blocking operation then entire process get blocked

DATE : __/__/____
PAGE : _____

② Kernel-level threads.

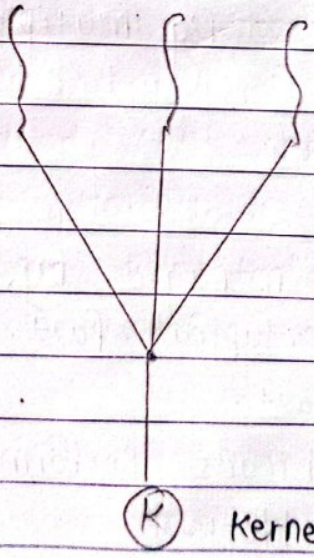
⇒ Kernel-level threads are managed by OS.
⇒ Kernel-level are slower than user-level threads.
⇒ Context-switching is slow.
⇒ If one Kernel-level threads blocked, no offect on other.

Multithreading models.

① Many to one relationship
② One to one relationship.
③ Many to Many relationship.

① Many to one relationship.

⇒ In the many to one model, many user level threads are all mapped onto a single kernel thread.
⇒ Thread management is handle by the thread library in user space, which is very efficient.
⇒ Only one thread can access the kernel at a time, so multiple threads are unable to run in parallel on multiprocessor.

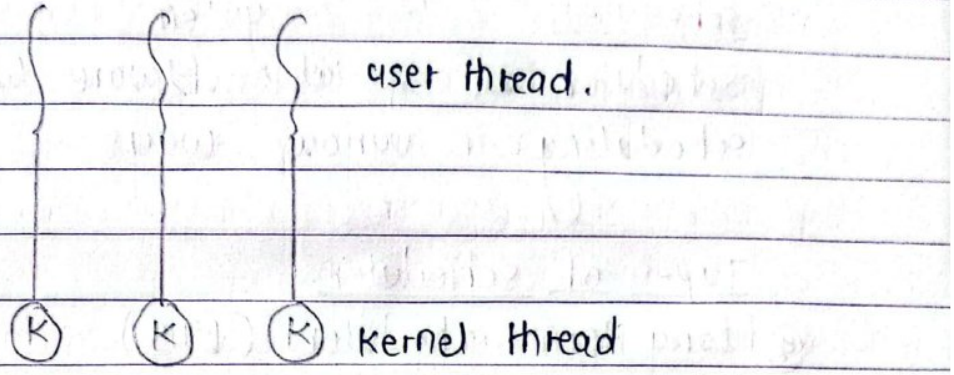


DATE : __/__/__
PAGE : _____

② One to one relationship.

⇒ It support multiple threads to execute in parallel on up.

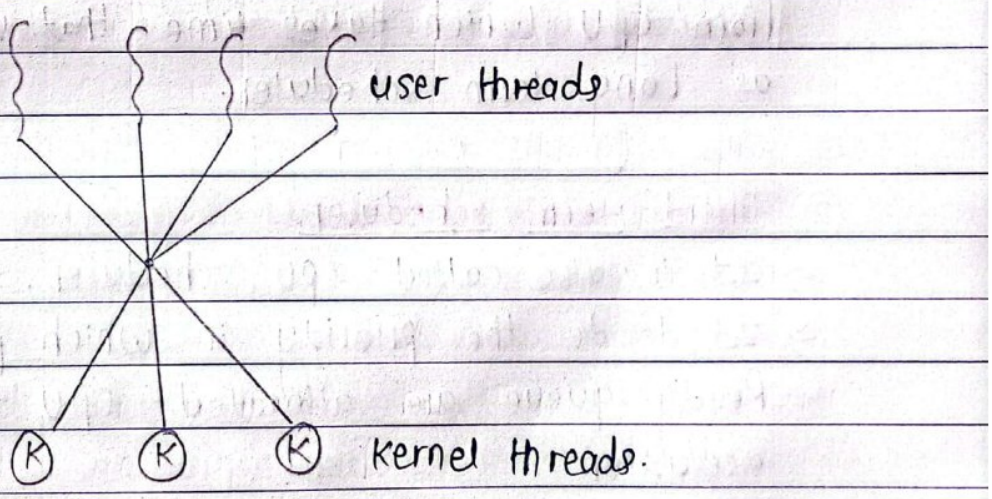
⇒ One to one model overcomes the problem listed above involving blocking system calls and the splitting a processes across multiple cpus.



③ Many to many relationship

⇒ Many to many model, many user level threads, multiplexed to the kernel threads of small or equal numbers. The no. of kernel thread may be specific to either a particular application or a particular machine.

⇒ User have no restriction on the no. of threads created.



Scheduling

Process scheduling in the OS that schedules processes in various states such as Ready, waiting and Running process scheduling allows the OS to assign each

process a time interval for CPU execution.

⇒ The prime aim of the process scheduling system is to keep the CPU busy all the time and to deliver minimum response time for all programs.

Scheduler

Scheduler are special system software which handle process scheduling in various ways

Types of scheduler.

1. Long term scheduler (LTS)
2. Short term scheduler (STS)
3. Medium term scheduler (MTS)

① Long term scheduler

process of long term scheduler are placed in the ready state because in this state the process is ready to execute waiting for calls for execution from CPU which takes time thatswhy this is known as Long term scheduler.

② Short term scheduler.

⇒ It is also called CPU scheduler.

⇒ It decide the priority in which processes is in the Ready queue are allocated CPU time for their execution.

③ Medium term scheduler.

⇒ It is also called swap scheduler.

⇒ The task of moving from main memory to secondary memory is called swapping out. The task of moving

a swapedout process from secondary memory to main memory is known as swapping in.

→ The swapping of processes is perform to ensure the best utilization of main memory.

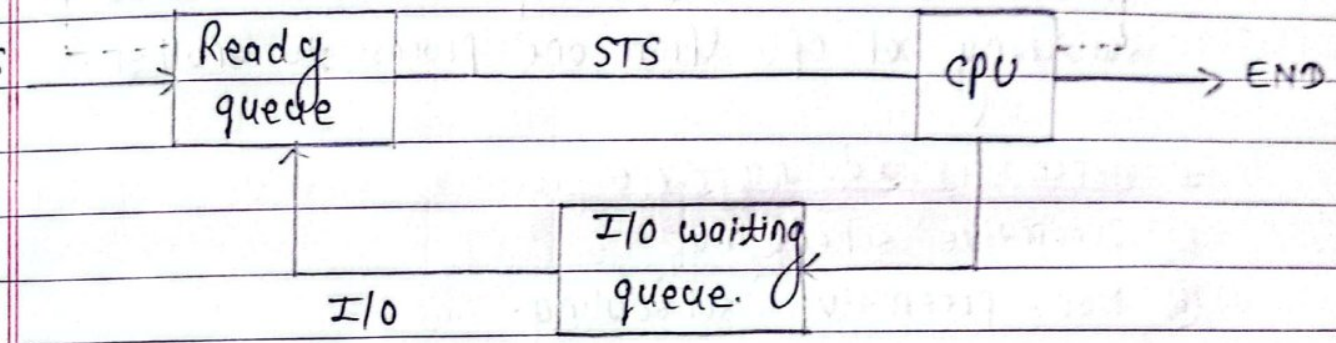


Fig: Types of scheduler.

Scheduling Criteria.

① Arrival Time.
A type at which the process arrives in the Ready queue

② Completion time.
Time at which process completes its execution.

③ Burst Time
Time required by a process for CPU execution. This doesn't include waiting time.

④ Turn Around Time (TAT)
Time difference betⁿ completion time and Arrival time.
OR,
Time Around Time = Completion Time - Arrival Time.

⑤ Watting Time (WT)
Timing difference betⁿ Turn Around Time and Burst

Time.

OR,

Waiting time = Turn around time - Burst time.

⑥ Context ~~Time~~ switching

A context switching is a process that involves switching of CPU from one process to another.

Process scheduling types.

ⓐ Preemtive scheduling

ⓑ Non-preemtive scheduling.

ⓐ Preemptive scheduling.

The scheduling in which a running process can be interrupted if a high priority process enters the queue and is allocated to the CPU is called preemtive scheduling. In this case, current process switches from the running queue to Ready queue and the high priority process utilize the CPU cycle.

ⓑ Non-preemptive scheduling.

→ The scheduling in which a running process connot be interrupted by any other process is called Non-preemtive scheduling.

→ Any other process which enters the queue has to wait until the current process finish its CPU cycle.

→ When a Non-preemtive process with a high CPU brust time is running, the other process would have to wait for a long time, and that increased the process average waiting time in the Ready queue.

DATE : __/__/__
PAGE :

Scheduling algorithms.

The various scheduling algorithm are as follows:

① Scheduling in batch system
- → First come first served (FCFS) scheduling
- → shortest Job first (SJF) scheduling
- → shortest Remaining Time first (SRTF)

② Scheduling in Interactive system.
- → Priority scheduling.
- → Round Robin (RR) scheduling.
- → Highest Response ratio next (HRRN) scheduling.

First come first served (FCFS) scheduling.

- → One of the simplest scheduling algorithm.
- → As the name implies, the processes are executed in the order of arrival in the ready queue which means the process that enters the ready queue first gets CPU first.
- → It is non preemitive algorithm. Therefore once a process gets the CPU, it retains control until it block or terminate.
- → Implementation of Ready queue is manage as FIFO queue.

Advantage

It is simple and Easy to understand.

Disadvantage

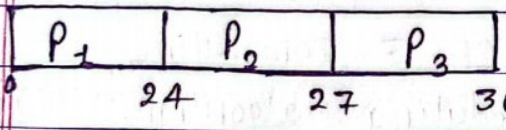
If processes with higher brast time arrived before processes with smaller ~~brust~~ burst time then smaller processes have to wait for a long time for long processes to release CPU.

DATE : __/__/__
PAGE : _____

Consider the following set of processes that arrives at time 0 with the length of the CPU burst time given in milisecond.

process	Burst time
P_1	24
P_2	3
P_3	3

Gantt chart : suppose processes arrives in order P_1, P_2 and P_3



process	Burst time (BT)	Arrival time (T_0)	Finish time (T_f)	TAT	WT	RT
P_1	24	0	24	24	0	0
P_2	3	0	27	27	24	24
P_3	3	0	30	30	27	27

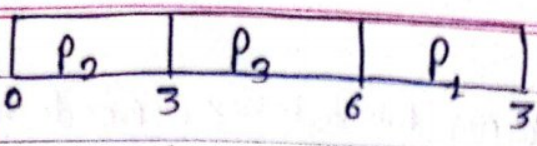
Now

Average turn around time (ATAT) = $\left(\frac{24+27+30}{3}\right)$ = 27

Average waiting time (AWT) = $\frac{(0+24+27)}{3}$ = 17 ms

suppose processes arrives in order P_2, P_3 P_1

Process	Burst time
P_2	3
P_3	3
P_1	24



process	Burst time ΔT	Arrival time (T_0)	Finish time (T_1)	TAT	WT	RT
P_2	3	0	3	3	0	0
P_3	3	0	6	6	3	3
P_1	24	0	30	30	6	6

Now

Average turn around time (ATAT) $= \left(\frac{3+6+30}{3}\right) = 13$ ms

Average wasting time (AWT) $= \left(\frac{0+6+3}{3}\right) = 3$ ms

Shortest Job first (SJF) scheduling.

⇒ Also called shortest process next (SPN) or shortest requires next (SRN).

⇒ It is a scheduling algorithm that schedules the processes according to the length of CPU burst they required.

⇒ When the CPU is available, it is assigned to the process that has smallest next CPU burst.

⇒ If the next burst of two processes are the same, first come first serve is used

⇒ Shortest Job first scheduling can be either preemitive or non-preemitive.

Advantage

⇒ Shortest Job are favored.

⇒ It gives minimum average waiting time for a given set of processes.

DATE : __/__/__
PAGE : _____

Disadvantage

⇒ It is difficult to implement as it needs to know the length of CPU burst a processes in advanced.

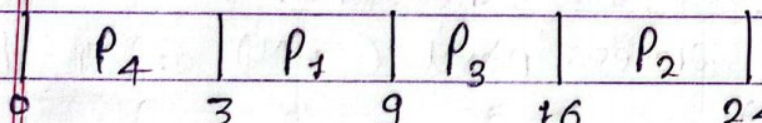
example

Consider the following set of processes that arrives at time 0 with the length of CPU burst in milisecond.

process	Burst time	Arrival time
P_1	6	0
P_2	8	0
P_3	7	0
P_4	3	0

soln

Gantt chart



Process	Burst time	Arrival time	Final time	TAT	WT	RT
P_1	6	0	9	9	3	3
P_2	8	0	24	24	16	16
P_3	7	0	16	16	9	9
P_4	3	0	3	3	0	0

Average turn around time (ATAT) $= \frac{9+24+16+3}{4} = 13$ ms

Average waiting time (AWT) $= \frac{3+16+9+0}{4} = 7$ ms.

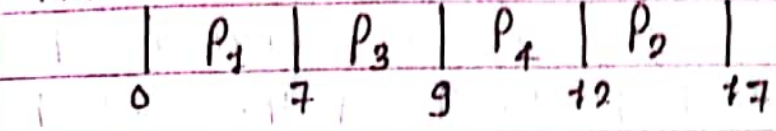
Response time = cpu time - arrival time

DATE / /
PAGE

Q.

process	Burst time	AT
P_1	7	0
P_2	5	1
P_3	2	3
P_4	3	4

Gantt chart



process	Burst time	Arrival time	final time	TAT	WT	RT
P_1	7	0	7	7	0	0
P_2	5	1	17	16	11	11
P_3	2	3	9	6	4	4
P_4	3	4	12	8	5	5

Now,

$$ATAT = \left(\frac{7+16+6+8}{4}\right) = 9.25 ms$$

$$AWT = \left(\frac{0+11+4+5}{4}\right) = 5 ms.$$

Shortest remaining time first (SRTF). (Preemitive)

Shortest remaining time first (SRTF) is a preemitive version of shortest job first (SJF). In this case, the scheduler always choose the process that has shortest remaining processing time. when a new process joins the ready queue, the scheduler compares the remaining time of executing process and new process. If the new process has the least cpu burst time, the scheduler select that job and allocate cpu, otherwise it continue with the old process.

Advantage

⇒ SRTF does not have the bias in favour of long process found in FCFS.

Disadvantage

⇒ There is a risk of starvation of longer process.

Q. Consider the following set of processes that arrives at given time, with the length of the CPU burst given in milisecond.

(a) Calculate AWT and ATAT.

process	Arrival time	Burst time
P_1	0	7
P_2	2	4
P_3	4	1
P_4	5	4

Soln

Gantt chart:

P_1	P_1	P_2	P_2	P_3	P_2	P_4	P_1
0	1	2	3	4	5	7	11	16

OR

P_1	P_2	P_3	P_2	P_4	P_1
0	2	4	5	7	11	16

Process	Arrival time	Burst time	final time	TAT	WT	RT
P_1	0	7	16	16	9	0
P_2	2	4	7	5	1	0
P_3	4	1	5	1	0	0
P_4	5	4	11	6	2	2

$$\therefore ATAT = \left(\frac{16+5+1+6}{4}\right) = 7 \text{ ms}$$

$$AWT = \left(\frac{9+1+0+2}{4}\right) = 3 \text{ ms}$$

DATE : __/__/__
PAGE : _____

Priority scheduling

① priority scheduling can be either preemptive or non-preemptive.

② A preemptive priority scheduling algorithm will preempt the CPU if the priority of newly arrived process is higher than the priority of the currently running process.

③ Non-preemptive scheduling algorithm will simply put the new process at the head of ready queue.

④ Equal priorities are scheduled in FCFS order.

Advantage

process are executed on the basis of priority so higher priority doesnot need to wait for long which save time.

Disadvantage.

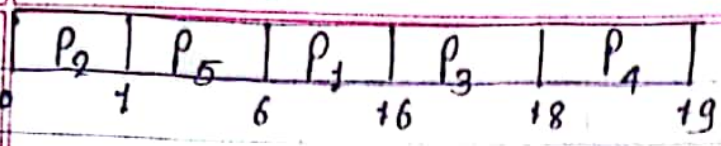
If high priority process use of a lot of CPU time, lower priority process may ~~stop~~ starve and be postponed indefinetely.

for Eg, Let us consider following 5 process with arrival time zero (0) ms and length of the CPU ~~burst~~ burst given in milisecond consider one is highest priority. Calculate, Average waiting time (AWT)
Average turn around time (ATAT)

process	Burst time	priority	
P_1	10	3	[Non preemptive way]
P_2	1	1	
P_3	2	4	
P_4	1	5	
P_5	5	2	

DATE : __/__/__
PAGE : _____

Gantt chart :



process	Burst time	finish time	Arrival time	TAT	WT
P_1	10	16	0	16	6
P_2	1	1	0	1	0
P_3	2	18	0	18	16
P_4	1	19	0	19	18
P_5	5	6	0	6	1

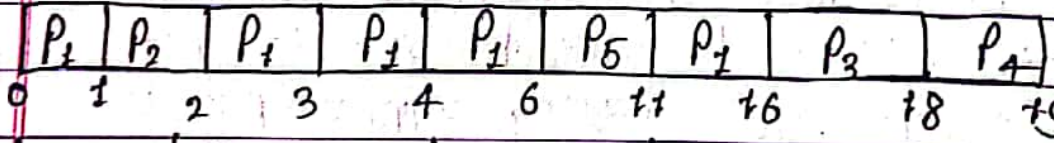
ATAT = (16+1+18+19+6)/5 = 12 ms

AWT = (6+0+16+18+1)/5 = 8.2 ms

[preemptive way]

Process	Burst time	priority	Arrival time
P_1	10	3	0
P_2	1	1	1
P_3	2	4	2
P_4	1	5	4
P_5	5	2	6

Gantt chart:



process	Burst time	priority	Arrival time	finish time	ATAT	W
P_1	10	3	0	16	16	6
P_2	1	1	1	2	1	0
P_3	2	4	2	18	16	14
P_4	1	5	4	19	15	14
P_5	5	2	6	11	5	0

Now.

$$ATAT = \frac{16+1+16+15+5}{5} = 10.6 \text{ ms}$$

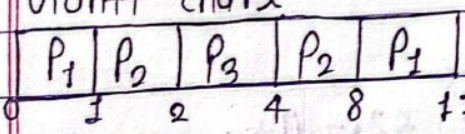
$$AWT = \frac{6+0+14+14+0}{5} = 6.8 \text{ ms}$$

Consider a set of three processes P_1, P_2, P_3 with their priorities and arrival time as given below. calculate AWT for preemptive priority based algorithm.

process	Burst time	priority	Arrival time
P_1	10	3	0
P_2	5	2	1
P_3	2	1	2

where '1' denotes highest priority and '3' denotes lowest priority.

Gantt chart



process	Burst time	Arrival time	finish time	ATAT	WT
P_1	10	0	17	17	7
P_2	5	1	8	7	2
P_3	2	2	4	2	0

Now

$$AWT = \frac{7+2+0}{3} = 3 \text{ ms}$$

Consider the set of processes with arrival time (in ms), CPU burst time (in ms) and priority 0 is the highest priority using preemptive scheduling algorithms.

Process	AT	Burst time	priority
P_1	0	11	2
P_2	5	28	0
P_3	12	2	3
P_4	2	10	1
P_5	9	16	4

Gantt chart

P_1	P_4	P_2	P_4	P_1	P_3	P_5

0 2 5 33 40 49 51 67

process	AT	Burst time	finish time	TAT	WT
P_1	0	11	49	49	38
P_2	5	28	33	28	0
P_3	12	2	51	39	37
P_4	2	10	40	38	28
P_5	9	16	67	58	42

$$ATAT = \frac{49+28+39+38+58}{5} = 42.4 \text{ ms.}$$

$$AWT = \frac{38+0+37+28+42}{5} = 29 \text{ ms.}$$

Round Robin scheduling.

⇒ Most widely used preemptive scheduling algorithm.
⇒ Specially design for time sharing system.
⇒ It considers all processes equally important and treates in an impartial manner.
⇒ Each process in the ready queue gets a fixed amount of cpu time. (generally from 10-100ms) known as time slice or time quantum for its execution)

DATE : __/__/__
PAGE : _____

⇒ CPU is assigned to the process on the basis of FCFS for a fixed amount of time called quantum time. After the time quantum expire the running process is ~~printed~~ preempted and send to the ready queue and then processor is assigned to the next arrive process. This process continous in circular fashion.

Advantages.

⇒ Every process gets an equal share of the CPU
⇒ Round Robin is cyclic in nature, so there is no starvation.

Disadvantages.

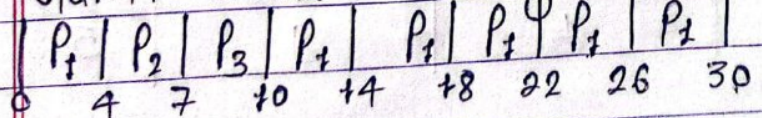
⇒ This methods spends more time on contacts switching.
⇒ Average waiting time is not minimal.

Q. Let us consider following three processes with arrival time 0ms and length of the CPU burst given in ms consider quantum time (QT) = 4ms find

(a) AWT
(b) ATAT

process	Burst time
P_1	24
P_2	3
P_3	3

Gantt chart (running state)



process	Burst time	finish time	WT	TAT
P_1	24	30	6	30
P_2	3	7	4	7
P_3	3	10	7	10

$$\therefore AWT = \frac{6+4+7}{3} = \frac{17}{3} = 5.66$$

$$ATAT = \frac{30+7+10}{3} = 47/3 = 15.66$$

Q.

process	Arrival time	Burst time
P_1	0	5
P_2	1	4
P_3	2	2
P_4	4	1

Quantum time = 2

Ready queue

$|P_1|P_2|P_3|P_1|P_4|P_2|P_1|$

Running queue

$|P_1|P_2|\Phi|P_3|P_1|P_4|P_2|P_1|$

0 2 4 6 8 9 11 12

process	Arrival time	Burst time	finish time	TAT	WT
P_1	0	5	12	12	7
P_2	1	4	11	10	6
P_3	2	2	6	4	2
P_4	4	1	9	5	4

Now,

$$ATAT = \frac{12+10+4+5}{4} = 7.75$$

$$AWT = \frac{7+6+2+4}{4} = 4.75$$

DATE : __/__/__
PAGE :

Highest Response ratio next (HRRN) scheduling:

⇒ Non-preemptive scheduling algorithm that schedules the process according to their response ratio.

⇒ Whenever the CPU becomes available, the process having the highest value of response ratio among all the ready processes is scheduled next.

⇒ Jobs gain higher priority the longer they wait, which prevent indefinite postponment (process starvation)

⇒ The response ratio of a process in the queue is computed by using the following eqn.

$$\text{Response Ratio (RR)} = \frac{\text{time since arrived} + \text{CPU burst time}}{\text{CPU burst time}}$$

Advantage

Improves upon SJF scheduling

⇒ Consider how long process has been waiting

Disadvantage

Doesnot support external priority system process are scheduled by using internal priority system.

Q. Let us consider following three processes with given arrival and length of the CPU burst is given in ms calculate

(a) AWT

ATAT

process	Arrival time	Burst time	priority
P_1	0	7	3
P_2	2	4	1
P_3	3	4	2

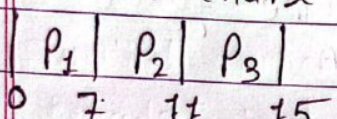
DATE : / /
PAGE :

RR = (WT + BT) / B.T

RR for $P_2 = \frac{(7-2)+4}{4} = 2.25$

RR for $P_3 = \frac{(7-3)+4}{4} = 2$

Gantt chart



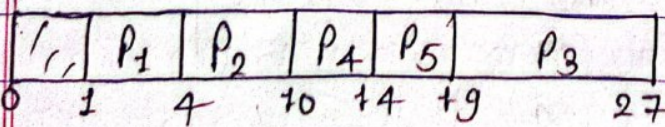
Process	Arrival time	Burst time	finish time	TAT	WT
P_1	0	7	7	7	0
P_2	2	4	11	9	5
P_3	3	4	15	12	8

ATAT = $\frac{7+9+12}{3} = 9.33$

AWT = $\frac{0+5+8}{3} = 4.33$

Q.

Process	Arrival time	Burst time
P_1	1	3
P_2	3	6
P_3	5	8
P_4	7	4
P_5	8	5



DATE : __/__/__
PAGE : _____

Response ratio for P_3 $= \frac{(10-5)+8}{8} = 1.625$

RR for $P_4 = \frac{(10-7)+4}{4} = 1.75$

RR for $P_5 = \frac{(10-8)+5}{5} = 1.40$

again

RR for $P_3 = \frac{(14-5)+8}{8} = 2.125$

RR for $P_5 = \frac{(14-8)+5}{5} = 2.2$

process	Arrival time	burst time	finish time	TAT	WT
P_1	1	3	4	3	0
P_2	3	6	10	7	1
P_3	5	8	27	22	14
P_4	7	4	14	7	3
P_5	8	5	19	11	6

Now,

ATAT $= \frac{3+7+22+7+11}{5} = 10$

AWT $= \frac{0+1+14+3+6}{5} = 4.8$