3 m Operating System Services

OS Services:-

An os provides services both to the programs and to the users. They are classified into two categories.

*) User point of view
*) System point of view

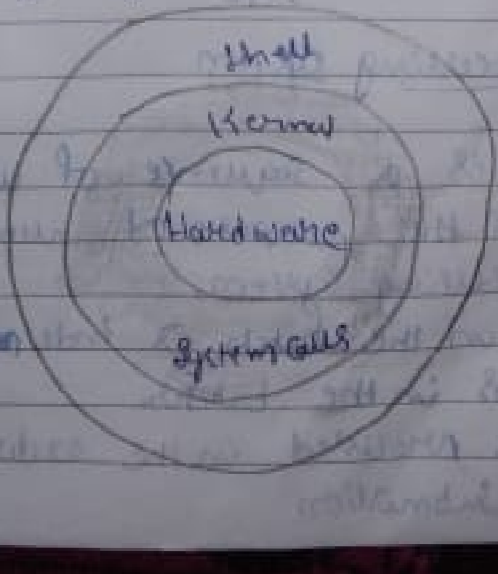
User point of view

*) user interface
*) program execution
*) I/O operation.
*) File system manipulation
*) error detection
*) Communication

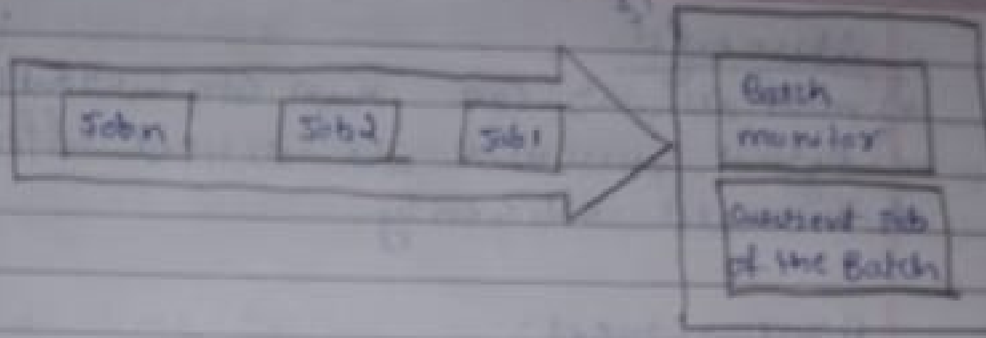
System point of view

*) Resource management
*) protection and security.
*) Accounting.

Operating system Structure



3



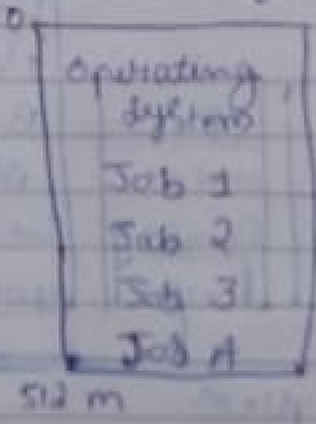
Advantages:-

1) Simple scheduling
2) Simple mechanism for file management

Disadvantages

1) execution time of a job is dependent on batch size
2) Some jobs can enter an infinite loop

3) multi programming system



* multi programming is the rapid switching of CPU between multiple process in memory.
1) multi-programming organises jobs in such way that CPU always has atleast one job to execute.
2) multiple job can run concurrently

2

Operating system is divided into a number of layers, in which the lowest layer represent the hardware part and the highest layer represents the user interface part.

The two important layers of any OS are:-

1) Kernal :- The core of the operating system is the kernal - the operating system program. It controls the computer's resource, allotting them to different users and to different tasks. It interacts directly with the hardware.

2) Shell :- The shell is a utility which acts as the command interpreter for the kernal and is the interface between the user and the kernel.

Type of operating system

1) Batch processing system
2) multi programming system
3) Time sharing system
4) Real time system

1 Batch processing system

1) A batch is a sequence of user jobs formed for the purpose of processing by a batch processing system.
2) Each job in the batch is independent of other jobs in the batch.
3) Jobs are processed in the order of submission

Disadvantages

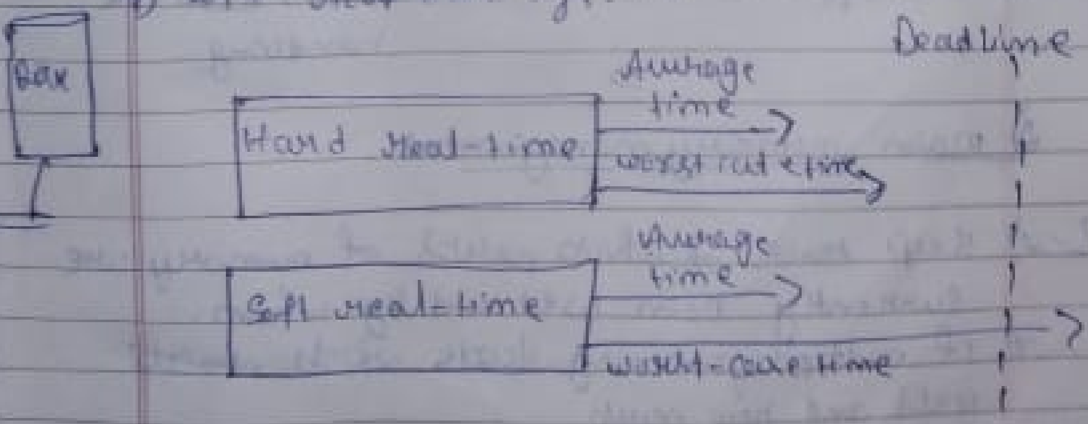
*) more Complex compared to multi programming os
*) Required memory management and protection.

4 Real-time System

*) A real time operating system is used when rigid time requirements have been placed on the operation of a processor or the flow of data.
*) It is often used as a control device in a dedicated application.

There are two types of real time operating system

*) Hard real time system
*) Soft real time system



Advantages

*) memory management in real time systems is comparatively less demanding than in other types of operating system

Explaination

*) First the process is created ["new state"]. It is located from the secondary storage device into a main memory after that the process schedular assigns to the "waiting state".

*) when the process is "waiting" it waits for the schedular to do the context switch and load the process into the processor. The process is said to be in "ready state".

*) when the process becomes "running" and the processor executes the process instruction.

*) if the process needs to look for a resource, It is assigned to the "blocked state". when it is changed back to waiting. the process is no longer needs to waits

=> process control Block:

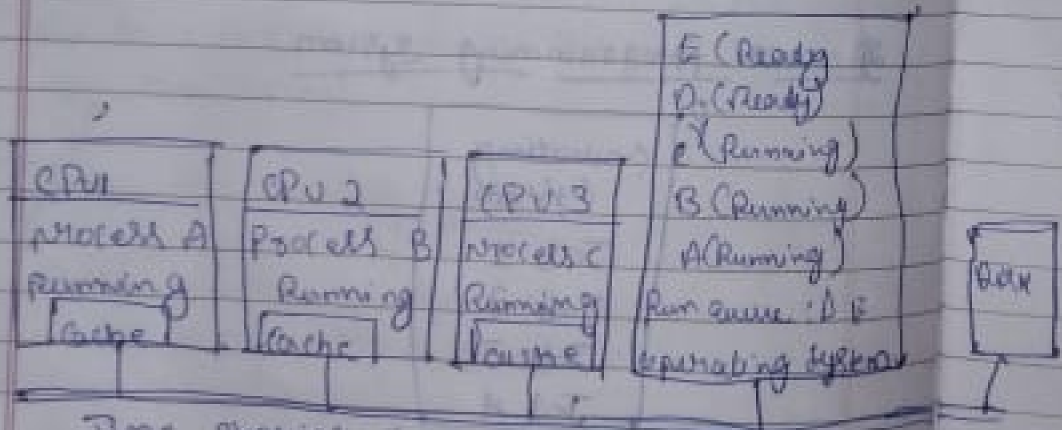
process state
process number
process counter
Register
Memory limits
List of open files
• • •

Advantages:
*) multiple job can run concurrently
*) System resources are utilised effectively
*) Increased efficiency.

Disadvantages:

*) user cannot interact with the job when it is executing
*) programmers cannot modify the programs as it executes

3) Time sharing system



Time sharing system

→ A time shared os allows many users to share the computer simultaneously

Advantages:
*) allow many users to share the computer simultaneously
*) user can interact with the job when it is executing
*) CPU idle time is minimal.

* File management in real-time systems usually increase the speed of access

Disadvantages

* There is a time limit allocated for each event
* Implementation is too costly.

Operating System Components (Functions)

1) Process management
* The creation and deletion of both users and system processes.
* The suspension and resumption for process
* The provision of mechanisms for process synchronization
* The provision of mechanisms for process communication
* The provision of mechanisms for deadlock handling

2) Main memory management.

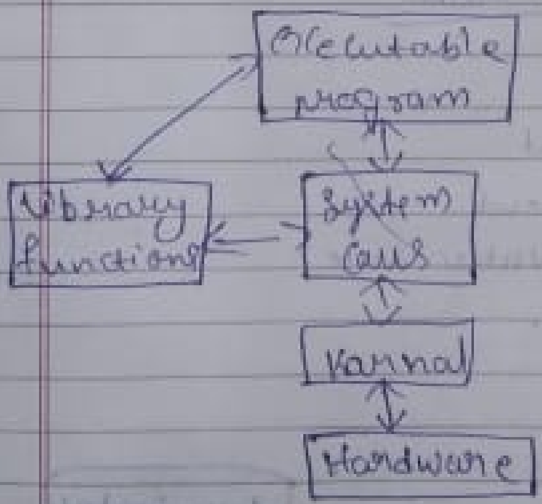
* Keep track of which parts of memory are currently been used and by whom.
* If multiprogramming, decide which process gets and how much
* Allocate and de-allocate memory space as needed.

3. File management

3 *) Creation and deletion of files and directories
*) Support of primitives for manipulating files and directories
*) mapping of files onto secondary storage
*) Backup of files on stable storage media.

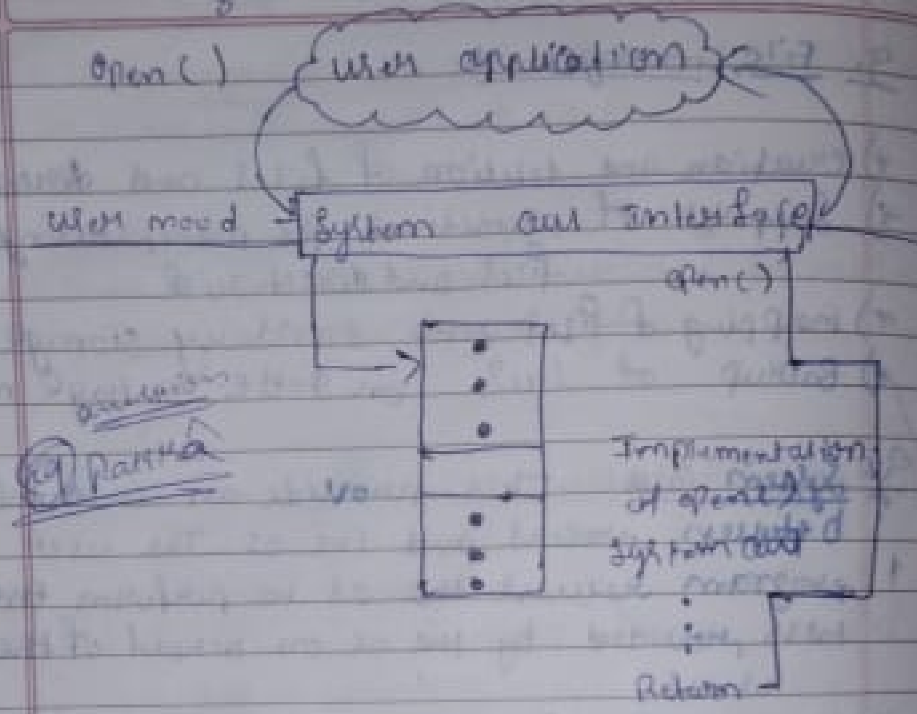
* System call:- This provide an interface between process and the OS. The user
1 program request the OS to perform the task provided by the OS on behalf of the user.

Ex:- process of writting a program to read data from 1 file and copying it to another



System call implementation:

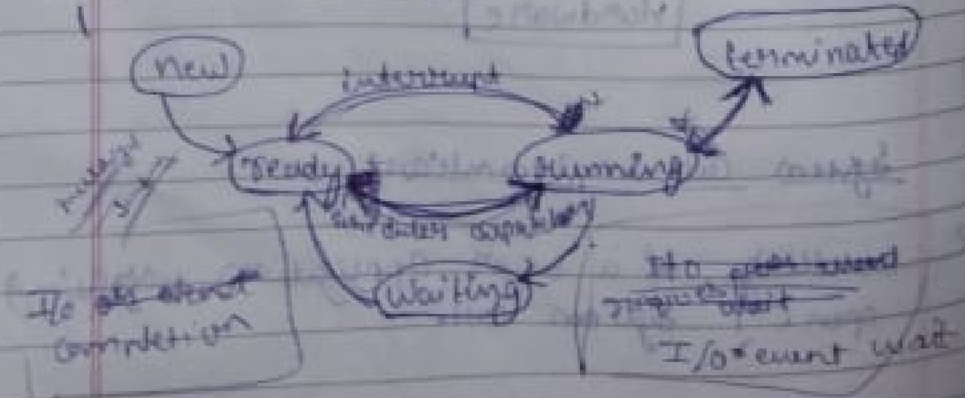
2 Handling of a user application involving open( ) system call.



Types of System Call:

- process control
- file management
- device management
- Information maintenance
- Communication.

States of process



Each process is represented in the operating system by a PCB (process control block)
It is also known as task control block (tcb).

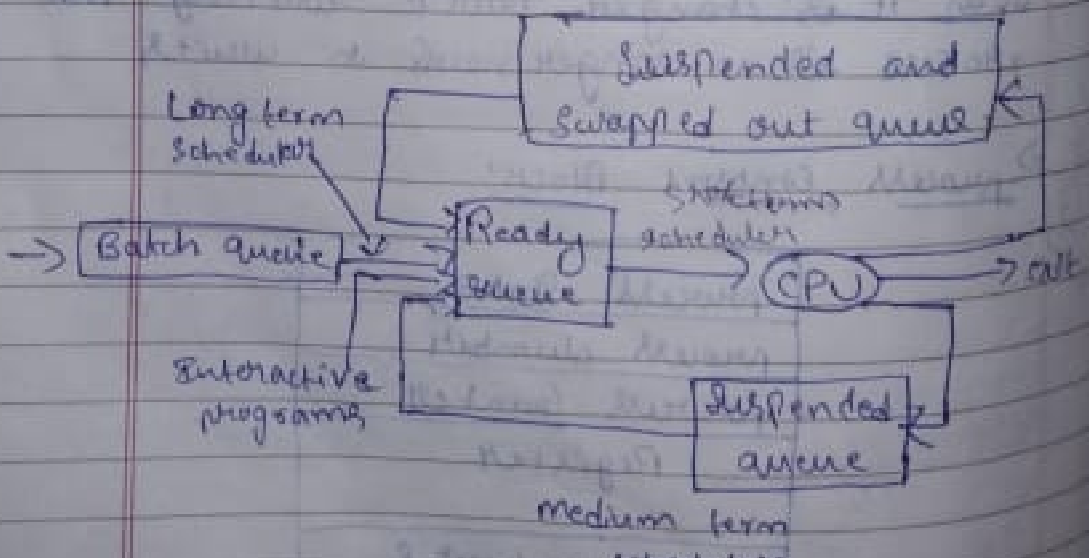
The information stored in the PCB has 3 categories:

1) process identification P.I.P
2) process state information PPPP
3) process control information

Process identification information

PID process Id (PID) or process Number - It is a numeric identifier for the process
PPID parent process Id (PPId) - Identifier of the parent process
UID UID - identifies the user who is responsible for creation of the process.

Schedular:



The selection process is carried out by an appropriate schedular. There are 3 types

* The long term schedular
* the medium term schedular
* The short term schedular

(2m)

1. Context switch :- when an interrupt occurs the system needs to save the current context of the process currently running on the cpu so that it can restore that context when its processing is done. essentially suspending the process and then the resuming it.

2. Dispatcher :- It is a module that gives control to the cpu to the process selected by the short-term scheduler. the function involves.

a) switching involves in context
b) switching to user mode.
c) Jumping to the proper location in the user program to restart that program.

4. Threads (Connectivity)
A thread is a basic unit of cpu utilization it comprises a thread id, a program, a counter, a register set and a stack

There are two types

1) Single threaded process.
2) multi threaded process.

SURYA Gold

Date ____ Page ____

13

V-I

Scheduling Criteria

1) CPU utilization:- It may range from 0% to 100%. In a real system, it will range from 40% (for a lightly loaded system) to 90% for a heavily used system.

2) Throughput:- If the CPU is busy executing process, then work is being done. One measure of work is the number of processes that are completed per time unit call throughput.

3) Turn around time:- For any process, the important criterion is how long it takes to execute that process. The interval from the time of submission of a process to the time of completion is the turn around time.

Turnaround time = waiting time + executing time.

4) Waiting time:- It is the sum of the periods spent waiting in the ready queue.

5) Response time:- Is defined as the time from the submission of a request until the 1st response is produced.

Semaphore

wait | signal

1) Each process must request permission to enter its critical section
2) The section of code implementing this request is entry section
3) The critical section is followed by an exit section
4) The remaining code is remainder section.

Semaphores: A semaphore S is an integer variable in a non-negative values, which can be all used only through 2 standard indivisible (atomic) operations wait() and signal()

wait(): wait is repeated by P. The operation is implemented to acquire a resources.

```
wait (s)
{
    while (s<=0) do no-op;
    s--;
}
```

signal(): when the semaphore is incremented we go to this operation

```
signal(s)
{
    s++
}
```

The wait() and the signal() operations must be executed individually. It increments the value of the semaphore S. It is used to release a resource.

Long term schedular:- The process in batch systems cannot be executed immediately and hence are spooled to the disk and kept for later execution. The long term schedular selects the batch job from the queue.

medium term schedular:- During the execution of jobs priority wise few jobs are be suspended and cannot be executed till it is removed from suspension condition.

Short term schedular:- It allocates the processor among the pool of ready processes residant in memory. It is in charge of ready to running state tranlaction.

*. D/F B/W Non-pre-emptive and pre-emptive.

Non pre-emptive	pre-emptive
*) once the CPU is given to a process it cannot be taken away from that process	*) the CPU can be taken away from a running process.
*) cost is low	*) cost is high
*) Suitable for batch processing	*) Suitable for realtime and interactive timesharing system.
*) scheduling is done once	*) Rescheduling is necessary
*) If an interrupt occurs, the process is terminated	*) when an interrupt occurs, the process is temporarily suspended to resume later.

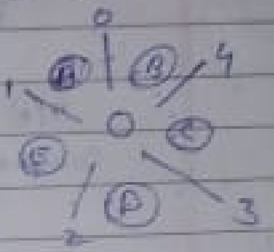
SURYA Gold

Date ______ Page ______

17

The Dining - philosopher problem

Solution to the Dining philosopher problem using semaphore



- Each chopstick should be represented by a semaphore
- Each philosopher picks up the left chop stick first, and then the right chopstick (if right chopstick is used by the neighbour. the particular person has to wait (semaphore).
- After eating, the philosopher execute the signal operation by realising the chopsticks.
- Here we use semaphore chopstick (5)

```
do
{
  wait (chopstick [i]); // left chopstick
  wait (chopstick [(i+1) % 5] // right chopstick
  eat ();
  signal (chopstick[ (i+1) % 5]);
  signal (chopstick [i]);
   think ();
} while (True);
```

if chopstick [0]

- i=0 left
  (0+1 % 5) = 1 right
- i=1 left
  (1+1 % 5) = 2 right
- i=2 left
  (2+1 % 5) = 3 right

continue:-

- i = 3 left
  (3+1 % 5) = 4 right
- i=4 left
  (4+1 % 5) = 0 right

a, b, c, d, ------ n, +14B

1h int a, int b (a) = [11] (b) = [13]

multiprocess Scheduling:
There are 2 types
i) Homogeneous
ii) Hetrogeneous

(2m) Homogeneous:- process are identical in terms of their functionality

Hetrogeneous:- process are of different types
approaches are of different types
- Asymmetric multi processing: master server technique will be used.
- Symmetric multi processing: self scheduling.

V. Imp ★ Critical Section problem:
(Consider a system of n process (P0, P1, .. Pn)
Each process has a segment of code called a critical section.) In which the process may be changing common variables, updating a record in a table, writing a file and so on.
ii) when one process is executing in its critical section no other process is to be allowed to execute in its critical section.

do
{
[entry section] → (controls the entry in critical section and gets a lock a required resources).
critical section → [The critical part]
[exit section] → (Remove the lock from the resources & let others know that its critical section is over)
remainder section → [rest of the section]
} while (true);

* Banker algorithms

Resource allocation graph

Nodes: process ○ Resource □

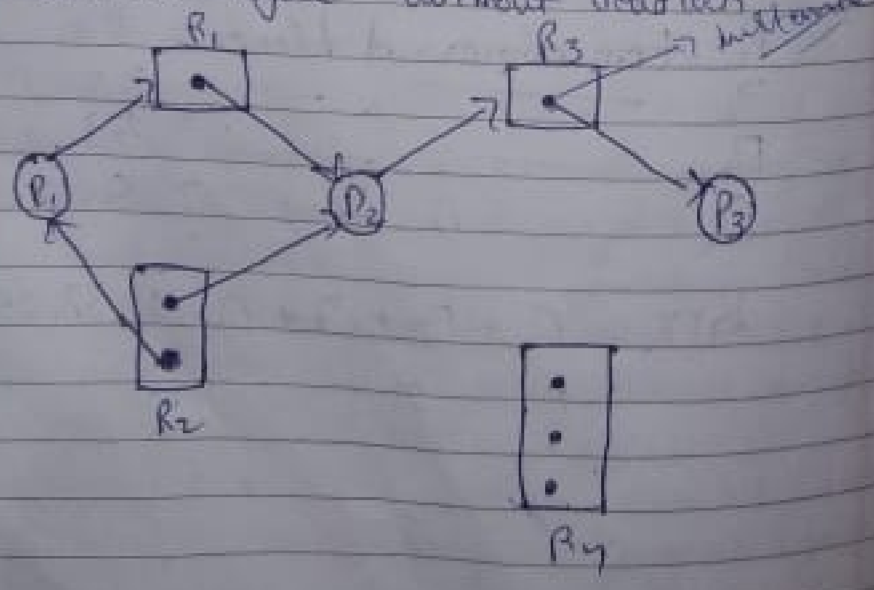
Arcs: resource request - request edge : ○ → □

Resource allocated - assignment edge □ → ○

→ The graph consists of V vertices and E edges. V is partitioned into 2 types P and R.
→ P = {P1, P2, ..., Pn} set of all active processes
→ R = {R1, R2, ..., Rn} Set of all resources.
→ Pi → Rj means Pi request for the resource Rj
Rj : request edge
→ Rj → Pi = assignment edge

example
a) without cycle without deadlock
b) with cycle with deadlock.
c) with cycle without deadlock.

a) without cycle without deadlock



2.V. ## Types of semaphores

*) Binary semaphore
*) counting semaphore

*) **Binary semaphore**:- A semaphore whose variable is allowed to take only 2 values of 0 (busy) and 1 (free) is called binary semaphore. Binary semaphore are known as mutex lock. as they provide mutual exclusion. They can be used to solve critical section problem for multiple processes.

```
do
{
    wait (mutex);
    critical section;
    signal (mutex);
    remainder section;
} while (true);
```

*) **Counting semaphore**:- Counting semaphore can be used to control access to a given resource consisting of finite number of instance.

```
In process P1
    S1;
  signal (synch);
In process P2
    wait (synch);
    S2;
```

Initially synch = 0, P2 execute S2 only after P1 executing S1.

Over lays :- over lays means replacement of a block of stored instruction

Ex :-

pass 1	70KB
pass 2	80KB
Symbol table	20KB
COMMON ROUTINES	30KB

Available memory = 150KB

Step 1 :- Start and Complete pass 1
Step 2 :- Jumps to the overlay driver, reads overlay B into memory overwriting overlay A
Step 3 :- Transfer control to pass 2
Step 4 :- completes pass 2.

The set of process P, resource R and edges E

i) P = {P1, P2, P3}
ii) R = {R1, R2, R3, R4}
iii) E = {P1 → R1, P2 → R3, R1 → P2, R3 → P3, R2 → P2, R2 → P1}

The resource instances are:
- one instance for R1
- Two instance for R2
- one instance for R3

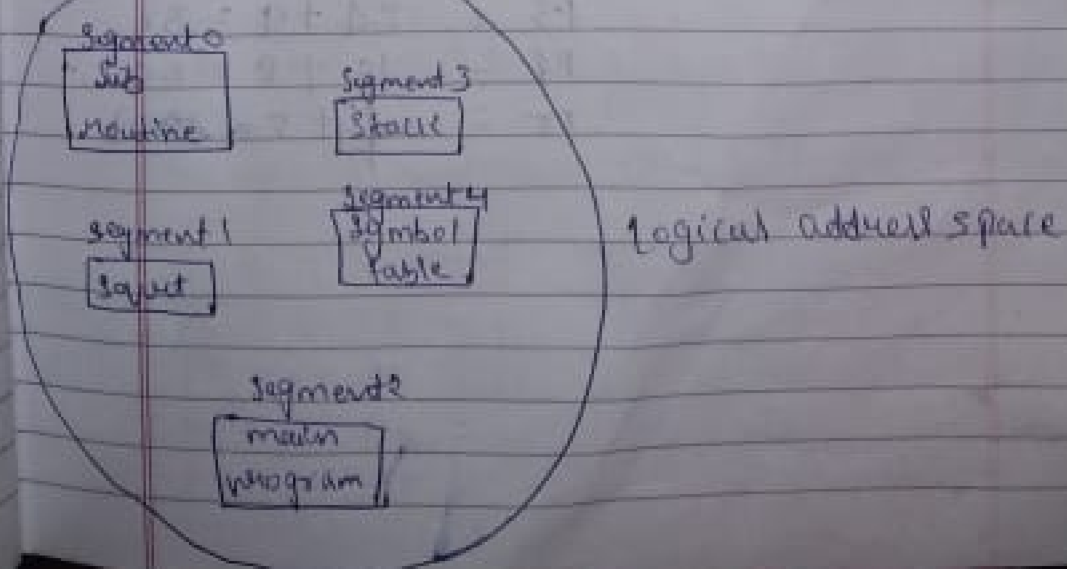
The process states are:-
→ P1 is holding an instance of R2, and is waiting for an instance of R1.
→ P2 is holding an instance of R2, R1 and is waiting for an instance of R3.
→ P3 is holding an instance of R3.
The resource allocation graph contains no cycle. There fore there is no deadlock situation.

5m
4. Deadlock:- (prevention) (or) necessary
A deadlock is a situation in which each process in a set of ~~process in a set of processes~~ is waiting for an event that only another process in the set can cause.

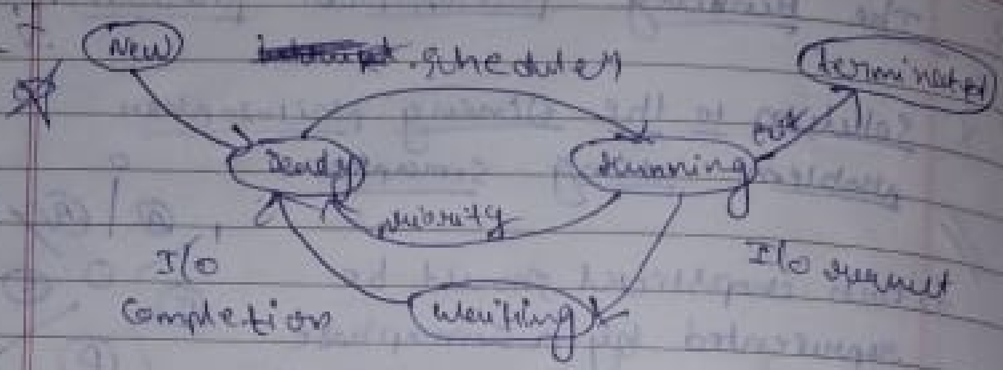
Segmentation:

*) A program is a collection of segments.
*) A segment is a logical unit such as main program
procedure
function
method
object
local variable, global variable
common block
stack
symbol table
arrays
*) Each segment is of different length element in the segment are identified by their offset from the beginning of the segment.
*) Each segment has a name and length
*) logical address consists of 2 tuple
$\langle$segment_numbers, offset$\rangle$
*) segment map table is created from each job dynamically.

Example



Different states of process is



- New ⇒ process create which is on second memory and will put into main memory
- After creation goes to ready state and we can say the process is ready.
- After ready state the process will go to running state and this process is known as scheduler
- If the process is completed then the process will be exited to terminate state.
- If the process need any I/O operation then the process will go to waiting state and after the completion of I/O operation it again goes to ready state and continue the process.

Starvation:-

Starvation:- In a heavily loaded computer system continuous stream of higher priority process can prevent the lower priority from scheduling into the CPU. This indefinite blocking for low priority process is called starvation.

Aging:- A solution to this problem can be done by the process called aging. Aging is a technique of gradually increasing the priority of a process.

→ what are the necessary conditions (or) prevention for deadlock.

Mutual exclusion: mutual exclusion means only one process can use the resource at time. if another process requests that resource The requesting process must wait until the resource has been released.

Hold and wait: A process must be holding at least one resource and waiting to acquire additional resource that are currently being held by other processes.

No pre-emption: Resources cannot be preempted, that is a resource can be released voluntarily by the process holding i.e. after it has completed its task. Resources cannot be forcibly taken from processes.

Circular wait: A set {P_0, P_1, .., P_n} of waiting processes must exist such that P_0 is waiting for a resource held by P_1. P_1 is waiting for the resource held by P_2 and P_{n-1} is waiting for the resource held by P_n. P_n is waiting for the resource held by P_0.

*. 2m Requirement of memory management

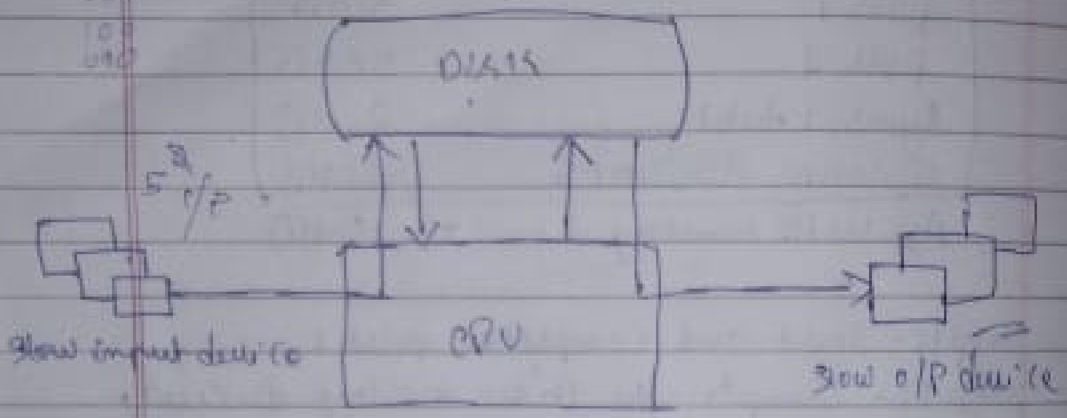
*) Address Binding
*) Dynamic loading
*) Dynamic linking
*) logical and physical address space.
*) overlays.

7.
Sol
*) A
*) A
Sy
a
*) Ea
in
of
*) Ea
*) lo
*)
Example
Separate
sub
routine
segment
sqrt

Spooling with a diagram

Ans:-

Diagram



(A spool is a buffer that holds output for a device) such as a printer, that cannot accepts interleaved data streams

-> Although a printer can serve only one job at a time, several applications may wish to print their o/p concurrently without having their o/p mixed together the o.s solves this problem by intercepting all o/p to the printer. Each application's o/p is spooled to a separate disk file.

Q) First fit, best fit, worst fit
-> First fit:- Allocate the first hole that is big enough to hold the process
•) Searching can start either at beginning or where the previous first-fit search ended.

→ **Best fit:-** Allocate the smallest hole that is big enough

*) Entire list must be searched, unless list is ordered.

*) produces smallest leftover hole.

Worst fit:- Allocate the largest hole.

*) must also search entire list

*) produces largest leftover hole.

Diagram

Device management

i) request device, release device
ii) read, write, reposition

information maintenance

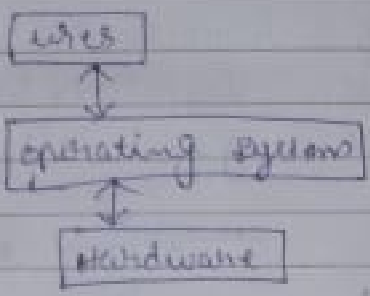
i) get time, set time
ii) get system data, set system date.

communications

i) create, delete communication connection
ii) send, receive messages.

Objective or Functions of O.S

An operating System acts as an interface between the user and the Hardware.



→ Objectives of OS

- To control and coordinate the use of System resources like CPU time, memory space, IO devices etc
- To provide a convenient environment for a user to access the available resources
- To increase the productivity of processing resources.
- To use the computer hardware in an efficient manner.

Types of System Call

Process control

- end, abort
- load, Execute

File management

- create file, delete file
- open, close