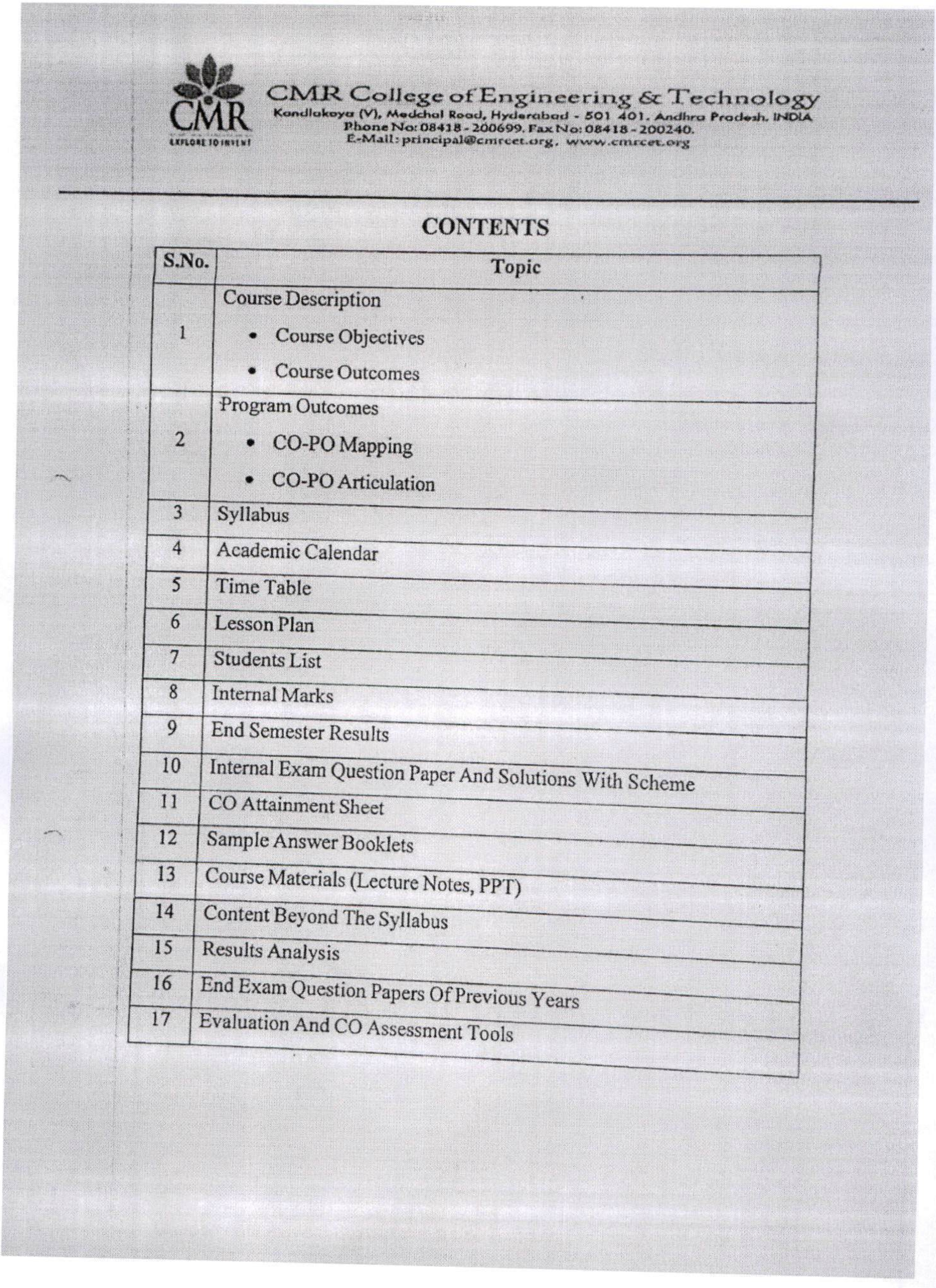


CMR College of Engineering & Technology
Kandlakoya (V), Medchal Road, Hyderabad - 501 401. Andhra Pradesh. INDIA
Phone No: 08418 - 200699. Fax No: 08418 - 200240.
E-Mail: principal@cmrcet.org, www.cmrcet.org

## CONTENTS

# Operating Systems

## Course Objectives:

Introduce operating system concepts (i.e., processes, threads, scheduling, synchronization,

- deadlocks, memory management, file and I/O subsystems and protection) Introduce the issues to be considered in the design and development of operating system
- Introduce basic Unix commands, system call interface for process management, interprocess
- communication and I/O in Unix

## Course Outcomes:

Will be able to control access to a computer and the files that may be shared

- Demonstrate the knowledge of the components of computer and their respective roles in
- computing. Ability to recognize and resolve user problems with standard operating environments

.• Gain practical knowledge of how programming languages, operating systems, and

- architectures interact and how to use each effectively.

## (A30516) OPERATING SYSTEMS

**B. Tech (CSE) V Semester**

| L | T | P | C |
|---|---|---|---|
| 3 | 0 | 0 | 3 |

### UNIT-I

Operating System Introduction, Structures - Simple Batch, Multi-programmed, Time-shared, Personal Computer, Parallel, Distributed Systems, Real-Time Systems, System components, Operating System services, System Calls.

### UNIT –II

**Process and CPU Scheduling**- Process concepts and scheduling, Operations on processes, Cooperating Processes, Threads, and Interposes Communication, Scheduling Criteria, Scheduling Algorithms, Multiple -Processor Scheduling.
System call interface for process management-fork, exit, wait, waitpid, exec

### UNIT –III

**Deadlocks** - System Model, Deadlocks Characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, and Recovery from Deadlock.

**Process Management and Synchronization**- The Critical Section Problem, Synchronization Hardware, Semaphores, and Classical Problems of Synchronization, Critical Regions, Monitors. Inter process Communication Mechanisms: IPC between processes on a single computer system, IPC
between processes on different systems, using pipes, FIFOs, message queues, shared memory.

### UNIT –IV

**Memory Management and Virtual Memory**- Logical versus Physical Address Space, Swapping, Contiguous Allocation, Paging, Segmentation, Segmentation with Paging, Demand Paging, Page Replacement, Page Replacement Algorithms.

### UNIT -V

**File System Interface and Operations**-Access methods, Directory Structure, Protection, File System Structure, Allocation methods, Free-space Management. usage of open, create, read, write, close, lseek, stat, ioctl, system calls

**Text Books:**

1. Operating System Principles- Abraham Silberchatz, Peter B. Galvin, Greg Gagne7th Edition, John Wiley
2. Advanced programming in the Unix environment, W.R. Stevens, Pearson education.

**Reference Books:**

1. Operating Systems – Internals and Design Principles, Stallings, 5th Edition, Pearson Education/PHI,2005.
2. Operating System A Design Approach-Crowley, TMH.
3. Modern Operating Systems, Andrew S Tanenbaum 2nd edition, Pearson/PHI.
4. Unix programming environment, Kernighan and Pike, PHI. / Pearson Education
5. Unix Internals the New Frontiers, U. Vahalia, Pearson Education.

**Course Outcomes**

Students shall be able to

1.Describe the components of computer and their respective roles in computing.

2.Explain process concepts and CPU Scheduling Algorithms

3.Demonstrate the Mutual exclusion, deadlock detection and Inter Process Communications.

4.Analyze various memory management and allocation methods.

5.Discuss File System Interface and Operations.

**END**

CMR
EXPLORE TO INVENT

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(UGC AUTONOMOUS)

Kandlakoya, Medchal Road, Hyderabad – 501401.

## ACADEMIC CALENDAR

Date: 24.06.2023

### B.Tech III Year - Academic Year 2023-2024

### I Semester

| S.No. | Description | Period | Duration |
|---|---|---|---|
| 1 | Commencement of Class Work | 21.08.2023 | -------- |
| 2 | First Spell of Instructions | 21.08.2023 to 14.10.2023 | 8 Weeks |
| 3 | *First Mid Examinations* | *16.10.2023 to 21.10.2023* | 1 Week |
| 4 | Dusara Vacation* | *23.10.2023to 28.10.2023* | 1 Week |
| 5 | Submission of Mid-I Marks to Exam Branch | 30.10.2023 | |
| 6 | Parent-Teacher Meeting | 04.11.2023 | |
| 7 | Second Spell of Instructions | 30.10.2023 to 23.12.2023 | 8 Weeks |
| 8 | *Second MidExaminations* | *25.12.2023 to 30.12.2023* | 1 Week |
| 9 | Submission of Mid-II Marks to Exam Branch | 06.01.2024 | |
| 10 | Preparations and Practical Examinations | 01.01.2024 to 06.01.2024 | 1 Week |
| 11 | *End Semester & Supplementary Examinations* | *08.01.2024 to 20.01.2024* | 2 Weeks |

### II Semester

| S.No | Description | Period | Duration |
|---|---|---|---|
| 1 | Commencement of Class Work | 22.01.2024 | -------- |
| 2 | First Spell of Instructions | 22.01.2024 to 16.03.2024 | 8 Weeks |
| 3 | *First Mid Examinations* | *18.03.2024 to 23.03.2024* | 1 Week |
| 4 | Submission of Mid-I Marks to Exam Branch | 30.03.2024 | |
| 5 | Parent-Teacher Meeting | 06.04.2024 | |
| 6 | Second Spell of Instructions | 25.03.2024 to 18.05.2024 | 8 Weeks |
| 7 | *Second MidExaminations* | *20.05.2024 to 25.05.2024* | 1 Week |
| 8 | Submission of Mid-II Marks to Exam Branch | 01.06.2024 | |
| 9 | Preparations and Practical examinations | 27.05.2024 to 01.06.2024 | 1 Week |
| 10 | *End Semester & Supplementary Examinations* | *03.06.2024 to 15.06.2024* | *2 Weeks* |
| 11 | *Summer vacation* | *17.06.2024 to 29.06.2024* | *2 Weeks* |
| 12 | Commencement of Class Work for the next A.Y 2024-2025 | 01.07.2024 | |

*Dusara Vacation (Subjected to declaration by JNTUH & TS Govt.)

24/6/2023

Principal
PRINCIPAL
CMR College of Engineering & Technology
(UGC.Autonomous)
Kandlakoya, Medchal Road, Hyderabad, T.S.

Copy submitted to Secretary: for kind information please

Copy to : 1. Deans
2. IQAC
3. All HODs
4. Administrative Officer
5. Accounts Officer
6. Web Portal In charge
7. ERP In Charge
8. Library
9. Student Notice Boards.

# CMR College of Engineering & Technology

Department of Computer Science & Engineering

## SESSION PLANNER

**Academic Year: 2022-2023** **Regulation:R18**

**Subject: Operating System** **Sub Code: A3051**

**Semester : V** **Class III -I Section :C&D**

**Faculty Name:P.Sravanthi**

| S.No | Subject<br>Topic Name/ Sub Topic Name | Books | No. of Periods | Cumulative No. of Periods | Delivery Method (Black Board/, PPT/Video ...etc) |
|---|---|---|---|---|---|
| | **UNIT-I** | | | | |
| 1 | Operating System Introduction | T1 | 1 | 1 | PPT,WB,NPTEL |
| 2 | Operating System Structures-Simple Batch, Multi-Programmed, | T1 | 2 | 3 | PPT, WB,NPTEL |
| 3 | Time-Shared, Personal Computer, Parallel, Distributed System, Real time | T1 | 2 | 5 | PPT,WB, |
| 4 | System Components | T1 | 1 | 6 | PPT,WB,NPTEL |
| 5 | Operating System Services | T1 | 2 | 8 | PPT,WB,NPTEL |
| 6 | System Calls | T1 | 2 | 10 | PPT,WB |
| | **UNIT-II** | | | | |
| 11 | Process Concepts And Scheduling | T1 | 2 | 12 | PPT,WB |
| 12 | Operations on Processes, Cooperation Processes | T1 | 2 | 14 | PPT,WB,NPTEL |
| 13 | Threads, Interposes Communication | T1 | 3 | 17 | PPT,WB,NPTEL |
| 14 | Scheduling Criteria, Scheduling Algorithms | T1 | 3 | 20 | PPT,WB,NPTEL |
| 15 | Multiple-Processor Scheduling | T1 | 1 | 21 | PPT,WB,NPTEL |
| 16 | System Call interface for process Management | T1 | 2 | 23 | PPT,WB,NPTEL |
| | **UNIT-III** | | | | |
| 17 | Deadlocks, Deadlock Characterization | T1 | 2 | 25 | PPT,WB,NPTEL |
| 18 | Methods for Handling Deadlock ,Prevention, Avoidance Detection, Recovery from Deadlock | T1 | 2 | 27 | PPT,WB |
| 19 | Critical Section Problem, Synchronization Hardware | T1 | 2 | 29 | PPT,WB |
| 20 | Semaphores, Classic problem of Synchronization | T1 | 2 | 31 | PPT,WB |
| 21 | Critical Regions, Monitor | T1 | 1 | 32 | PPT,WB,NPTEL |
| 22 | Inter Process Communication Mechanism between Processes on a Single | T1 | 1 | 33 | PPT,WB,NPTEL, |

| | | | | | |
|---|---|---|---|---|---|
| | Computer System | | | | |
| 23 | IPC between processes different System ,using ,pipes, FIFOs, Message Queues ,Shared Memory | T1 | 2 | 35 | PPT,WB,NPTEL |
| | **UNIT-IV** | | | | |
| 24 | Memory Management And Virtual Memory | T1 | 2 | 37 | PPT,WB,NPTEL |
| 25 | Segmentation | T1 | 2 | 39 | PPT,WB |
| 26 | Paging, Page Replacement Algorithms | T1 | 2 | 41 | PPT,WB |
| | **UNIT-V** | | | | |
| 27 | File System Structure, Directory Structure | T1 | 2 | 43 | PPT,WB,NPTEL |
| 28 | File Access Methods, Protection | T1 | 2 | 45 | PPT,WB , NPTEL |
| 29 | Free Space Management | T1 | 2 | 47 | PPT,WB, NPTEL |
| 30 | System Calls | T1 | 1 | 48 | PPT,WB |

**Text Books:**

1. Operating System Principles-Abraham Silberchatz, Peter B. Galvin, Greg Gagn7th Edition ,John Wiley
2. Advanced Programming in th Unix environment, W.R .Stevens, Pearson Education

**Reference Books:**

1. Operating Systems – Internals and Design Principles, Stallings, 5th Edition, Pearson Education/PHI,2005.
2. Operating System A Design Approach-Crowley, TMH.
3. Modern Operating Systems, Andrew S Tanenbaum 2nd edition, Pearson/PHI.
4. Unix programming environment, Kernighan and Pike, PHI. / Pearson Education
5. Unix Internals the New Frontiers, U. Vahalia, Pearson Education.

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

Kandlakoya (V), Medchal Road, Hyderabad -501401

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

| Sl. No. | Roll Number | Student Name | SEC |
|---|---|---|---|
| 1 | 21H51A0501 | BINGI NITHYASRI | A |
| 2 | 21H51A0503 | DASI RASHMIKA | A |
| 3 | 21H51A0505 | GOUNI PAVANI | A |
| 4 | 21H51A0508 | KOMMU VEERENDAR | A |
| 5 | 21H51A0514 | MOHAMMED ABDUL SAMEER | A |
| 6 | 21H51A0515 | MUAAZ MOHAMMED MUNEER | A |
| 7 | 21H51A0518 | PALTHYA SUMAN | A |
| 8 | 21H51A0519 | PAPPULA KARTHIK REDDY | A |
| 9 | 21H51A0520 | POSHETTY VARSHITH | A |
| 10 | 21H51A0521 | RITESH KUMAR | A |
| 11 | 21H51A0524 | TEJAVATH VASANTHA | A |
| 12 | 21H51A0525 | THOTA MAHESHWARI | A |
| 13 | 21H51A0526 | VEERELLI SAIVENKATA REDDY | A |
| 14 | 21H51A0529 | BELKONI ANVESH | A |
| 15 | 21H51A0533 | DASARI AJAY KUMAR | A |
| 16 | 21H51A0537 | GANTA NISHAL | A |
| 17 | 21H51A0540 | KOMMANABOINA ANUSHA | A |
| 18 | 21H51A0541 | LOKOTI SRICHARAN | A |
| 19 | 21H51A0542 | M KAVYA | A |
| 20 | 21H51A0544 | OJAS RAKESH GARPALLIWAR | A |
| 21 | 21H51A0545 | PEDDINTI SAI VARDHAN | A |
| 22 | 21H51A0547 | SATVIKA KARUMUDI | A |
| 23 | 21H51A0549 | THAMMISHETTY SHASHANK | A |
| 24 | 21H51A0550 | TUMMALA VENGAL RAYUDU | A |
| 25 | 21H51A0551 | UMMEDA SHIVA SAI KRISHNA | A |
| 26 | 21H51A0552 | VEMULA PRIYA PRAMIDHA | A |
| 27 | 21H51A0554 | ABHISHEK KUMAR SINGH | A |
| 28 | 21H51A0555 | ALETI ASHWITHA REDDY | A |
| 29 | 21H51A0556 | BATTU VICTOR DINAKAR BABU | A |
| 30 | 21H51A0559 | GANDRATH SRI YAGNA | A |
| 31 | 21H51A0562 | JOGU TARUN TEJA | A |
| 32 | 21H51A0563 | KARRA VINAY REDDY | A |
| 33 | 21H51A0569 | MOHAMMAD FERIA | A |
| 34 | 21H51A0570 | NAGULAPALLY UDAYKIRAN | A |
| 35 | 21H51A0572 | SARVADEY ZANETA | A |
| 36 | 21H51A0573 | SATHYARAM DHANA LAKSHMI | A |
| 37 | 21H51A0574 | SHA SOPNIL JAIN | A |
| 38 | 21H51A0578 | VUPPALA SHLAGHA | A |
| 39 | 21H51A0582 | JYOTHI BALAJI | A |
| 40 | 21H51A0583 | K RITIKA REDDY | A |
| 41 | 21H51A0584 | KOPPULA VENKATA SAI NANDINI | A |
| 42 | 21H51A0586 | M GANESH | A |

| Sl. No. | Roll Number | Student Name | SEC |
|---|---|---|---|
| 43 | 21H51A0592 | NENAVATH SRAVANI RATHOD | A |
| 44 | 21H51A0595 | PAVAN KUMAR | A |
| 45 | 21H51A0597 | ROSHAN TALARI | A |
| 46 | 21H51A0598 | S VARUN | A |
| 47 | 21H51A05A5 | AILENI SATHWIK | A |
| 48 | 21H51A05A6 | AKURATHI RITHVIK SESHAGIRI | A |
| 49 | 21H51A05A9 | BIJJAM SOUMIKA | A |
| 50 | 21H51A05B0 | BODA ASHOK | A |
| 51 | 21H51A05B6 | GOLLAPUDI NITHIN | A |
| 52 | 21H51A05C1 | NALLAKULA KIRANKUMAR | A |
| 53 | 21H51A05C4 | RITVIK PRATHAPANI | A |
| 54 | 22H55A0501 | AILLURI AMARDEEP REDDY | A |
| 55 | 22H55A0502 | BAIROJU SINDHU | A |
| 56 | 22H55A0503 | BODA AVINASH | A |
| 57 | 22H55A0504 | BODA RAHUL SAI KIRAN | A |
| 58 | 22H55A0505 | CHAKILAM BHARAT KUMAR | A |
| 59 | 22H55A0506 | ERLA VENU | A |
| 60 | 22H55A0507 | JONNALA SOWMYA | A |
| 61 | 22H55A0508 | KALE PRABHAS | A |
| 62 | 22H55A0509 | KATKAM MANASWINI | A |
| 63 | 22H55A0510 | KODIDALA KOMALI | A |
| 64 | 22H55A0511 | KONDA MAHIMASRI | A |
| 65 | 22H55A0512 | KONDAPARTHI MANJEERA | A |
| 66 | 22H55A0513 | KUMMARI RAJESH | A |
| 67 | 22H55A0514 | KURUMULA LOKESH | A |

CMR COLLEGE OF ENGINEERING & TECHNOLOGY
Kandlakoya (V), Medchal Road, Hyderabad -501401

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

| Sl. No. | Roll Number | Student Name | SEC |
|---|---|---|---|
| 1 | 21H51A0502 | DASARI HARINI | B |
| 2 | 21H51A0504 | GAJULAPALLE SREE LAKSHMI | B |
| 3 | 21H51A0506 | J AKANSH | B |
| 4 | 21H51A0507 | K ZAYD AHMED | B |
| 5 | 21H51A0509 | KURAPATI ESHWAR | B |
| 6 | 21H51A0510 | LAVANGU VAISHNAVI | B |
| 7 | 21H51A0511 | MAHANTHI SAI MANYA SRI | B |
| 8 | 21H51A0512 | MANAS CHHATWAL | B |
| 9 | 21H51A0513 | MANGINA SRI VENKATA SAI | B |
| 10 | 21H51A0516 | NAGIREDDY ANVITHA | B |
| 11 | 21H51A0517 | PADALA ANIL KUMAR | B |
| 12 | 21H51A0522 | SHREYASH SANJEEV KUMAR | B |
| 13 | 21H51A0523 | SIDDAMSHETTI SUMITH | B |
| 14 | 21H51A0527 | AKSHAT KALA | B |
| 15 | 21H51A0528 | ALAVALA KAVYA | B |
| 16 | 21H51A0530 | BENKI JYOTHIKA | B |
| 17 | 21H51A0531 | BENKI VARSHITHA RANI | B |
| 18 | 21H51A0532 | BOLLU HARI CHARHAN | B |
| 19 | 21H51A0534 | DAVULURI SAI SUJAN | B |
| 20 | 21H51A0535 | DESHAPATHI SAHITHI | B |
| 21 | 21H51A0536 | DHULIPALLA VENKATA SAI SIVA | B |
| 22 | 21H51A0539 | KOLAN SAHASRA REDDY | B |
| 23 | 21H51A0543 | MANGA TARAKA RATNA YOSHITH | B |
| 24 | 21H51A0546 | SAPNA TIWARI | B |
| 25 | 21H51A0548 | THAKUR ABHINAV SINGH | B |
| 26 | 21H51A0553 | ABBULA VINUTHNA | B |
| 27 | 21H51A0557 | BUCHENELLI NIKHILESH REDDY | B |
| 28 | 21H51A0558 | DANDA VENKATA SATHWIK REDDY | B |
| 29 | 21H51A0560 | GORINTA RAHULU | B |
| 30 | 21H51A0561 | GUNREDDY AKSHITH REDDY | B |
| 31 | 21H51A0564 | KODURU PRANATHI | B |
| 32 | 21H51A0565 | KONDA VISHAL GOUD | B |
| 33 | 21H51A0566 | KURAKULA SHAILESH | B |
| 34 | 21H51A0567 | MADIRA SAI RISHITHA | B |
| 35 | 21H51A0568 | MANURI CHANDU BABU | B |
| 36 | 21H51A0571 | NIMMALA SAI | B |
| 37 | 21H51A0575 | TUDURU SATHWIK | B |
| 38 | 21H51A0576 | U NAGA MANASWINI | B |
| 39 | 21H51A0577 | VARLA RAMAKRISHNA REDDY | B |

| Sl. No. | Roll Number | Student Name | SEC |
|---|---|---|---|
| 40 | 21H51A0579 | AMBATI ROHITH RAJU | B |
| 41 | 21H51A0580 | BAIRA ANUSHA | B |
| 42 | 21H51A0581 | GUNNALA AKHILA | B |
| 43 | 21H51A0585 | KUDUMULA ANVESH REDDY | B |
| 44 | 21H51A0587 | MANDALOJU VASANTH KUMAR | B |
| 45 | 21H51A0588 | MOHAMMAD ABDUL KALAM | B |
| 46 | 21H51A0589 | MOHAMMED MUDASSIR ALI | B |
| 47 | 21H51A0590 | NALABOLU MOUNIKA | B |
| 48 | 21H51A0591 | NAMPALLY SIDDHARTHA | B |
| 49 | 21H51A0593 | PAMULA BEULAH SUPRAGNYA | B |
| 50 | 21H51A0594 | PANCHAGNULA VINUTNA | B |
| 51 | 21H51A0596 | RAGE DAMODHAR | B |
| 52 | 21H51A0599 | SAI KIRAN B L S | B |
| 53 | 21H51A05A0 | SHESHAVAMATAM SUCHIT PAUL | B |
| 54 | 21H51A05A1 | TEEGALA BHANU TEJA REDDY | B |
| 55 | 21H51A05A2 | VADDI RISHIKA | B |
| 56 | 21H51A05A3 | YADDANAPUDI VISHNU SRIVATSAVA | B |
| 57 | 21H51A05A4 | YELDI ARUN | B |
| 58 | 21H51A05A7 | BAJRANG HARSH SINGH | B |
| 59 | 21H51A05A8 | BASAR SHYAM SUNDER RAO | B |
| 60 | 21H51A05B1 | BUNNI SHARANYA | B |
| 61 | 21H51A05B2 | C J VISHNU PRAKASH | B |
| 62 | 21H51A05B3 | CHIMMULA SHIVA PRASAD REDDY | B |
| 63 | 21H51A05B4 | DOLLA RENUKA | B |
| 64 | 21H51A05B5 | ERUKULA RAJASREE | B |
| 65 | 21H51A05B7 | HARIKA REDDY GANTA | B |
| 66 | 21H51A05B8 | INDUPALLI SHARONSUDHA | B |
| 67 | 21H51A05B9 | MADULAPURAM SAI YASHWANTH RAJ | B |
| 68 | 21H51A05C0 | MALLELA SINDHUJA | B |
| 69 | 21H51A05C2 | RANGU ABHINAV | B |
| 70 | 21H51A05C3 | RAYABARAPU CHATHURYA | B |
| 71 | 21H51A05C5 | SEGU JAYA BALA HARSHAVARDHAN | B |
| 72 | 21H51A05C8 | THATIKONDA AKHILA | B |
| 73 | 21H51A05C9 | VAKALA KAVYA SAI SUMA SRI | B |
| 74 | 21H51A05D1 | ANUJ KUMAR | B |
| 75 | 21H51A05D2 | BACHAWAR VINITHA | B |

CMR COLLEGE OF ENGINEERING & TECHNOLOGY
Kandlakoya (V), Medchal Road, Hyderabad -501401

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

| Sl. No. | Roll Number | Student Name | SEC |
|---|---|---|---|
| 1 | 21H51A05C6 | SOMU KOTESWARA REDDY | C |
| 2 | 21H51A05C7 | SUNKAPAKA JOHN | C |
| 3 | 21H51A05D0 | VALLAMKONDA POOJITHA | C |
| 4 | 21H51A05D3 | BASHAM RAJU | C |
| 5 | 21H51A05D4 | BUSSA TEJASWINI | C |
| 6 | 21H51A05D5 | DADE DINISHA | C |
| 7 | 21H51A05D6 | DEEKONDA SAKETH | C |
| 8 | 21H51A05E3 | MANCHI AKSHAYA | C |
| 9 | 21H51A05E4 | MOHAMMAD ARSHAD NIZAMI | C |
| 10 | 21H51A05F1 | P Y GEETHA MADHURI | C |
| 11 | 21H51A05F3 | SHAIK ILLIYAZ | C |
| 12 | 21H51A05F5 | TUSHAR PUNIA | C |
| 13 | 21H51A05F8 | DODDI SAI PHANI HARI CHANDANA | C |
| 14 | 21H51A05F9 | GADUGULA KALYANI | C |
| 15 | 21H51A05G2 | IYLA SNEHARIKA | C |
| 16 | 21H51A05G5 | KANUGO NESHIT RAJ | C |
| 17 | 21H51A05H2 | PODDUTURI NITHIN REDDY | C |
| 18 | 21H51A05H8 | TADEM RAVITEJA | C |
| 19 | 21H51A05J8 | GUNTHAPALLI MALINI | C |
| 20 | 21H51A05J9 | GURRAM KRISHNA PRASANTH | C |
| 21 | 21H51A05K3 | KODIGANTI SAI KISHORE | C |
| 22 | 21H51A05K8 | SEELAMSETTY PRASANNA GAYATHRI | C |
| 23 | 21H51A05L1 | SRIRAM NAGARAJU | C |
| 24 | 21H51A05L7 | YALLA TEJASWIK REDDY | C |
| 25 | 21H51A05M0 | CHILUKA SAI KARTHIK | C |
| 26 | 21H51A05M1 | DAMARLA HEMAVATHI | C |
| 27 | 21H51A05M4 | GIRAVENA ARYA | C |
| 28 | 21H51A05M9 | MOKIRALA JHANSI | C |
| 29 | 21H51A05N1 | NEELA SAI ADITYA | C |
| 30 | 21H51A05N3 | POTRU SAI NITISH | C |
| 31 | 21H51A05N4 | PRAHARSHITHA SURAGONI | C |
| 32 | 21H51A05N5 | PULI PRANEETH GOUD | C |
| 33 | 21H51A05P0 | TALOORI PRABHU KIRAN | C |
| 34 | 21H51A05P2 | VAVILLA RAVITEJA | C |
| 35 | 21H51A05P4 | ALLURI SAI SATHWIK REDDY | C |
| 36 | 21H51A05P5 | ANDE AJAY | C |
| 37 | 21H51A05P7 | BESTHA NANDA KISHORE | C |
| 38 | 21H51A05P8 | CHAVATAPALLI MUKUNDA SRI HASINI | C |
| 39 | 21H51A05P9 | CHEPYALA SATHWIK REDDY | C |

| Sl. No. | Roll Number | Student Name | SEC |
|---|---|---|---|
| 40 | 21H51A05Q1 | DAGGULA PRASHANTH | C |
| 41 | 21H51A05Q2 | GAJULA NAVANEETH | C |
| 42 | 21H51A05Q3 | GUDAPATI NITHIN KUMAR | C |
| 43 | 21H51A05R3 | PINAPATI ABHISHEK | C |
| 44 | 21H51A05R4 | RACHAMALLA SAI UJITHA REDDY | C |
| 45 | 21H51A05R5 | SATTU RAKESH | C |
| 46 | 21H51A05R6 | SHREYA M | C |
| 47 | 21H51A05R7 | YERAVELLI RUCHITHA | C |
| 48 | 22H55A0515 | M. SAI RANJITH REDDY | C |
| 49 | 22H55A0516 | MAHATHI DESAI | C |
| 50 | 22H55A0517 | MD TOWHEED | C |
| 51 | 22H55A0518 | MOHAMMED HANEF | C |
| 52 | 22H55A0519 | NAGARAM SHIVA CHAND | C |
| 53 | 22H55A0520 | NARGE CHARANETEJA | C |
| 54 | 22H55A0521 | NEELAM RAMYA SARI | C |
| 55 | 22H55A0522 | PANDAV SONIA | C |
| 56 | 22H55A0523 | PATHLAVATH SUNITHA | C |
| 57 | 22H55A0524 | POTTIPALLY DEEPIKA | C |
| 58 | 22H55A0525 | PULIGANTI MAHENDAR | C |
| 59 | 22H55A0526 | SARDESHI PRAVEEN KUMAR | C |
| 60 | 22H55A0527 | VISLAVATH ANITHA | C |

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

| Sl. No. | Roll Number | Student Name | SEC |
|---|---|---|---|
| 1 | 21H51A05D7 | DHUDURI SATHVIKA | D |
| 2 | 21H51A05D8 | GAMPALA SRI DURGA PRABHATH | D |
| 3 | 21H51A05D9 | GUNDLA VAMSHIDHAR | D |
| 4 | 21H51A05E0 | KASANAGOTTU AMULYA | D |
| 5 | 21H51A05E1 | KOTHA VAISHNAVI | D |
| 6 | 21H51A05E2 | KUMBALA ABHILASH REDDY | D |
| 7 | 21H51A05E6 | NAKKALA KEERTHANA | D |
| 8 | 21H51A05E7 | NEELAM BHARATH KUMAR | D |
| 9 | 21H51A05E8 | NEERUDI HARIPRASAD | D |
| 10 | 21H51A05E9 | ODURI VEERAMANIKANTA | D |
| 11 | 21H51A05F0 | OM GUPTA | D |
| 12 | 21H51A05F2 | ROHAN SACHIN RAKHE | D |
| 13 | 21H51A05F4 | SHAIK TASNIM | D |
| 14 | 21H51A05F6 | YARRAMSETTI MADHU VENKATA | D |
| 15 | 21H51A05F7 | BABBI THAPA | D |
| 16 | 21H51A05G0 | GUDIPALLY SAI SANJAY | D |
| 17 | 21H51A05G1 | GUNNA VINAY KUMAR REDDY | D |
| 18 | 21H51A05G3 | K SRI HARINI | D |
| 19 | 21H51A05G4 | KANDI SWETHA | D |
| 20 | 21H51A05G6 | KHANDESH THANU SRI | D |
| 21 | 21H51A05G7 | MAMIDI VENU GOPAL | D |
| 22 | 21H51A05G8 | MARAGONI KARTHIKEYA | D |
| 23 | 21H51A05G9 | NALIMELA JITHIN REDDY | D |
| 24 | 21H51A05H1 | PATRAYADI RAVI | D |
| 25 | 21H51A05H3 | POTHARAJU SAI KIRAN | D |
| 26 | 21H51A05H5 | SHERIKAR RAHUL | D |
| 27 | 21H51A05H6 | SOMARAJUPALLI THEJASWI | D |
| 28 | 21H51A05H7 | SUDAM SHIVA | D |
| 29 | 21H51A05H9 | THALLAM GEETHAN | D |
| 30 | 21H51A05J0 | TODUPUNURI SHAI PRIYA | D |
| 31 | 21H51A05J1 | TUMMALA SAHITH | D |
| 32 | 21H51A05J2 | VIJAYAGIRI AMULYA | D |
| 33 | 21H51A05J3 | ABHAY PRATAP SINGH | D |
| 34 | 21H51A05J4 | AYEMON ZEBA | D |
| 35 | 21H51A05J5 | BONDALA SRINATH | D |
| 36 | 21H51A05J6 | DODDAPANENI MEGHAN CHOWDARY | D |
| 37 | 21H51A05J7 | GORANTI SANTHU SATHWIK | D |
| 38 | 21H51A05K0 | KACHIREDDY JAYASREE | D |
| 39 | 21H51A05K1 | KAJA SANJEEV KUMAR | D |

| Sl. No. | Roll Number | Student Name | SEC |
|---|---|---|---|
| 40 | 21H51A05K2 | KANTU ANANTHKUMAR | D |
| 41 | 21H51A05K4 | KONDETI VIKRAMREDDY | D |
| 42 | 21H51A05K5 | KRITIKA KHATRI | D |
| 43 | 21H51A05K6 | NITYANANDAYYA MATHPATHI | D |
| 44 | 21H51A05K9 | SHANIGALA VISHNU | D |
| 45 | 21H51A05L0 | SINDEY ABHIGNA | D |
| 46 | 21H51A05L2 | SUMESH | D |
| 47 | 21H51A05L3 | TANNIRU MAHESH | D |
| 48 | 21H51A05L4 | TUSYAA SREERALA | D |
| 49 | 21H51A05L5 | UNI SAILESH | D |
| 50 | 21H51A05L6 | VAGUAMRI SRINANDHAN | D |
| 51 | 21H51A05L8 | BEHARA SURAJ | D |
| 52 | 21H51A05L9 | BHAKE SHASHANK | D |
| 53 | 21H51A05M2 | DIVYA GAUTAM | D |
| 54 | 21H51A05M3 | GANGASANI SHANKARSHAN | D |
| 55 | 21H51A05M5 | GUMMIREDDY SAINATH REDDY | D |
| 56 | 21H51A05M6 | KALLURI THANMAI | D |
| 57 | 21H51A05M7 | KATRAVATH MANJULA | D |
| 58 | 21H51A05M8 | MOHAMMED SAMEER ALI | D |
| 59 | 21H51A05N0 | NANCHARLA SAI AKSHITHA | D |
| 60 | 21H51A05N2 | OLIGE RANI | D |
| 61 | 21H51A05N6 | SAKKERLA RAJ KUMAR | D |
| 62 | 21H51A05N7 | SALENDRA MANOJ KUMAR | D |
| 63 | 21H51A05N8 | SHAIK JAVED | D |
| 64 | 21H51A05N9 | SHRIYA MALANI | D |
| 65 | 21H51A05P1 | VASURI VINAY KUMAR | D |
| 66 | 21H51A05P3 | VITTAPUR BINNU REDDY | D |
| 67 | 21H51A05P6 | BANOTHU DALI HIMASRI | D |
| 68 | 21H51A05Q0 | D GAYATHRI | D |
| 69 | 21H51A05Q4 | GUDIPUDI DHEERAJ | D |
| 70 | 21H51A05Q5 | GURRAM SRIKANTH | D |
| 71 | 21H51A05Q6 | KALVAKUNTA CHANDRASHEKAR | D |
| 72 | 21H51A05Q7 | KAPU HARSHA VARDAN REDDY | D |
| 73 | 21H51A05Q8 | KOTTE MOUNIKA | D |
| 74 | 21H51A05Q9 | MANDA VIGHNESHWARA REDDY | D |
| 75 | 21H51A05R0 | MANDHUMULA DEEPAK | D |
| 76 | 21H51A05R1 | PEDDI PRAVALIKA REDDY | D |
| 77 | 21H51A05R2 | PENDEM YOGITHA | D |
| 78 | 21H51A05R8 | YESUGARI ADHARSH | D |

# CMR College of Engineering & Technology

(UGC AUTONOMOUS)

Kandlakoya , Medchal Road - 501401

Department of Computer Science and Engineering

MID-I MARKS LIST

A.Y.2023-24

Class : III B.Tech. I SEM CSE SECTION-A

SUBJECT : G. Saidulu Operating System

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 1 | 21H51A0501 | BINGI NITHYASRI | 5 | 20 | 25 |
| 2 | 21H51A0503 | DASI RASHMIKA | 5 | 20 | 25 |
| 3 | 21H51A0505 | GOUNI PAVANI | 5 | 22 | 27 |
| 4 | 21H51A0508 | KOMMU VEERENDAR | 5 | 17 | 22 |
| 5 | 21H51A0514 | MOHAMMED ABDUL SAMEER | AB | 18 | 18 |
| 6 | 21H51A0515 | MUAAZ MOHAMMED MUNEER | 5 | 22 | 27 |
| 7 | 21H51A0518 | PALTHYA SUMAN | 5 | 07 | 12 |
| 8 | 21H51A0519 | PAPPULA KARTHIK REDDY | AB | 14 | 14 |
| 9 | 21H51A0520 | POSHETTY VARSHITH | AB | 14 | 14 |
| 10 | 21H51A0521 | RITESH KUMAR | 5 | 11 | 16 |
| 11 | 21H51A0524 | TEJAVATH VASANTHA | 5 | 22 | 27 |
| 12 | 21H51A0525 | THOTA MAHESHWARI | 5 | 17 | 22 |
| 13 | 21H51A0526 | VEERELLI SAIVENKATA REDDY | AB | AB | AB |
| 14 | 21H51A0529 | BELKONI ANVESH | 5 | 20 | 25 |
| 15 | 21H51A0533 | DASARI AJAY KUMAR | 5 | 16 | 21 |
| 16 | 21H51A0537 | GANTA NISHAL | 5 | 18 | 23 |
| 17 | 21H51A0540 | KOMMANABOINA ANUSHA | 5 | 21 | 26 |
| 18 | 21H51A0541 | LOKOTI SRICHARAN | 5 | 08 | 13 |
| 19 | 21H51A0542 | M KAVYA | 5 | 18 | 23 |
| 20 | 21H51A0544 | OJAS RAKESH GARPALLIWAR | AB | 17 | 17 |
| 21 | 21H51A0545 | PEDDINTI SAI VARDHAN | 5 | 0 | 5 |
| 22 | 21H51A0547 | SATVIKA KARUMUDI | 5 | 20 | 25 |
| 23 | 21H51A0549 | THAMMISHETTY SHASHANK | 5 | 13 | 18 |
| 24 | 21H51A0550 | TUMMALA VENGAL RAYUDU | 5 | 05 | 10 |
| 25 | 21H51A0551 | UMMEDA SHIVA SAI KRISHNA | AB | 18 | 18 |
| 26 | 21H51A0552 | VEMULA PRIYA PRAMIDHA | 5 | 24 | 29 |
| 27 | 21H51A0554 | ABHISHEK KUMAR SINGH | 5 | 24 | 29 |
| 28 | 21H51A0555 | ALETI ASHWITHA REDDY | 5 | 21 | 26 |
| 29 | 21H51A0556 | BATTU VICTOR DINAKAR BABU | 5 | 16 | 21 |
| 30 | 21H51A0559 | GANDRATH SRI YAGNA | AB | 21 | 21 |
| 31 | 21H51A0562 | JOGU TARUN TEJA | AB | 15 | 15 |
| 32 | 21H51A0563 | KARRA VINAY REDDY | 5 | 20 | 25 |
| 33 | 21H51A0569 | MOHAMMAD FERIA | AB | 19 | 19 |
| 34 | 21H51A0570 | NAGULAPALLY UDAYKIRAN | 5 | 16 | 21 |
| 35 | 21H51A0572 | SARVADEY ZANETA | 5 | 21 | 26 |
| 36 | 21H51A0573 | SATHYARAM DHANA LAKSHMI | 5 | 21 | 26 |
| 37 | 21H51A0574 | SHA SOPNIL JAIN | 5 | 19 | 24 |
| 38 | 21H51A0578 | VUPPALA SHLAGHA | 5 | 16 | 21 |
| 39 | 21H51A0582 | JYOTHI BALAJI | 5 | 5 | 10 |
| 40 | 21H51A0583 | K RITIKA REDDY | AB | 7 | 07 |

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 41 | 21H51A0584 | KOPPULA VENKATA SAI NANDINI | 5 | 19 | 24 |
| 42 | 21H51A0586 | M GANESH | 5 | 17 | 22 |
| 43 | 21H51A0592 | NENAVATH SRAVANI RATHOD | 5 | 20 | 25 |
| 44 | 21H51A0595 | PAVAN KUMAR | AB | 14 | 14 |
| 45 | 21H51A0597 | ROSHAN TALARI | AB | 15 | 15 |
| 46 | 21H51A0598 | S VARUN | 5 | 19 | 24 |
| 47 | 21H51A05A5 | AILENI SATHWIK | 5 | 16 | 21 |
| 48 | 21H51A05A6 | AKURATHI RITHVIK SESHAGIRI | 5 | 16 | 21 |
| 49 | 21H51A05A9 | BIJJAM SOUMIKA | AB | 15 | 15 |
| 50 | 21H51A05B0 | BODA ASHOK | AB | 19 | 19 |
| 51 | 21H51A05B6 | GOLLAPUDI NITHIN | 5 | 14 | 19 |
| 52 | 21H51A05C1 | NALLAKULA KIRANKUMAR | 5 | 15 | 20 |
| 53 | 21H51A05C4 | RITVIK PRATHAPANI | 5 | 13 | 18 |
| 54 | 22H55A0501 | AILLURI AMARDEEP REDDY | 5 | 19 | 24 |
| 55 | 22H55A0502 | BAIROJU SINDHU | 5 | 20 | 25 |
| 56 | 22H55A0503 | BODA AVINASH | 5 | 21 | 26 |
| 57 | 22H55A0504 | BODA RAHUL SAI KIRAN | AB | 19 | 19 |
| 58 | 22H55A0505 | CHAKILAM BHARAT KUMAR | 5 | 24 | 29 |
| 59 | 22H55A0506 | ERLA VENU | 5 | 23 | 28 |
| 60 | 22H55A0507 | JONNALA SOWMYA | 5 | 21 | 26 |
| 61 | 22H55A0508 | KALE PRABHAS | 5 | 25 | 30 |
| 62 | 22H55A0509 | KATKAM MANASWINI | 5 | 20 | 25 |
| 63 | 22H55A0510 | KODIDALA KOMALI | 5 | 22 | 27 |
| 64 | 22H55A0511 | KONDA MAHIMASRI | 5 | 23 | 28 |
| 65 | 22H55A0512 | KONDAPARTHI MANJEERA | 5 | 21 | 26 |
| 66 | 22H55A0513 | KUMMARI RAJESH | 5 | 18 | 23 |
| 67 | 22H55A0514 | KURUMULA LOKESH | AB | 17 | 17 |

| | |
|---|---|
| Name&Signature of the Faculty : | G.Saidulu |
| Department : | CSE |
| Mobile No : | 9603131030 |

HOD/CSE

# CMR College of Engineering & Technology

(UGC AUTONOMOUS)

Kandlakoya , Medchal Road - 501401

CMR

## Department of Computer Science and Engineering

MID-I MARKS LIST

A.Y.2023-24

Class : III B.Tech. I SEM CSE SECTION-B

SUBJECT : Operating Systems

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 1 | 21H51A0502 | DASARI HARINI | 5 | 21 | 26 |
| 2 | 21H51A0504 | GAJULAPALLE SREE LAKSHMI | 5 | 21 | 26 |
| 3 | 21H51A0506 | J AKANSH | 5 | 18 | 23 |
| 4 | 21H51A0507 | K ZAYD AHMED | 5 | 12 | 17 |
| 5 | 21H51A0509 | KURAPATI ESHWAR | 5 | 21 | 26 |
| 6 | 21H51A0510 | LAVANGU VAISHNAVI | 5 | 20 | 25 |
| 7 | 21H51A0511 | MAHANTHI SAI MANYA SRI | 5 | 20 | 25 |
| 8 | 21H51A0512 | MANAS CHHATWAL | 5 | 21 | 26 |
| 9 | 21H51A0513 | MANGINA SRI VENKATA SAI | 5 | 16 | 21 |
| 10 | 21H51A0516 | NAGIREDDY ANVITHA | 5 | 20 | 25 |
| 11 | 21H51A0517 | PADALA ANIL KUMAR | 5 | 11 | 16 |
| 12 | 21H51A0522 | SHREYASH SANJEEV KUMAR | 5 | 21 | 26 |
| 13 | 21H51A0523 | SIDDAMSHETTI SUMITH | 5 | 14 | 19 |
| 14 | 21H51A0527 | AKSHAT KALA | 5 | 22 | 27 |
| 15 | 21H51A0528 | ALAVALA KAVYA | 5 | 23 | 28 |
| 16 | 21H51A0530 | BENKI JYOTHIKA | 5 | 20 | 25 |
| 17 | 21H51A0531 | BENKI VARSHITHA RANI | 5 | 22 | 27 |
| 18 | 21H51A0532 | BOLLU HARI CHARHAN | 5 | 15 | 20 |
| 19 | 21H51A0534 | DAVULURI SAI SUJAN | 5 | 24 | 29 |
| 20 | 21H51A0535 | DESHAPATHI SAHITHI | 5 | 20 | 25 |
| 21 | 21H51A0536 | DHULIPALLA VENKATA SAI SIVA | 5 | 17 | 22 |
| 22 | 21H51A0539 | KOLAN SAHASRA REDDY | 5 | 21 | 26 |
| 23 | 21H51A0543 | MANGA TARAKA RATNA YOSHITH | 5 | 18 | 23 |
| 24 | 21H51A0546 | SAPNA TIWARI | 5 | 15 | 20 |
| 25 | 21H51A0548 | THAKUR ABHINAV SINGH | 5 | 20 | 25 |
| 26 | 21H51A0553 | ABBULA VINUTHNA | 5 | 20 | 25 |
| 27 | 21H51A0557 | BUCHENELLI NIKHILESH REDDY | 5 | 20 | 25 |
| 28 | 21H51A0558 | DANDA VENKATA SATHWIK REDDY | 5 | 19 | 24 |
| 29 | 21H51A0560 | GORINTA RAHULU | 5 | 19 | 24 |
| 30 | 21H51A0561 | GUNREDDY AKSHITH REDDY | 5 | 19 | 24 |
| 31 | 21H51A0564 | KODURU PRANATHI | 5 | 25 | 30 |
| 32 | 21H51A0565 | KONDA VISHAL GOUD | 5 | 17 | 22 |
| 33 | 21H51A0566 | KURAKULA SHAILESH | 5 | 19 | 24 |
| 34 | 21H51A0567 | MADIRA SAI RISHITHA | 5 | 22 | 27 |
| 35 | 21H51A0568 | MANURI CHANDU BABU | 5 | 18 | 23 |
| 36 | 21H51A0571 | NIMMALA SAI | 5 | 14 | 19 |
| 37 | 21H51A0575 | TUDURU SATHWIK | 5 | 23 | 28 |
| 38 | 21H51A0576 | U NAGA MANASWINI | 5 | 21 | 26 |
| 39 | 21H51A0577 | VARLA RAMAKRISHNA REDDY | 5 | 21 | 26 |
| 40 | 21H51A0579 | AMBATI ROHITH RAJU | 5 | 19 | 24 |

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 41 | 21H51A0580 | BAIRA ANUSHA | 5 | 20 | 25 |
| 42 | 21H51A0581 | GUNNALA AKHILA | 5 | 25 | 30 |
| 43 | 21H51A0585 | KUDUMULA ANVESH REDDY | 5 | 21 | 26 |
| 44 | 21H51A0587 | MANDALOJU VASANTH KUMAR | 5 | 20 | 25 |
| 45 | 21H51A0588 | MOHAMMAD ABDUL KALAM | 5 | 14 | 19 |
| 46 | 21H51A0589 | MOHAMMED MUDASSIR ALI | 5 | 14 | 19 |
| 47 | 21H51A0590 | NALABOLU MOUNIKA | 5 | 16 | 21 |
| 48 | 21H51A0591 | NAMPALLY SIDDHARTHA | 5 | 24 | 29 |
| 49 | 21H51A0593 | PAMULA BEULAH SUPRAGNYA | 5 | 13 | 18 |
| 50 | 21H51A0594 | PANCHAGNULA VINUTNA | 5 | 24 | 29 |
| 51 | 21H51A0596 | RAGE DAMODHAR | 5 | 15 | 20 |
| 52 | 21H51A0599 | SAI KIRAN B L S | 5 | 10 | 15 |
| 53 | 21H51A05A0 | SHESHAVAMATAM SUCHIT PAUL | 5 | 14 | 19 |
| 54 | 21H51A05A1 | TEEGALA BHANU TEJA REDDY | 5 | 23 | 28 |
| 55 | 21H51A05A2 | VADDI RISHIKA | 5 | 22 | 27 |
| 56 | 21H51A05A3 | YADDANAPUDI VISHNU SRIVATSAVA | 5 | 18 | 23 |
| 57 | 21H51A05A4 | YELDI ARUN | 5 | 17 | 22 |
| 58 | 21H51A05A7 | BAJRANG HARSH SINGH | 5 | 19 | 24 |
| 59 | 21H51A05A8 | BASAR SHYAM SUNDER RAO | 5 | 22 | 27 |
| 60 | 21H51A05B1 | BUNNI SHARANYA | 5 | 25 | 30 |
| 61 | 21H51A05B2 | C J VISHNU PRAKASH | 5 | 22 | 27 |
| 62 | 21H51A05B3 | CHIMMULA SHIVA PRASAD REDDY | 5 | 21 | 26 |
| 63 | 21H51A05B4 | DOLLA RENUKA | 5 | 19 | 24 |
| 64 | 21H51A05B5 | ERUKULA RAJASREE | 5 | 16 | 21 |
| 65 | 21H51A05B7 | HARIKA REDDY GANTA | 5 | 18 | 23 |
| 66 | 21H51A05B8 | INDUPALLI SHARONSUDHA | 5 | 18 | 23 |
| 67 | 21H51A05B9 | MADULAPURAM SAI YASHWANTH RAJ | 5 | 18 | 23 |
| 68 | 21H51A05C0 | MALLELA SINDHUJA | 5 | 24 | 29 |
| 69 | 21H51A05C2 | RANGU ABHINAV | 5 | 16 | 21 |
| 70 | 21H51A05C3 | RAYABARAPU CHATHURYA | 5 | 21 | 26 |
| 71 | 21H51A05C5 | SEGU JAYA BALA HARSHAVARDHAN | 5 | 19 | 24 |
| 72 | 21H51A05C8 | THATIKONDA AKHILA | 5 | 19 | 24 |
| 73 | 21H51A05C9 | VAKALA KAVYA SAI SUMA SRI | 5 | 18 | 23 |
| 74 | 21H51A05D1 | ANUJ KUMAR | 5 | 18 | 23 |
| 75 | 21H51A05D2 | BACHAWAR VINITHA | 5 | 17 | 22 |

Name&Signature of the Faculty : Dr. G. RAVI KUMAR

Department : CSE

Mobile No : 9849037283

HOD/CSE

# CMR College of Engineering & Technology

(UGC AUTONOMOUS)

Kandlakoya , Medchal Road - 501401

Department of Computer Science and Engineering

MID-I MARKS LIST

Class : III B.Tech. I SEM CSE SECTION-C A.Y.2023-24

SUBJECT : Operating Systems

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 1 | 21H51A05C6 | SOMU KOTESWARA REDDY | 5 | 13 | 18 |
| 2 | 21H51A05C7 | SUNKAPAKA JOHN | 5 | 11 | 16 |
| 3 | 21H51A05D0 | VALLAMKONDA POOJITHA | 5 | 06 | 11 |
| 4 | 21H51A05D3 | BASHAM RAJU | 5 | 17 | 22 |
| 5 | 21H51A05D4 | BUSSA TEJASWINI | 5 | 09 | 14 |
| 6 | 21H51A05D5 | DADE DINISHA | 5 | 13 | 18 |
| 7 | 21H51A05D6 | DEEKONDA SAKETH | 0 | 02 | 02 |
| 8 | 21H51A05E3 | MANCHI AKSHAYA | 5 | 20 | 25 |
| 9 | 21H51A05E4 | MOHAMMAD ARSHAD NIZAMI | 5 | 07 | 12 |
| 10 | 21H51A05F1 | P Y GEETHA MADHURI | 5 | 10 | 15 |
| 11 | 21H51A05F3 | SHAIK ILLIYAZ | 0 | 08 | 08 |
| 12 | 21H51A05F5 | TUSHAR PUNIA | 5 | 12 | 17 |
| 13 | 21H51A05F8 | DODDI SAI PHANI HARI CHANDANA | 5 | 15 | 20 |
| 14 | 21H51A05F9 | GADUGULA KALYANI | 5 | 13 | 18 |
| 15 | 21H51A05G2 | IYLA SNEHARIKA | 5 | 20 | 25 |
| 16 | 21H51A05G5 | KANUGO NESHIT RAJ | 5 | 08 | 13 |
| 17 | 21H51A05H2 | PODDUTURI NITHIN REDDY | 5 | 0 | 5 |
| 18 | 21H51A05H8 | TADEM RAVITEJA | 5 | 09 | 14 |
| 19 | 21H51A05J8 | GUNTHAPALLI MALINI | 5 | 09 | 14 |
| 20 | 21H51A05J9 | GURRAM KRISHNA PRASANTH | 5 | 13 | 18 |
| 21 | 21H51A05K3 | KODIGANTI SAI KISHORE | ~~AB~~ | ~~12~~ | ~~12~~ |
| 22 | 21H51A05K8 | SEELAMSETTY PRASANNA GAYATHRI | 5 | 21 | 26 |
| 23 | 21H51A05L1 | SRIRAM NAGARAJU | 5 | 18 | 23 |
| 24 | 21H51A05L7 | YALLA TEJASWIK REDDY | 5 | 05 | 10 |
| 25 | 21H51A05L8 | BEHARA SURAJ | 4 | 14 | 18 |
| 26 | 21H51A05M0 | CHILUKA SAI KARTHIK | 5 | 22 | 27 |
| 27 | 21H51A05M1 | DAMARLA HEMAVATHI | 5 | 12 | 17 |
| 28 | 21H51A05M4 | GIRAVENA ARYA | 5 | 13 | 18 |
| 29 | 21H51A05M9 | MOKIRALA JHANSI | 5 | 16 | 21 |
| 30 | 21H51A05N1 | NEELA SAI ADITYA | 5 | 23 | 28 |
| 31 | 21H51A05N3 | POTRU SAI NITISH | 5 | 09 | 14 |
| 32 | 21H51A05N4 | PRAHARSHITHA SURAGONI | 5 | 21 | 26 |
| 33 | 21H51A05N5 | PULI PRANEETH GOUD | 5 | 15 | 20 |
| 34 | 21H51A05P0 | TALOORI PRABHU KIRAN | 5 | 07 | 12 |
| 35 | 21H51A05P2 | VAVILLA RAVITEJA | 5 | 14 | 19 |
| 36 | 21H51A05P4 | ALLURI SAI SATHWIK REDDY | 5 | 17 | 22 |
| 37 | 21H51A05P5 | ANDE AJAY | 5 | 16 | 21 |
| 38 | 21H51A05P7 | BESTHA NANDA KISHORE | 5 | 23 | 28 |
| 39 | 21H51A05P8 | CHAVATAPALLI MUKUNDA SRI HASINI | 5 | 14 | 19 |
| 40 | 21H51A05P9 | CHEPYALA SATHWIK REDDY | 0 | 10 | 10 |

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 41 | 21H51A05Q1 | DAGGULA PRASHANTH | 5 | 08 | 13 |
| 42 | 21H51A05Q2 | GAJULA NAVANEETH | 5 | 13 | 18 |
| 43 | 21H51A05Q3 | GUDAPATI NITHIN KUMAR | 5 | 10 | 15 |
| 44 | 21H51A05R3 | PINAPATI ABHISHEK | 5 | 14 | 19 |
| 45 | 21H51A05R4 | RACHAMALLA SAI UJITHA REDDY | 5 | 06 | 11 |
| 46 | 21H51A05R5 | SATTU RAKESH | 5 | 08 | 13 |
| 47 | 21H51A05R6 | SHREYA M | 5 | 13 | 18 |
| 48 | 21H51A05R7 | YERAVELLI RUCHITHA | 5 | 15 | 20 |
| 49 | 22H55A0515 | M. SAI RANJITH REDDY | 5 | 16 | 21 |
| 50 | 22H55A0516 | MAHATHI DESAI | 5 | 10 | 15 |
| 51 | 22H55A0517 | MD TOWHEED | 5 | 5 | 10 |
| 52 | 22H55A0518 | MOHAMMED HANEF | 5 | 03 | 08 |
| 53 | 22H55A0519 | NAGARAM SHIVA CHAND | 5 | 17 | 22 |
| 54 | 22H55A0520 | NARGE CHARANETEJA | 5 | 13 | 18 |
| 55 | 22H55A0521 | NEELAM RAMYA SARI | 5 | 17 | 22 |
| 56 | 22H55A0522 | PANDAV SONIA | 5 | 21 | 26 |
| 57 | 22H55A0523 | PATHLAVATH SUNITHA | 5 | 21 | 26 |
| 58 | 22H55A0524 | POTTIPALLY DEEPIKA | 5 | 18 | 23 |
| 59 | 22H55A0525 | PULIGANTI MAHENDAR | 5 | 15 | 20 |
| 60 | 22H55A0526 | SARDESHI PRAVEEN KUMAR | 5 | 14 | 19 |
| 61 | 22H55A0527 | VISLAVATH ANITHA | 5 | 15 | 20 |

| |
|---|
| Name&Signature of the Faculty : P. Sravanthi |
| Department : CSE |
| Mobile No : 9989448869 |

HOD/CSE

CMR

# CMR College of Engineering & Technology

(UGC AUTONOMOUS)

Kandlakoya , Medchal Road - 501401

Department of Computer Science and Engineering

MID-I MARKS LIST

Class : III B.Tech. I SEM CSE SECTION-D A.Y.2023-24

SUBJECT : operating system

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 1 | 21H51A05D7 | DHUDURI SATHVIKA | 5 | 20 | 25 |
| 2 | 21H51A05D8 | GAMPALA SRI DURGA PRABHATH | 5 | 20 | 25 |
| 3 | 21H51A05D9 | GUNDLA VAMSHIDHAR | 5 | 16 | 21 |
| 4 | 21H51A05E0 | KASANAGOTTU AMULYA | 5 | 20 | 25 |
| 5 | 21H51A05E1 | KOTHA VAISHNAVI | 5 | 18 | 23 |
| 6 | 21H51A05E2 | KUMBALA ABHILASH REDDY | 5 | 20 | 25 |
| 7 | 21H51A05E6 | NAKKALA KEERTHANA | 5 | 17 | 22 |
| 8 | 21H51A05E7 | NEELAM BHARATH KUMAR | 5 | 17 | 22 |
| 9 | 21H51A05E8 | NEERUDI HARIPRASAD | 5 | 23 | 28 |
| 10 | 21H51A05E9 | ODURI VEERAMANIKANTA | 5 | 13 | 18 |
| 11 | 21H51A05F0 | OM GUPTA | 5 | 20 | 25 |
| 12 | 21H51A05F2 | ROHAN SACHIN RAKHE | 5 | 20 | 25 |
| 13 | 21H51A05F4 | SHAIK TASNIM | 5 | 17 | 22 |
| 14 | 21H51A05F6 | YARRAMSETTI MADHU VENKATA | 5 | 22 | 27 |
| 15 | 21H51A05F7 | BABBI THAPA | 5 | 16 | 21 |
| 16 | 21H51A05G0 | GUDIPALLY SAI SANJAY | 5 | 21 | 26 |
| 17 | 21H51A05G1 | GUNNA VINAY KUMAR REDDY | 5 | 20 | 25 |
| 18 | 21H51A05G3 | K SRI HARINI | 5 | 20 | 25 |
| 19 | 21H51A05G4 | KANDI SWETHA | 5 | 20 | 25 |
| 20 | 21H51A05G6 | KHANDESH THANU SRI | 5 | 18 | 23 |
| 21 | 21H51A05G7 | MAMIDI VENU GOPAL | 5 | 11 | 16 |
| 22 | 21H51A05G8 | MARAGONI KARTHIKEYA | 5 | 20 | 25 |
| 23 | 21H51A05G9 | NALIMELA JITHIN REDDY | 5 | 09 | 14 |
| 24 | 21H51A05H1 | PATRAYADI RAVI | 5 | 14 | 19 |
| 25 | 21H51A05H3 | POTHARAJU SAI KIRAN | 5 | 18 | 23 |
| 26 | 21H51A05H5 | SHERIKAR RAHUL | 5 | 20 | 25 |
| 27 | 21H51A05H6 | SOMARAJUPALLI THEJASWI | 5 | 24 | 29 |
| 28 | 21H51A05H7 | SUDAM SHIVA | 5 | 15 | 20 |
| 29 | 21H51A05H9 | THALLAM GEETHAN | 5 | 16 | 21 |
| 30 | 21H51A05J0 | TODUPUNURI SHAI PRIYA | 5 | 20 | 25 |
| 31 | 21H51A05J1 | TUMMALA SAHITH | 4 | 01 | 5 |
| 32 | 21H51A05J2 | VIJAYAGIRI AMULYA | 5 | 21 | 26 |
| 33 | 21H51A05J3 | ABHAY PRATAP SINGH | 5 | 16 | 21 |
| 34 | 21H51A05J4 | AYEMON ZEBA | 5 | 14 | 19 |
| 35 | 21H51A05J5 | BONDALA SRINATH | 5 | 16 | 21 |
| 36 | 21H51A05J6 | DODDAPANENI MEGHAN CHOWDARY | 5 | 13 | 18 |
| 37 | 21H51A05J7 | GORANTI SANTHU SATHWIK | 5 | 19 | 24 |
| 38 | 21H51A05K0 | KACHIREDDY JAYASREE | 5 | 22 | 27 |
| 39 | 21H51A05K1 | KAJA SANJEEV KUMAR | 5 | 19 | 24 |
| 40 | 21H51A05K2 | KANTU ANANTHKUMAR | 5 | 17 | 22 |

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 41 | 21H51A05K4 | KONDETI VIKRAMREDDY | 5 | 18 | 23 |
| 42 | 21H51A05K5 | KRITIKA KHATRI | 5 | 19 | 24 |
| 43 | 21H51A05K6 | NITYANANDAYYA MATHPATHI | 5 | 15 | 20 |
| 44 | 21H51A05K9 | SHANIGALA VISHNU | 5 | 19 | 24 |
| 45 | 21H51A05L0 | SINDEY ABHIGNA | 5 | 23 | 28 |
| 46 | 21H51A05L2 | SUMESH | 5 | 19 | 24 |
| 47 | 21H51A05L3 | TANNIRU MAHESH | 5 | 21 | 26 |
| 48 | 21H51A05L4 | TUSYAA SREERALA | 5 | 16 | 21 |
| 49 | 21H51A05L5 | UNI SAILESH | 5 | 23 | 28 |
| 50 | 21H51A05L6 | VAGUAMRI SRINANDHAN | 5 | 20 | 25 |
| 51 | 21H51A05L9 | BHAKE SHASHANK | 5 | 20 | 25 |
| 52 | 21H51A05M2 | DIVYA GAUTAM | 5 | 22 | 27 |
| 53 | 21H51A05M3 | GANGASANI SHANKARSHAN | 5 | 16 | 21 |
| 54 | 21H51A05M5 | GUMMIREDDY SAINATH REDDY | 5 | 23 | 28 |
| 55 | 21H51A05M6 | KALLURI THANMAI | 5 | 23 | 28 |
| 56 | 21H51A05M7 | KATRAVATH MANJULA | 5 | 21 | 26 |
| 57 | 21H51A05M8 | MOHAMMED SAMEER ALI | 5 | 21 | 26 |
| 58 | 21H51A05N0 | NANCHARLA SAI AKSHITHA | 5 | 20 | 25 |
| 59 | 21H51A05N2 | OLIGE RANI | 5 | 22 | 27 |
| 60 | 21H51A05N6 | SAKKERLA RAJ KUMAR | 5 | 10 | 15 |
| 61 | 21H51A05N7 | SALENDRA MANOJ KUMAR | 5 | 07 | 12 |
| 62 | 21H51A05N8 | SHAIK JAVED | 5 | 18 | 23 |
| 63 | 21H51A05N9 | SHRIYA MALANI | 5 | 22 | 27 |
| 64 | 21H51A05P1 | VASURI VINAY KUMAR | 5 | 15 | 20 |
| 65 | 21H51A05P3 | VITTAPUR BINNU REDDY | 5 | 21 | 26 |
| 66 | 21H51A05P6 | BANOTHU DALI HIMASRI | 5 | 18 | 23 |
| 67 | 21H51A05Q0 | D GAYATHRI | 5 | 22 | 27 |
| 68 | 21H51A05Q4 | GUDIPUDI DHEERAJ | 5 | 15 | 20 |
| 69 | 21H51A05Q5 | GURRAM SRIKANTH | 5 | 17 | 22 |
| 70 | 21H51A05Q6 | KALVAKUNTA CHANDRASHEKAR | 5 | 17 | 22 |
| 71 | 21H51A05Q7 | KAPU HARSHA VARDAN REDDY | 5 | 23 | 28 |
| 72 | 21H51A05Q8 | KOTTE MOUNIKA | 5 | 15 | 20 |
| 73 | 21H51A05Q9 | MANDA VIGHNESHWARA REDDY | 5 | 14 | 19 |
| 74 | 21H51A05R0 | MANDHUMULA DEEPAK | 5 | 17 | 22 |
| 75 | 21H51A05R1 | PEDDI PRAVALIKA REDDY | 5 | 18 | 23 |
| 76 | 21H51A05R2 | PENDEM YOGITHA | 5 | 17 | 22 |
| 77 | 21H51A05R8 | YESUGARI ADHARSH | 5 | 18 | 23 |

| | |
|---|---|
| Name&Signature of the Faculty : | E.krishnaveni |
| Designation: | (CSE)-Assistent professor |
| Department : | CSE |
| Mobile No : | 8688396393 |

HOD/CSE

# CMR College of Engineering & Technology

(UGC AUTONOMOUS)

Kandlakoya , Medchal Road - 501401

CMR

## Department of Computer Science and Engineering

MID-II MARKS LIST

Class : III B.Tech. I SEM CSE SECTION-A A.Y.2023-24

SUBJECT : Operating System

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 1 | 21H51A0501 | BINGI NITHYASRI | 5 | 15 | 20 |
| 2 | 21H51A0503 | DASI RASHMIKA | 5 | 20 | 25 |
| 3 | 21H51A0505 | GOUNI PAVANI | 5 | 23 | 28 |
| 4 | 21H51A0508 | KOMMU VEERENDAR | 5 | 18 | 23 |
| 5 | 21H51A0514 | MOHAMMED ABDUL SAMEER | 5 | 16 | 21 |
| 6 | 21H51A0515 | MUAAZ MOHAMMED MUNEER | 5 | 17 | 22 |
| 7 | 21H51A0518 | PALTHYA SUMAN | Ab | 19 | 19 |
| 8 | 21H51A0519 | PAPPULA KARTHIK REDDY | Ab | 17 | 17 |
| 9 | 21H51A0520 | POSHETTY VARSHITH | Ab | 18 | 18 |
| 10 | 21H51A0521 | RITESH KUMAR | 5 | 18 | 23 |
| 11 | 21H51A0524 | TEJAVATH VASANTHA | 5 | 22 | 27 |
| 12 | 21H51A0525 | THOTA MAHESHWARI | 5 | 16 | 21 |
| 13 | 21H51A0526 | VEERELLI SAIVENKATA REDDY | Ab |  |  |
| 14 | 21H51A0529 | BELKONI ANVESH | 5 | 18 | 23 |
| 15 | 21H51A0533 | DASARI AJAY KUMAR | 5 | 20 | 25 |
| 16 | 21H51A0537 | GANTA NISHAL | 5 | 16 | 21 |
| 17 | 21H51A0540 | KOMMANABOINA ANUSHA | 5 | 18 | 23 |
| 18 | 21H51A0541 | LOKOTI SRICHARAN | 5 | 11 | 16 |
| 19 | 21H51A0542 | M KAVYA | 5 | 19 | 24 |
| 20 | 21H51A0544 | OJAS RAKESH GARPALLIWAR | Ab | 13 | 13 |
| 21 | 21H51A0545 | PEDDINTI SAI VARDHAN | 5 | 12 | 17 |
| 22 | 21H51A0547 | SATVIKA KARUMUDI | 5 | 19 | 24 |
| 23 | 21H51A0549 | THAMMISHETTY SHASHANK | 5 | 14 | 19 |
| 24 | 21H51A0550 | TUMMALA VENGAL RAYUDU | Ab | 10 | 10 |
| 25 | 21H51A0551 | UMMEDA SHIVA SAI KRISHNA | Ab | 4 | 04 |
| 26 | 21H51A0552 | VEMULA PRIYA PRAMIDHA | 5 | 22 | 27 |
| 27 | 21H51A0554 | ABHISHEK KUMAR SINGH | 5 | 24 | 29 |
| 28 | 21H51A0555 | ALETI ASHWITHA REDDY | 5 | 14 | 19 |
| 29 | 21H51A0556 | BATTU VICTOR DINAKAR BABU | 5 | 12 | 17 |
| 30 | 21H51A0559 | GANDRATH SRI YAGNA | Ab | 19 | 19 |
| 31 | 21H51A0562 | JOGU TARUN TEJA | Ab | 16 | 16 |
| 32 | 21H51A0563 | KARRA VINAY REDDY | 5 | 13 | 18 |
| 33 | 21H51A0569 | MOHAMMAD FERIA | 5 | 18 | 23 |
| 34 | 21H51A0570 | NAGULAPALLY UDAYKIRAN | 5 | 15 | 20 |
| 35 | 21H51A0572 | SARVADEY ZANETA | 5 | 20 | 25 |

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 36 | 21H51A0573 | SATHYARAM DHANA LAKSHMI | 5 | 19 | 24 |
| 37 | 21H51A0574 | SHA SOPNIL JAIN | 5 | 22 | 27 |
| 38 | 21H51A0578 | VUPPALA SHLAGHA | 5 | 16 | 21 |
| 39 | 21H51A0582 | JYOTHI BALAJI | Ab | 05 | 05 |
| 40 | 21H51A0583 | K RITIKA REDDY | 5 | 16 | 21 |
| 41 | 21H51A0584 | KOPPULA VENKATA SAI NANDINI | 5 | 19 | 24 |
| 42 | 21H51A0586 | M GANESH | 5 | 16 | 21 |
| 43 | 21H51A0592 | NENAVATH SRAVANI RATHOD | 5 | 22 | 27 |
| 44 | 21H51A0595 | PAVAN KUMAR | Ab | 18 | 18 |
| 45 | 21H51A0597 | ROSHAN TALARI | Ab | 19 | 19 |
| 46 | 21H51A0598 | S VARUN | 5 | 16 | 21 |
| 47 | 21H51A05A5 | AILENI SATHWIK | Ab | 18 | 18 |
| 48 | 21H51A05A6 | AKURATHI RITHVIK SESHAGIRI | 5 | 16 | 21 |
| 49 | 21H51A05A9 | BIJJAM SOUMIKA | 5 | 17 | 22 |
| 50 | 21H51A05B0 | BODA ASHOK | Ab | 15 | 15 |
| 51 | 21H51A05B6 | GOLLAPUDI NITHIN | 5 | 11 | 16 |
| 52 | 21H51A05C1 | NALLAKULA KIRANKUMAR | 5 | 14 | 19 |
| 53 | 21H51A05C4 | RITVIK PRATHAPANI | 5 | 9 | 14 |
| 54 | 22H55A0501 | AILLURI AMARDEEP REDDY | 5 | 20 | 25 |
| 55 | 22H55A0502 | BAIROJU SINDHU | 5 | 20 | 25 |
| 56 | 22H55A0503 | BODA AVINASH | 5 | 20 | 25 |
| 57 | 22H55A0504 | BODA RAHUL SAI KIRAN | Ab | 13 | 13 |
| 58 | 22H55A0505 | CHAKILAM BHARAT KUMAR | 5 | 25 | 30 |
| 59 | 22H55A0506 | ERLA VENU | 5 | 22 | 27 |
| 60 | 22H55A0507 | JONNALA SOWMYA | 5 | 21 | 26 |
| 61 | 22H55A0508 | KALE PRABHAS | 5 | 21 | 26 |
| 62 | 22H55A0509 | KATKAM MANASWINI | 5 | 23 | 28 |
| 63 | 22H55A0510 | KODIDALA KOMALI | 5 | 20 | 25 |
| 64 | 22H55A0511 | KONDA MAHIMASRI | 5 | 18 | 23 |
| 65 | 22H55A0512 | KONDAPARTHI MANJEERA | 5 | 23 | 28 |
| 66 | 22H55A0513 | KUMMARI RAJESH | Ab | 20 | 20 |
| 67 | 22H55A0514 | KURUMULA LOKESH | 5 | 16 | 21 |

| | |
|---|---|
| Name&Signature of the Faculty : | G. Saidulu |
| Department : | CSE |
| Mobile No : | 9603131030 |

HOD/CSE

# CMR College of Engineering & Technology

(UGC AUTONOMOUS)

Kandlakoya , Medchal Road - 501401

## Department of Computer Science and Engineering

MID-II MARKS LIST

Class : III B.Tech. I SEM CSE SECTION-B A.Y.2023-24

SUBJECT : Operating Systems

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 1 | 21H51A0502 | DASARI HARINI | 5 | 17 | 22 |
| 2 | 21H51A0504 | GAJULAPALLE SREE LAKSHMI | 5 | 22 | 27 |
| 3 | 21H51A0506 | J AKANSH | 5 | 22 | 27 |
| 4 | 21H51A0507 | K ZAYD AHMED | 5 | 15 | 20 |
| 5 | 21H51A0509 | KURAPATI ESHWAR | 5 | 19 | 24 |
| 6 | 21H51A0510 | LAVANGU VAISHNAVI | 5 | 20 | 25 |
| 7 | 21H51A0511 | MAHANTHI SAI MANYA SRI | 5 | 20 | 25 |
| 8 | 21H51A0512 | MANAS CHHATWAL | 5 | 20 | 25 |
| 9 | 21H51A0513 | MANGINA SRI VENKATA SAI | 5 | 18 | 23 |
| 10 | 21H51A0516 | NAGIREDDY ANVITHA | 5 | 21 | 26 |
| 11 | 21H51A0517 | PADALA ANIL KUMAR | 5 | 14 | 19 |
| 12 | 21H51A0522 | SHREYASH SANJEEV KUMAR | 5 | 20 | 25 |
| 13 | 21H51A0523 | SIDDAMSHETTI SUMITH | 5 | 19 | 24 |
| 14 | 21H51A0527 | AKSHAT KALA | 5 | 20 | 25 |
| 15 | 21H51A0528 | ALAVALA KAVYA | 5 | 21 | 26 |
| 16 | 21H51A0530 | BENKI JYOTHIKA | 5 | 18 | 23 |
| 17 | 21H51A0531 | BENKI VARSHITHA RANI | 5 | 21 | 26 |
| 18 | 21H51A0532 | BOLLU HARI CHARHAN | 5 | 20 | 25 |
| 19 | 21H51A0534 | DAVULURI SAI SUJAN | 5 | 20. | 25 |
| 20 | 21H51A0535 | DESHAPATHI SAHITHI | 5 | 23 | 28 |
| 21 | 21H51A0536 | DHULIPALLA VENKATA SAI SIVA | 5 | 21 | 26 |
| 22 | 21H51A0539 | KOLAN SAHASRA REDDY | 5 | 22 | 27 |
| 23 | 21H51A0543 | MANGA TARAKA RATNA YOSHITH | 5 | 19 | 24 |
| 24 | 21H51A0546 | SAPNA TIWARI | 5 | 19 | 24 |
| 25 | 21H51A0548 | THAKUR ABHINAV SINGH | 5 | 22 | 27 |
| 26 | 21H51A0553 | ABBULA VINUTHNA | 5 | 20 | 25 |
| 27 | 21H51A0557 | BUCHENELLI NIKHILESH REDDY | 5 | 20 | 25 |
| 28 | 21H51A0558 | DANDA VENKATA SATHWIK REDDY | 5 | 17 | 22 |
| 29 | 21H51A0560 | GORINTA RAHULU | 5 | 16 | 21 |
| 30 | 21H51A0561 | GUNREDDY AKSHITH REDDY | 5 | 21 | 26 |
| 31 | 21H51A0564 | KODURU PRANATHI | 5 | 20 | 25 |
| 32 | 21H51A0565 | KONDA VISHAL GOUD | 5 | 19 | 24 |
| 33 | 21H51A0566 | KURAKULA SHAILESH | 5 | 17 | 22 |
| 34 | 21H51A0567 | MADIRA SAI RISHITHA | 5 | 23 | 28 |
| 35 | 21H51A0568 | MANURI CHANDU BABU | 5 | 17 | 22 |

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 36 | 21H51A0571 | NIMMALA SAI | 5 | 17 | 22 |
| 37 | 21H51A0575 | TUDURU SATHWIK | 5 | 18 | 23 |
| 38 | 21H51A0576 | U NAGA MANASWINI | 5 | 15 | 20 |
| 39 | 21H51A0577 | VARLA RAMAKRISHNA REDDY | 5 | 20 | 25 |
| 40 | 21H51A0579 | AMBATI ROHITH RAJU | 5 | 18 | 23 |
| 41 | 21H51A0580 | BAIRA ANUSHA | 5 | 23 | 28 |
| 42 | 21H51A0581 | GUNNALA AKHILA | 5 | 23 | 28 |
| 43 | 21H51A0585 | KUDUMULA ANVESH REDDY | 5 | 20 | 25 |
| 44 | 21H51A0587 | MANDALOJU VASANTH KUMAR | 5 | 21 | 26 |
| 45 | 21H51A0588 | MOHAMMAD ABDUL KALAM | 5 | 15 | 20 |
| 46 | 21H51A0589 | MOHAMMED MUDASSIR ALI | 5 | 22 | 27 |
| 47 | 21H51A0590 | NALABOLU MOUNIKA | 5 | 21 | 26 |
| 48 | 21H51A0591 | NAMPALLY SIDDHARTHA | 5 | 21 | 26 |
| 49 | 21H51A0593 | PAMULA BEULAH SUPRAGNYA | 5 | 21 | 26 |
| 50 | 21H51A0594 | PANCHAGNULA VINUTNA | 5 | 23 | 28 |
| 51 | 21H51A0596 | RAGE DAMODHAR | 5 | 21 | 26 |
| 52 | 21H51A0599 | SAI KIRAN B L S · | ~~5~~ 5 | 16 | 21 |
| 53 | 21H51A05A0 | SHESHAVAMATAM SUCHIT PAUL | 5 | 18 | 23 |
| 54 | 21H51A05A1 | TEEGALA BHANU TEJA REDDY | 5 | 23 | 28 |
| 55 | 21H51A05A2 | VADDI RISHIKA | 5 | 23 | 28 |
| 56 | 21H51A05A3 | YADDANAPUDI VISHNU SRIVATSAVA | 5 | 22 | 27 |
| 57 | 21H51A05A4 | YELDI ARUN | 5 | 19 | 24 |
| 58 | 21H51A05A7 | BAJRANG HARSH SINGH | 5 | 20 | 25 |
| 59 | 21H51A05A8 | BASAR SHYAM SUNDER RAO | 5 | 22 | 27 |
| 60 | 21H51A05B1 | BUNNI SHARANYA | 5 | 24 | 29 |
| 61 | 21H51A05B2 | C J VISHNU PRAKASH | 5 | 18 | 23 |
| 62 | 21H51A05B3 | CHIMMULA SHIVA PRASAD REDDY | 5 | 21 | 26 |
| 63 | 21H51A05B4 | DOLLA RENUKA | 5 | 21 | 26 |
| 64 | 21H51A05B5 | ERUKULA RAJASREE | 5 | 23 | 28 |
| 65 | 21H51A05B7 | HARIKA REDDY GANTA | 5 | 22 | 27 |
| 66 | 21H51A05B8 | INDUPALLI SHARONSUDHA | 5 | 22 | 27 |
| 67 | 21H51A05B9 | MADULAPURAM SAI YASHWANTH RAJ | 5 | 19 | 24 |
| 68 | 21H51A05C0 | MALLELA SINDHUJA | 5 | 22 | 27 |
| 69 | 21H51A05C2 | RANGU ABHINAV | 5 | 20 | 25 |
| 70 | 21H51A05C3 | RAYABARAPU CHATHURYA | 5 | 19 | 24 |
| 71 | 21H51A05C5 | SEGU JAYA BALA HARSHAVARDHAN | 5 | 22 | 27 |
| 72 | 21H51A05C8 | THATIKONDA AKHILA | 5 | 22 | 27 |
| 73 | 21H51A05C9 | VAKALA KAVYA SAI SUMA SRI | 5 | 21 | 26 |
| 74 | 21H51A05D1 | ANUJ KUMAR | 5 | 21 | 26 |
| 75 | 21H51A05D2 | BACHAWAR VINITHA | 5 | 23 | 28 |

| | |
|---|---|
| Name&Signature of the Faculty | Dr.G.RAVIKUMAR |
| Department : | C.S.E |
| Mobile No : | 9849037283 |

HOD/CSE

# CMR College of Engineering & Technology

(UGC AUTONOMOUS)

Kandlakoya , Medchal Road - 501401

## Department of Computer Science and Engineering

### MID-II MARKS LIST

Class : III B.Tech. I SEM CSE    SECTION-C    A.Y.2023-24

SUBJECT : Operating System

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 1 | 21H51A05C6 | SOMU KOTESWARA REDDY | 5 | 8 | 13 |
| 2 | 21H51A05C7 | SUNKAPAKA JOHN | 5 | 11 | 16 |
| 3 | 21H51A05D0 | VALLAMKONDA POOJITHA | 5 | 11 | 16 |
| 4 | 21H51A05D3 | BASHAM RAJU | 5 | 17 | 22 |
| 5 | 21H51A05D4 | BUSSA TEJASWINI | 5 | 08 | 13 |
| 6 | 21H51A05D5 | DADE DINISHA | 5 | 06 | 11 |
| 7 | 21H51A05D6 | DEEKONDA SAKETH | 0 | 03 | 03 |
| 8 | 21H51A05E3 | MANCHI AKSHAYA | 5 | 12 | 17 |
| 9 | 21H51A05E4 | MOHAMMAD ARSHAD NIZAMI | 5 | 10 | 15 |
| 10 | 21H51A05F1 | P Y GEETHA MADHURI | 5 | 13 | 18 |
| 11 | 21H51A05F3 | SHAIK ILLIYAZ | 0 | 06 | 06 |
| 12 | 21H51A05F5 | TUSHAR PUNIA | 5 | 07 | 12 |
| 13 | 21H51A05F8 | DODDI SAI PHANI HARI CHANDANA | 5 | 15 | 20 |
| 14 | 21H51A05F9 | GADUGULA KALYANI | 5 | 08 | 13 |
| 15 | 21H51A05G2 | IYLA SNEHARIKA | 5 | 12 | 17 |
| 16 | 21H51A05G5 | KANUGO NESHIT RAJ | 5 | 11 | 16 |
| 17 | 21H51A05H2 | PODDUTURI NITHIN REDDY | 5 | 12 | 17 |
| 18 | 21H51A05H8 | TADEM RAVITEJA | 5 | 15 | 20 |
| 19 | 21H51A05J8 | GUNTHAPALLI MALINI | 5 | 10 | 15 |
| 20 | 21H51A05J9 | GURRAM KRISHNA PRASANTH | 5 | 12 | 17 |
| 21 | 21H51A05K3 | KODIGANTI SAI KISHORE | 0 | 01 | 01 |
| 22 | 21H51A05K8 | SEELAMSETTY PRASANNA GAYATHRI | 5 | 12 | 17 |
| 23 | 21H51A05L1 | SRIRAM NAGARAJU | 5 | 09 | 14 |
| 24 | 21H51A05L7 | YALLA TEJASWIK REDDY | 5 | 11 | 16 |
| 25 | 21H51A05L8 | BEHARA SURAJ | 5 | 11 | 16 |
| 26 | 21H51A05M0 | CHILUKA SAI KARTHIK | 5 | 10 | 15 |
| 27 | 21H51A05M1 | DAMARLA HEMAVATHI | 5 | 03 | 08 |
| 28 | 21H51A05M4 | GIRAVENA ARYA | 5 | 03 | 08 |
| 29 | 21H51A05M9 | MOKIRALA JHANSI | 5 | ~~09~~ 12 | ~~14~~ 17 |
| 30 | 21H51A05N1 | NEELA SAI ADITYA | 5 | 10 | 15 |
| 31 | 21H51A05N3 | POTRU SAI NITISH | 5 | 10 | 15 |
| 32 | 21H51A05N4 | PRAHARSHITHA SURAGONI | 5 | 15 | 20 |
| 33 | 21H51A05N5 | PULI PRANEETH GOUD | 5 | 06 | 11 |
| 34 | 21H51A05P0 | TALOORI PRABHU KIRAN | 5 | 06 | 11 |
| 35 | 21H51A05P2 | VAVILLA RAVITEJA | 5 | 23 | 28 |

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 36 | 21H51A05P4 | ALLURI SAI SATHWIK REDDY | 5 | 15 | 20 |
| 37 | 21H51A05P5 | ANDE AJAY | 5 | 10 | 15 |
| 38 | 21H51A05P7 | BESTHA NANDA KISHORE | 5 | 16 | 21 |
| 39 | 21H51A05P8 | CHAVATAPALLI MUKUNDA SRI HASINI | 5 | 11 | 16 |
| 40 | 21H51A05P9 | CHEPYALA SATHWIK REDDY | 0 | 06 | 06 |
| 41 | 21H51A05Q1 | DAGGULA PRASHANTH | 5 | 14 | 19 |
| 42 | 21H51A05Q2 | GAJULA NAVANEETH | 5 | 07 | 12 |
| 43 | 21H51A05Q3 | GUDAPATI NITHIN KUMAR | 5 | 06 | 11 |
| 44 | 21H51A05R3 | PINAPATI ABHISHEK | 5 | 05 | 10 |
| 45 | 21H51A05R4 | RACHAMALLA SAI UJITHA REDDY | 5 | 00 | 05 |
| 46 | 21H51A05R5 | SATTU RAKESH | 5 | 09 | 14 |
| 47 | 21H51A05R6 | SHREYA M | 5 | 02 | 07 |
| 48 | 21H51A05R7 | YERAVELLI RUCHITHA | 5 | 13 | 18 |
| 49 | 22H55A0515 | M. SAI RANJITH REDDY | 5 | 14 | 17 |
| 50 | 22H55A0516 | MAHATHI DESAI | 0 | 13 | 13 |
| 51 | 22H55A0517 | MD TOWHEED | 5 | 11 | 16 |
| 52 | 22H55A0518 | MOHAMMED HANEF | 5 | 01 | 06 |
| 53 | 22H55A0519 | NAGARAM SHIVA CHAND | 5 | 07 ~~AB~~ | ~~05~~ 12 |
| 54 | 22H55A0520 | NARGE CHARANETEJA | 5 | 10 | 15 |
| 55 | 22H55A0521 | NEELAM RAMYA SARI | 5 | 15 | 20 |
| 56 | 22H55A0522 | PANDAV SONIA | 5 | 21 | 26 |
| 57 | 22H55A0523 | PATHLAVATH SUNITHA | 5 | 18 | 23 |
| 58 | 22H55A0524 | POTTIPALLY DEEPIKA | 5 | 19 | 24 |
| 59 | 22H55A0525 | PULIGANTI MAHENDAR | 5 | 17 | 22 |
| 60 | 22H55A0526 | SARDESHI PRAVEEN KUMAR | 5 | 19 | 24 |
| 61 | 22H55A0527 | VISLAVATH ANITHA | 5 | 13 | 18 |

Name&Signature of the Faculty : P. Sravanthi

Department : CSE

Mobile No : 9989448869

HOD/CSE

# CMR College of Engineering & Technology

(UGC AUTONOMOUS)

Kandlakoya , Medchal Road - 501401

CMR

Department of Computer Science and Engineering

MID-II MARKS LIST

Class : III B.Tech. I SEM CSE SECTION-D

A.Y.2023-24

SUBJECT : operating system (A30516)

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 1 | 21H51A05D7 | DHUDURI SATHVIKA | 5 | 24 | 29 |
| 2 | 21H51A05D8 | GAMPALA SRI DURGA PRABHATH | 5 | 16 | 21 |
| 3 | 21H51A05D9 | GUNDLA VAMSHIDHAR | 4 | 19 | 23 |
| 4 | 21H51A05E0 | KASANAGOTTU AMULYA | 5 | 21 | 26 |
| 5 | 21H51A05E1 | KOTHA VAISHNAVI | 5 | 16 | 21 |
| 6 | 21H51A05E2 | KUMBALA ABHILASH REDDY | 5 | 19 | 24 |
| 7 | 21H51A05E6 | NAKKALA KEERTHANA | 5 | 20 | 25 |
| 8 | 21H51A05E7 | NEELAM BHARATH KUMAR | 5 | 17 | 22 |
| 9 | 21H51A05E8 | NEERUDI HARIPRASAD | 5 | 20 | 25 |
| 10 | 21H51A05E9 | ODURI VEERAMANIKANTA | 5 | 16 | 21 |
| 11 | 21H51A05F0 | OM GUPTA | 5 | 22 | 27 |
| 12 | 21H51A05F2 | ROHAN SACHIN RAKHE | 5 | 18 | 23 |
| 13 | 21H51A05F4 | SHAIK TASNIM | 5 | 22 | 27 |
| 14 | 21H51A05F6 | YARRAMSETTI MADHU VENKATA | 5 | 22 | 27 |
| 15 | 21H51A05F7 | BABBI THAPA | 5 | 18 | 23 |
| 16 | 21H51A05G0 | GUDIPALLY SAI SANJAY | 5 | 19 | 24 |
| 17 | 21H51A05G1 | GUNNA VINAY KUMAR REDDY | 5 | 21 | 26 |
| 18 | 21H51A05G3 | K SRI HARINI | 5 | 23 | 28 |
| 19 | 21H51A05G4 | KANDI SWETHA | 5 | 20 | 25 |
| 20 | 21H51A05G6 | KHANDESH THANU SRI | 5 | 14 | 19 |
| 21 | 21H51A05G7 | MAMIDI VENU GOPAL | 5 | 18 | 23 |
| 22 | 21H51A05G8 | MARAGONI KARTHIKEYA | 5 | 24 | 29 |
| 23 | 21H51A05G9 | NALIMELA JITHIN REDDY | 5 | 14 | 19 |
| 24 | 21H51A05H1 | PATRAYADI RAVI | 5 | 18 | 23 |
| 25 | 21H51A05H3 | POTHARAJU SAI KIRAN | 5 | 17 | 22 |
| 26 | 21H51A05H5 | SHERIKAR RAHUL | 5 | 22 | 27 |
| 27 | 21H51A05H6 | SOMARAJUPALLI THEJASWI | 5 | 19 | 24 |
| 28 | 21H51A05H7 | SUDAM SHIVA | 5 | 19 | 24 |
| 29 | 21H51A05H9 | THALLAM GEETHAN | 5 | 19 | 24 |
| 30 | 21H51A05J0 | TODUPUNURI SHAI PRIYA | 5 | 19 | 24 |
| 31 | 21H51A05J1 | TUMMALA SAHITH | 5 | 18 | 23 |
| 32 | 21H51A05J2 | VIJAYAGIRI AMULYA | 5 | 24 | 29 |
| 33 | 21H51A05J3 | ABHAY PRATAP SINGH | 5 | 23 | 28 |
| 34 | 21H51A05J4 | AYEMON ZEBA | 5 | 19 | 24 |
| 35 | 21H51A05J5 | BONDALA SRINATH | 5 | 17 | 22 |

| S.No | Roll Number | Name of the Candidate | Assignment (5M) | MID Marks (25 M) | Total (30 M) |
|---|---|---|---|---|---|
| 36 | 21H51A05J6 | DODDAPANENI MEGHAN CHOWDARY | 4 | 16 | 20 |
| 37 | 21H51A05J7 | GORANTI SANTHU SATHWIK | 5 | 19 | 24 |
| 38 | 21H51A05K0 | KACHIREDDY JAYASREE | 5 | 21 | 26 |
| 39 | 21H51A05K1 | KAJA SANJEEV KUMAR | 5 | 22 | 27 |
| 40 | 21H51A05K2 | KANTU ANANTHKUMAR | 5 | 20 | 25 |
| 41 | 21H51A05K4 | KONDETI VIKRAMREDDY | 5 | 22 | 27 |
| 42 | 21H51A05K5 | KRITIKA KHATRI | 5 | 18 | 23 |
| 43 | 21H51A05K6 | NITYANANDAYYA MATHPATHI | 5 | 16 | 21 |
| 44 | 21H51A05K9 | SHANIGALA VISHNU | 5 | 21 | 26 |
| 45 | 21H51A05L0 | SINDEY ABHIGNA | 5 | 19 | 24 |
| 46 | 21H51A05L2 | SUMESH | 4 | 22 | 26 |
| 47 | 21H51A05L3 | TANNIRU MAHESH | 4 | 21 | 25 |
| 48 | 21H51A05L4 | TUSYAA SREERALA | 5 | 21 | 26 |
| 49 | 21H51A05L5 | UNI SAILESH | 5 | 23 | 28 |
| 50 | 21H51A05L6 | VAGUAMRI SRINANDHAN | 5 | 22 | 27 |
| 51 | 21H51A05L9 | BHAKE SHASHANK | 5 | 24 | 29 |
| 52 | 21H51A05M2 | DIVYA GAUTAM | 5 | 18 | 23 |
| 53 | 21H51A05M3 | GANGASANI SHANKARSHAN | 5 | 24 | 29 |
| 54 | 21H51A05M5 | GUMMIREDDY SAINATH REDDY | 5 | 24 | 29 |
| 55 | 21H51A05M6 | KALLURI THANMAI | 5 | 17 | 22 |
| 56 | 21H51A05M7 | KATRAVATH MANJULA | 5 | 20 | 25 |
| 57 | 21H51A05M8 | MOHAMMED SAMEER ALI | 5 | 19 | 24 |
| 58 | 21H51A05N0 | NANCHARLA SAI AKSHITHA | 5 | 16 | 21 |
| 59 | 21H51A05N2 | OLIGE RANI | 5 | 19 | 24 |
| 60 | 21H51A05N6 | SAKKERLA RAJ KUMAR | 5 | 18 | 23 |
| 61 | 21H51A05N7 | SALENDRA MANOJ KUMAR | 5 | 19 | 24 |
| 62 | 21H51A05N8 | SHAIK JAVED | 5 | ~~19~~ 20 | 25 ~~26~~ |
| 63 | 21H51A05N9 | SHRIYA MALANI | 5 | 19 | 24 |
| 64 | 21H51A05P1 | VASURI VINAY KUMAR | 5 | 17 | 22 |
| 65 | 21H51A05P3 | VITTAPUR BINNU REDDY | 5 | 19 | 24 |
| 66 | 21H51A05P6 | BANOTHU DALI HIMASRI | 5 | 18 | 23 |
| 67 | 21H51A05Q0 | D GAYATHRI | 5 | 19 | 24 |
| 68 | 21H51A05Q4 | GUDIPUDI DHEERAJ | 5 | 20 | 25 |
| 69 | 21H51A05Q5 | GURRAM SRIKANTH | 5 | 21 | 26 |
| 70 | 21H51A05Q6 | KALVAKUNTA CHANDRASHEKAR | 5 | 19 | 24 |
| 71 | 21H51A05Q7 | KAPU HARSHA VARDAN REDDY | 5 | 20 | 25 |
| 72 | 21H51A05Q8 | KOTTE MOUNIKA | 5 | 15 | 20 |
| 73 | 21H51A05Q9 | MANDA VIGHNESHWARA REDDY | 5 | 17 | 22 |
| 74 | 21H51A05R0 | MANDHUMULA DEEPAK | 5 | 22 | 27 |
| 75 | 21H51A05R1 | PEDDI PRAVALIKA REDDY | 5 | 20 | 25 |
| 76 | 21H51A05R2 | PENDEM YOGITHA | 5 | 22 | 27 |
| 77 | 21H51A05R8 | YESUGARI ADHARSH | 5 | 20 | 25 |

Name&Signature of the Faculty : E. krishnaveni 3/01

Designation: Asst. proof

Department : CSE

Mobile No : 8688396393

HOD/CSE

Hall Ticket No. □□□□□□□□□□

SET-2

Question Paper Code: A30516

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS)

B.Tech V Semester Mid-I Examinations October -2023

(Regulation: CMRCET-R18)

Subject Name: Operating Systems Ti

Date:02-11-2023 Branch: CSE

## PART A

Answer all FIVE questions (Compulsory)

Each question carries TWO marks. 5x2=10M

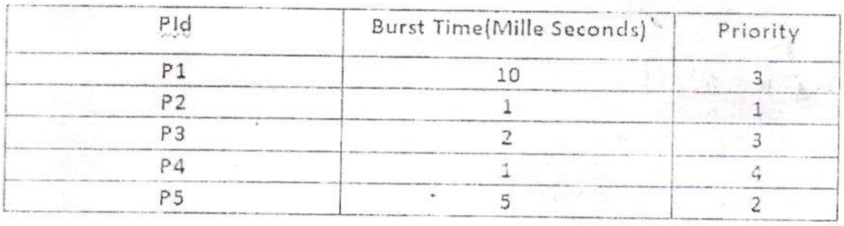
| Question Number | Question Format | CO | B.T Level |
|---|---|---|---|
| 1 | Distinguish between Symmetric and Asymmetric Multi-Processor Systems? | 1 | 2 |
| 2 | Define Distributed and Time Shared Systems. | 1 | 1 |
| 3 | What is Scheduling? Discuss about Scheduling Criteria? | 2 | 1 |
| 4 | Discuss Context Switch? | 2 | 2 |
| 5 | Draw and Explain about Resource Allocation Graph? | 3 | 4 |

## PART B

Answer ALL questions.

Each question carries FIVE Marks. 3x5=15M

| Question Number | Question Format | CO | B.T Level |
|---|---|---|---|
| 6A | Define an operating system? List and explain services provided by an operating system? | 1 | 1 |
|  | OR |  |  |
| 6B | What is a system call? Explain various types of System Calls? | 1 | 1 |
| 7A | Explain Shortest Job First CPU Scheduling Algorithm. Consider the following set of processes, with the length of the CPU Burst given in Milliseconds. | 2 | 4 |



| PId | Burst Time(Mille Seconds) | Priority |
|---|---|---|
| P1 | 10 | 3 |
| P2 | 1 | 1 |
| P3 | 2 | 3 |
| P4 | 1 | 4 |
| P5 | 5 | 2 |

i. Draw Gantt charts For FCFS, Non-Preemptive SJF, Non-Preemptive Priority and Round Robin (Time Quantum=1).

ii. Calculate Average Turnaround time and Average Waiting Time for above scheduling algorithms.

OR

7B Explain Various Operations of Processes.

8A Explain Inter Process Communication Mechanism? Discuss Shared Memory Model and Message Passing Model? 2 4

2 4

OR

8B Explain deadlock avoidance using banker's algorithm with the following 3 4

a) Data Structure b) Safety Algorithm
c) Resource Request Algorithm

| | Allocation | | | | Max | | | | Available | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | A | B | C | D | A | B | C | D | A | B | C | D |
| P0 | 0 | 0 | 1 | 2 | 0 | 0 | 1 | 2 | 1 | 5 | 2 | 0 |
| P1 | 1 | 0 | 0 | 0 | 1 | 7 | 5 | 0 | | | | |
| P2 | 1 | 3 | 5 | 4 | 2 | 3 | 5 | 6 | | | | |
| P3 | 0 | 6 | 3 | 2 | 0 | 6 | 5 | 2 | | | | |
| P4 | 0 | 0 | 1 | 4 | 0 | 6 | 5 | 6 | | | | |

i)Find the needed resources for every process?
ii)Is the system is safe state or not?

**END**

90

Subject Name: Operating Systems Branch: CSE

Subject Code: A30516

Mid-I Examinations B. Tech IV Sem

## PART-A

1A)

| S. No | Symmetric Multiprocessing | Asymmetric Multiprocessing. |
|---|---|---|
| 1. | In Symmetric Multiprocessing all the processors are treated equally. | In Assymmetric multiprocessing the processors are not treated equally. |
| 2. | Tasks of the operating system are done on individual processor. | Tasks of the operating System are done by master processor. |
| 3. | Symmetric multiprocessing systems are Complex to design. | Asymmetric multiprocessing systems are easier to design. |

2A) Distributed Systems:- A Distributed operating System refers to a model in which applications run on multiple interconnected Computers, offering enhanced Communication and integration Capabilities Compared to a network operating system.

Time Sharing Systems:- Time-Sharing enables many people, located at Various terminals, to use a particular Computer system at the Same time. Multitasking or Time-Sharing systems is a logical extension of multiprogramming.

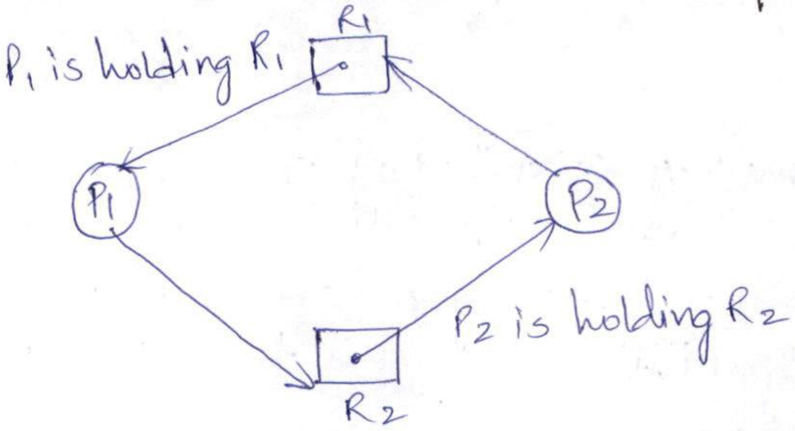
3A) Scheduling is a process of allowing one process to use the CPU resources, Keeping on hold the execution of another Process due to the unavailability of resources CPU.

Scheduling Criteria:-

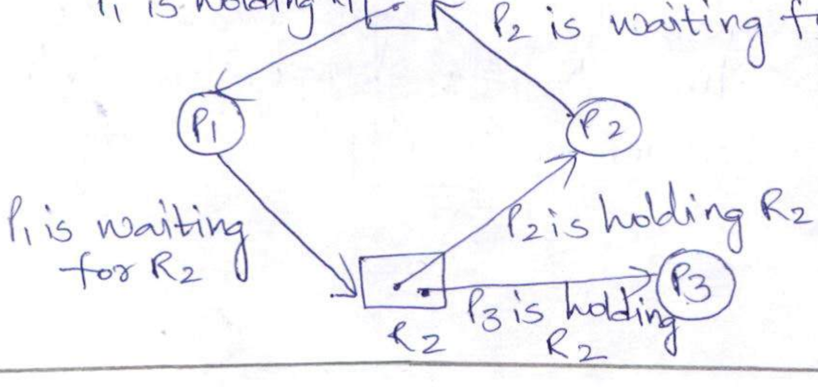
1) CPU Utilization
2) Throughoutput
3) Waiting Time
4) Response Time
5) Turnaround Time.

4A) A Context Switch is a procedure that a Computer's CPU (Central processing Unit) follows to change from one task (or process) to another while ensuring that the tasks do not conflict. Effective Context Switching is Critical if a Computer is to provide user-friendly multitasking.

5A) Single Instance Resource Allocation Graph:-



Multi-Instances Resource Allocation Graph:-



# PART-B

6A) Operating System:- OS lies in the Category of System Software. It basically manages all the resources of the Computer. An OS acts as an interface between Software and different parts of the Computer hardware.

Operating System Services:-

① User Interface :- 3types
a) CLI (Command line Interface)
b) Batch Interface
c) GUI (Graphical User Interface)

② Program Execution:- The program must end with its execution either normally or abnormally.

③ I/o operations:- A running program may require I/o which may involve a file or an I/o device.

④ File System Manipulation:- The programs need to read and write the files and directives.

⑤ Communication:- Communication may be implemented via shared memory and message passing in which Packets of information are moved between process by OS.

⑥ Error Detection:- for Each type of error the OS must take the appropriate action to ensure Correct and Consistent Computing.

⑦ Resource Allocation:- When there are multiple users or multiple jobs running at the Same time resources must be allocated to each of them.

⑧ Accounting:- We want to keep track of which users use how much and What kind of Computer resources.

(9) Protection & Security:- Protection involves ensuring that all access to System resources is Controlled. Security involves requiring each user to authenticate himself to the system usually by means of a password to gain access to System resources.

6B) System Call:- System calls are interfaces provisioned by the OS to allow user-level applications to interact with low-level hardware Components and make use of all the Services provided by the Kernel.

* System Calls are essential for every process to interact with the Kernal and leverage the Services provided by it.

Types of System Calls:-

(1) File System Operations:-

* open():- opens a file for reading or writing.
* read():- Reads data from a file. Just after the file is opened through open() System Call, then if some process want to read the data from a file, then it will make a read() system call.
* write():- Writes data to a file. whenever the user makes any kind of modification in a file & Saves it.

(2) Process Control:-

These types of System calls deals with process Creation, termination, process allocation, deallocation etc.

* fork():- Creates a new process (child) by duplicating the current process (parent).
* wait():- The purpose of this call is to ensure that the parent process doesn't proceed further with its exec. until all its child processes have finished their execution.

* exit():- Terminates the current process.

③ Memory Management:-

* These types of System calls deals with memory allocation, deallocation & dynamically changing the size of memory allocated to a process.

* brk():- Changes data segment size for a process in HEAP memory.

* sbrk():- This call is also for Memory Management in heap, it also takes an argument as an Integer Specifying whether to increase or decrease the size.

④ Interprocess Communication (IPC):-

* when two or more process are required to Communicate then Various IPC Mechanism are used by the OS.

* pipe():- Creates a Unidirectional Communication channel between processes.

* Socket():- Creates a network Socket for Communication.

* shmget():- It is short for - 'shared-memory-get'. It allows one or more processes to share a portion of memory and achieve Interprocess Communication.

⑤ Device Management:-

These System Calls are used to interact with Various pheripheral devices attached to Pc.

* Set ConsoleMode():- This Call is made to set the mode of console.

* writeConsole():- It allows us to write data on Console Screen.

* ReadConsole():- It allows us to read data from Console Screen.

7A) i) Gantt charts

a) FCFS :-

| P1 | P2 | P3 | P4 | P5 | |
|---|---|---|---|---|---|
| 0 | 10 | 11 | 13 | 14 | 19 |

b) Non Preemptive SJF :-

| P2 | P4 | P3 | P5 | P1 | |
|---|---|---|---|---|---|
| 0 | 1 | 2 | 4 | 9 | 19 |

c) Non Preemptive Priority :-

| P2 | P5 | P1 | P3 | P4 | |
|---|---|---|---|---|---|
| 0 | 1 | 6 | 16 | 18 | 19 |

d) Round Robin

| P1 | P2 | P3 | P4 | P5 | P1 | P3 | P5 | P1 | P5 | P1 | P5 | P1 | P5 | P1 | P1 | P1 | P1 | P1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |

ii)

| PId | Burst Time | Priority |
|---|---|---|
| P1 | 10 | 3 |
| P2 | 1 | 1 |
| P3 | 2 | 3 |
| P4 | 1 | 4 |
| P5 | 5 | 2 |

[TQ = 1]

| WT | TAT |
|---|---|
| 0 | 10 |
| 10 | 11 |
| 11 | 13 |
| 13 | 14 |
| 14 | 19 |

a) FCFS :-

$$\text{Avg wt} = \frac{0+10+11+13+14}{5} = 9.6$$

$$\text{Avg TAT} = \frac{10+11+13+14+9}{5} = 13.4$$

## b) Non Preemptive SJF

| WT | TAT |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 4 | 9 |
| 9 | 19 |

Average waiting Time :- $\frac{0+1+2+4+9}{5} = 3.2$

Average TurnAround Time :- $\frac{1+2+4+9+19}{5} = 7$

## c) Non Preemptive Priority :-

| WT | TAT |
|---|---|
| 0 | 1 |
| 1 | 6 |
| 6 | 16 |
| 18 | 18 |
| 18 | 19 |

Average waiting Time $= \frac{0+1+6+16+18}{5} = 8.2$

Average TurnAround Time $= \frac{1+6+16+18+19}{5} = 12$

D) Round Robin:-

| WT | TAT |
|---|---|
| 9 | 19 |
| 1 | 2 |
| 5 | 7 |
| 3 | 4 |
| 9 | 14 |

Average Waiting Time $= \frac{9+1+5+3+9}{5} = 5.4$

Average ~~w~~ Turnaround Time $= \frac{19+2+7+4+14}{5} = 9.2$

Therefore

for FCFS:

Average WT = 9.6

Average TAT = 13.4

for Non Preemptive SJF:

Average WT = 3.2

Average TAT = 7

for Non preemptive SJ Priority:

Average WT = 8.2

Average TAT = 12

for Round Robin:

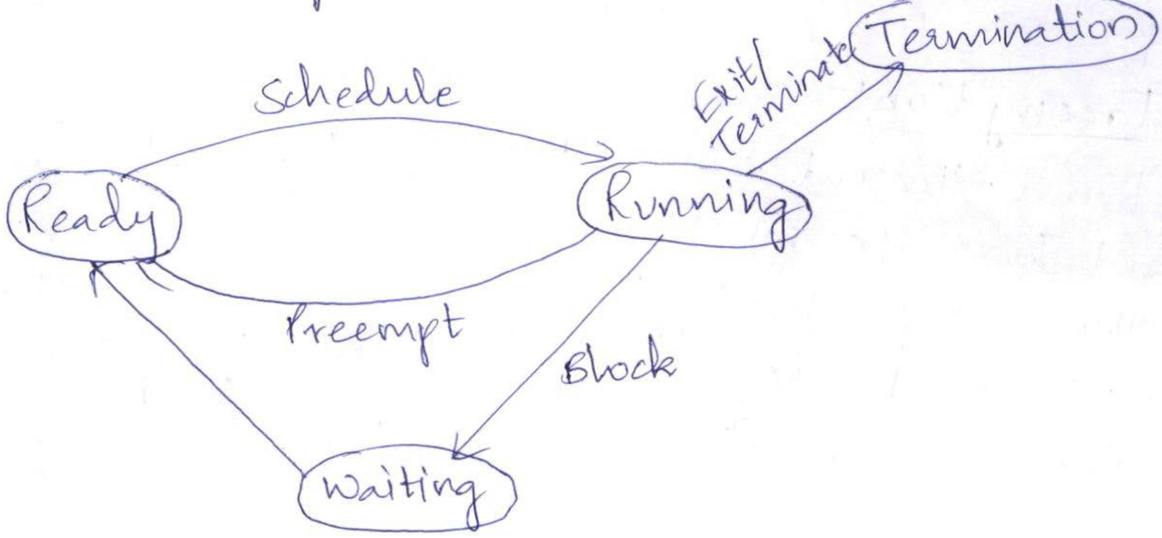
Average WT = 5.4

Average TAT = 9.2

→B) Operations of Processes:-

* A process is an activity of executing a program. Basically, it is a program under execution. Every Process needs Certain resources to Complete its task.

* The execution of a process is a Complex Activity. It involves Various operations.



* Creation:-

* When we Start the Computer, the System Creates Several background processes.

* A user may request to Create a new Process.

* A process Can Create a new process itself while executing.

* The batch system takes initiation of a batch Job.

* Scheduling/dispatching:-

* The event or activity in which the state of the process is changed from ready to run. It means the OS puts the process from the ready state into the running state.

* Dispatching is done by OS when the resources are free or the process has the higher priority than the ongoing process.

* Blocking:-

* When a process invokes an input-output system call that blocks the process, & OS is put in block mode.

* Block Mode is basically a mode where the process waits for Input-output. Hence on the demand of the process itself, the OS blocks the process and dispatches process to processor.

* Preemption:-

* When a timeout occurs that means the process hadn't been terminated in the allocatted time interval and the next process is ready to execute, then the OS Preempts the process. Hence, in Process preemption operation, the operating system puts the process in a 'ready' state.

* Process Termination:-

* The process Completes its execution fully and it indicates to the OS that it has finished.

* The OS itself terminates the process due to Service errors.

* There may be a problem in hardware that terminates the process.

* One process can be terminated by another process.

8A) IPC Mechanism :-

* A process Can be 2 types:-

* Independent process

* Co-operating process.

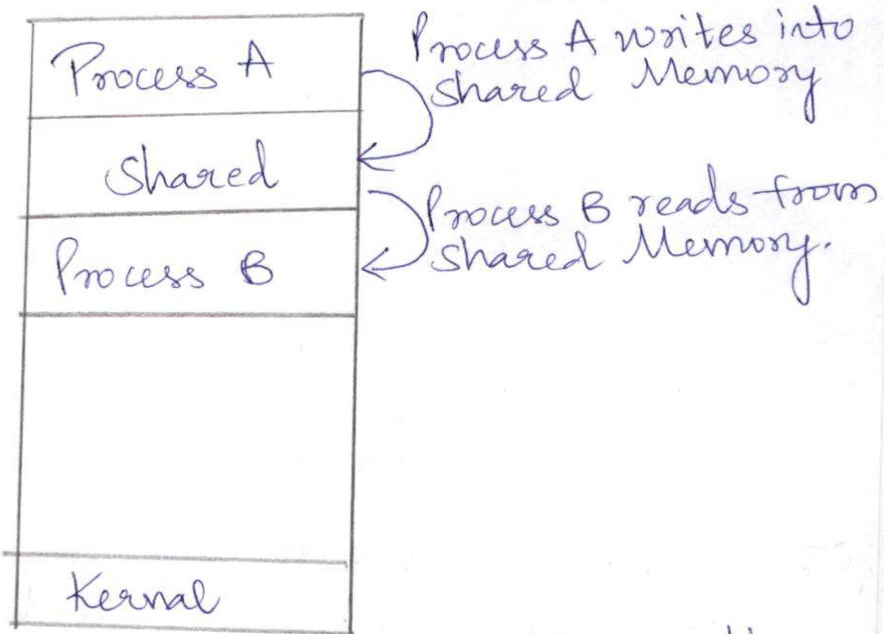
* An Independent Process is not affected by the execution of other process while a Co-operating process Can be affected by other executing processes. Through one Can think that those processes, which are running Independently, will execute very efficiently, in reality, there are many situations when Co-operative nature Can be utilised Increasing speed.

* IPc is a Mechanism that allows processes to communicate with each other and Synchronise their actions. The Communication b/w these processes Can be seen as a method of Co-operation b/w them.

Process Can Communicate through:

1) Shared Memory
2) Message Passing.

## (1) Shared Memory :-


Process A
Shared
Process B
Kernal
Process A writes into Shared Memory
Process B reads from Shared Memory.

* In the Shared Memory system, the Cooperating Process Communicate, to exchange the data with each other. Because of this, the Cooperating processes establish a shared region in their memory. The Processes Share data by reading and writing the data in a shared Segment of the processes.

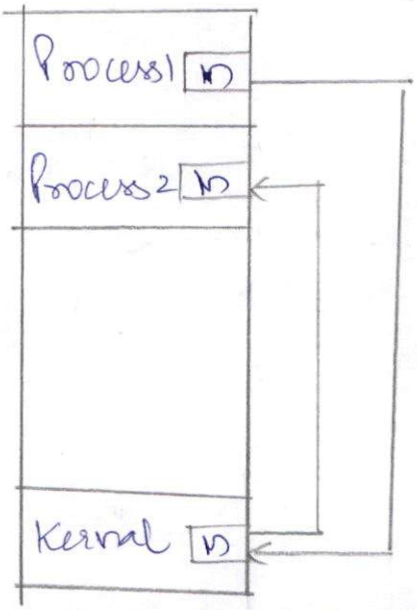
Advantages:-

* Shared Memory is a faster IPC System
* It allows Cooperating processes to access the Same pieces of data Concurrently.
* Users Can perform Multiple tasks at a time.
* Modularity is achieved in Shared Memory system.

Disadvantages:-

* It is of more Complexity.
* Since Shared Memory is open to different cycles, there is a gamble of one interaction getting to or changing information having a place with another interactio

② Message Passing:-



* It is used in distributed environments where the Communicating processes are present on remote machines which are Connected with the help of a network.
* Message passing is a time Consuming process because it is implemented through Kernal.
* It is useful for Sharing Small amounts of data So that Conflicts need not occur.

* If Message Passing the Communication is slower when Compared to shared Memory Technique.

Advantages:-

* Easier to Implement
* Quite tolerant of high Communication latencies.
* Easier to build massively Parallel hardware.
* It is more tolerant of high Communication latencies.

Disadvantages:-

* Programmer has todo everything.
* Connection setup takes time that's why it is slower.
* Data transfer usually requires cooperative operations which can be difficult to achieve.

8B) Data Structures used in Banker's Algorithm:-

→ Available
→ Max
→ Need
→ Allocation.

) Need Matrix:-

Need = Max - Allocation.

| | A | B | C | D |
|---|---|---|---|---|
| P0 | 0 | 0 | 0 | 0 |
| P1 | 0 | 7 | 5 | 0 |
| P2 | 1 | 0 | 0 | 2 |
| P3 | 0 | 0 | 2 | 0 |
| P4 | 0 | 6 | 4 | 0 |

ii) work = <1,5,2,0>

Need ≤ work <A,B,C,D>

work = work + Allocation <3,14,12,12>

$P_0$ <0,0,0,0> ≤ <1,5,2,0> (T)

work = <1,5,3,2>

$P_1$ <0,7,5,0 ~~8~~ ≤ <1,5,3,2> (F)

~~work = <2,8,8~~

$P_2$ <1,0,0,2> ≤ <1,5,3,2> (T)

work = <2,8,8,6>

$P_3$ <0,0,2,0> ≤ <2,8,8,6> (T)

work = <2,14,11,8>

$P_4$ <0,6,4,2> ≤ <2,14,11,8> (T)

work = <2,14,12,12>

$P_1$ <0,7,5,0> ≤ <2,14,12,12> (T)

work = <3,14,12,12>

Safe Sequence $P_0$ $P_2$ $P_3$ $P_4$ $P_1$.

Ticket No. □□□□□□□□□□ Question Paper Code:A30516

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS)

B.Tech V Semester Mid-II Examinations December -2023

(Regulation: CMRCET-R18)

ject Name: Operating Systems Time: 1.30 PM to 3.10 PM
e: 29-12-2023 Branch: CSE Max Marks: 25

## PART A

**Answer all FIVE questions (Compulsory)**
**Each question carries TWO marks.** 5x2=10M

1 What is process synchronization? Define Race Condition?
2 Describe about Swapping
3 Describe about Demand Paging?
4 List out File attributes.
5 Write short notes on the following System calls.
i)Open() ii) Read() iii)Write() iv) Create()

## PART B

**Answer ALL questions.**
**Each question carries FIVE Marks.** 3x5=15M

6. A. Explain the following classical problems of synchronization.
i)Bounded Buffer problem or Producer –Consumer problem
ii)Dining philosopher problem
iii)Reader Writers problem

**OR**

6. B. i) Explain about Critical Section Problem.
ii) Illustrate about Synchronization Hardware for Critical Section Problem?

7. A. i) Explain Paging memory management technique in detail

**OR**

7. B. Discuss about various Directory Structures with neat diagrams

8. A. i) Identify the various techniques for Free Space Management.
ii) Explain about File System Structure

**OR**

8. B. i) Explain various types of Files?
ii) Compare and contrast the FCFS, SSTF, SCAN,C-SCAN,LOOK,C-LOOK disk scheduling algorithms with following example.
Head starts at 53 and queue[98,183,37,122,14,124,65,67]

************

Hall Ticket No. □□□□□□□□□□ Question Paper Code:A30516

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS)

B.Tech V Semester Mid-II Examinations December -2023

(Regulation: CMRCET-R18)

Subject Name: Operating Systems Time: 1.30 PM to 3.10 PM
Date: 29-12-2023 Branch: CSE Max Marks: 25

PART A

Answer all FIVE questions (Compulsory)
Each question carries TWO marks. 5x2=10M

1 What is process synchronization? Define Race Condition?
2 Describe about Swapping
3 Describe about Demand Paging?
4 List out File attributes.
5 Write short notes on the following System calls.
i)Open() ii) Read() iii)Write() iv) Create()

PART B

Answer ALL questions.
Each question carries FIVE Marks. 3x5=15M

6. A. Explain the following classical problems of synchronization.
i)Bounded Buffer problem or Producer - Consumer problem
ii)Dining philosopher problem
iii)Reader Writers problem

OR

6. B. i) Explain about Critical Section Problem.
ii) Illustrate about Synchronization Hardware for Critical Section Problem?

7. A. i) Explain Paging memory management technique in detail

OR

7. B. Discuss about various Directory Structures with neat diagrams

8. A. i) Identify the various techniques for Free Space Management.
ii) Explain about File System Structure

OR

8. B. i) Explain various types of Files?
ii) Compare and contrast the FCFS, SSTF, SCAN,C-SCAN,LOOK,C-LOOK disk scheduling algorithms with following example.
Head starts at 53 and queue[98,183,37,122,14,124,65,67]

************

Subject name : Operating Systems Branch : CSE

Subject code : A30516

mid II Examinations B. Tech V Sem

## 1.] Process Synchronisation

→ Process synchronisation means sharing system resources by processes in such a way that, concurrent access to shared data is handled, thereby minimising the chance of inconsistent data.

→ Process synchronisation problem arises in the case of cooperative process.

### Race Condition

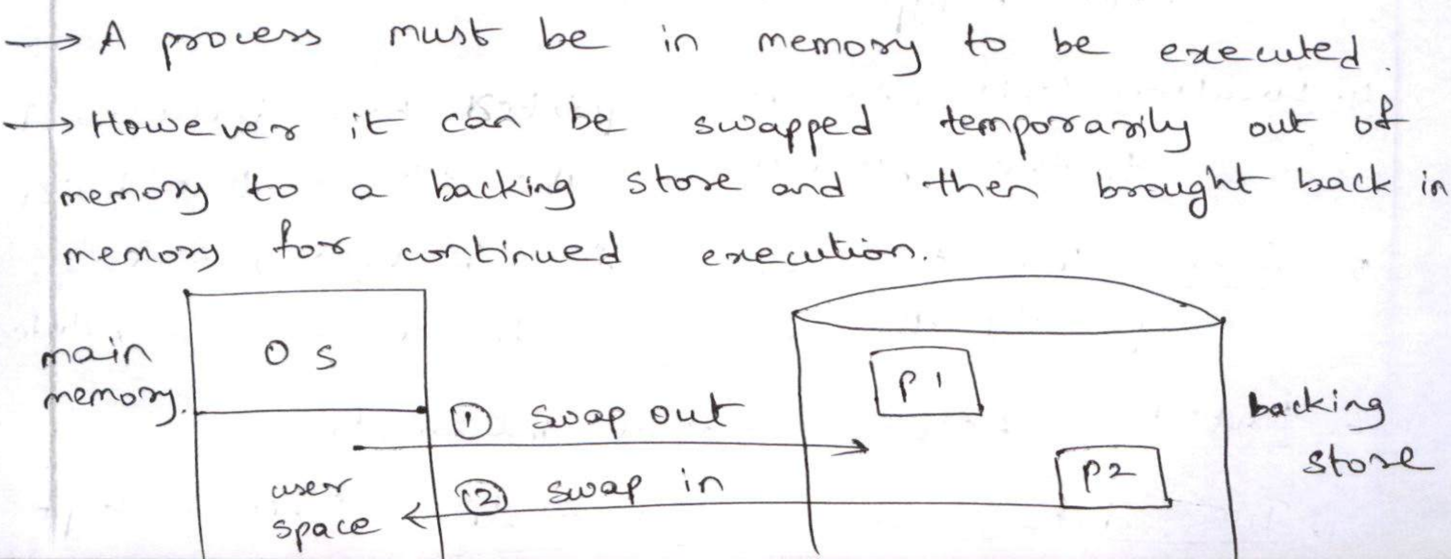
→ When several processes access and manipulate the same data concurrently and the outcome of the execution depends on the particular order in which the access takes place is called Race condition.

→ To guard against race condition, we need to ensure that only one process at a time can manipulate the variable.

## 2.] Swapping

→ A process must be in memory to be executed.

→ However it can be swapped temporarily out of memory to a backing store and then brought back into memory for continued execution.



## 3.] Demand Paging

→ Demand Paging is similar to paging system, with swapping

→ With Demand Paging a page is brought into main memory only when a reference is made to a locatio on the page.

→ Lazy swapper concept is used in demand paging

→ A lazy swapper never swaps a page into a memory unless that page will be needed.

→ Valid and invalid bits are used ~~[illegible]~~ to distinguish b/w those pages that are in memory and those on the disk.

## 4.] File Attributes

1. Name : The symbolic file name is the only information kept in human-readable form.

2. Identifier. A unique tag, a number, identifies the file within the file system.

3. Type : This info. is needed for systems that support different types of files.

4. Location : This info. is a pointer to a device and to the location of the file on that device

5. Size : Current size of file (in bytes) and possibly the maximum allowed size are included.

6. Protection : Access control information

7. Time, data, and user information : This is kept

for creation, last modification, and last use.

5. (i) Open() :

Used to open or create a file and obtain a file descriptor.

Syntax : int open(const char* pathname, int flags, mode_t mode);

(ii) Read() :

Used to read data from an open file descriptor into a buffer.

Syntax : ssize_t read(int fd, void *buf, size_t count);

(iii) Write () :

Used to write data from a buffer to an open file descriptor.

Syntax : ssize_t write (int fd, const void *buf, size_t count);

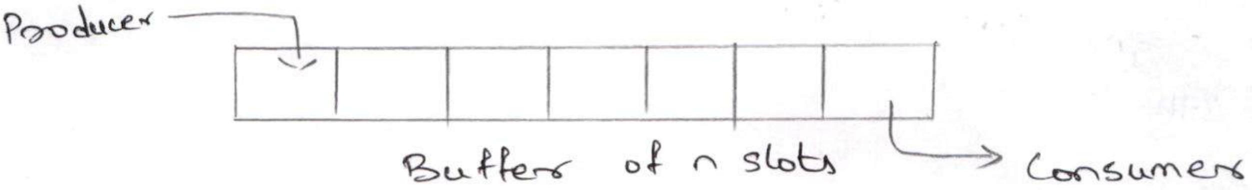
(iv) Create () :

Used to create a new empty file

Syntax : int ~~char~~ create (char *file name, mode_t mode);

## 6A. Bounded buffer problem

→ Producer tries to insert data into an empty slot and Consumer tries to remove data from filled slot in buffer.



### Producer operation

```
do{
    wait (empty);
    wait (mutex);
   //perform insert
   signal (mutex);
   signal (full);
}while (TRUE);
```

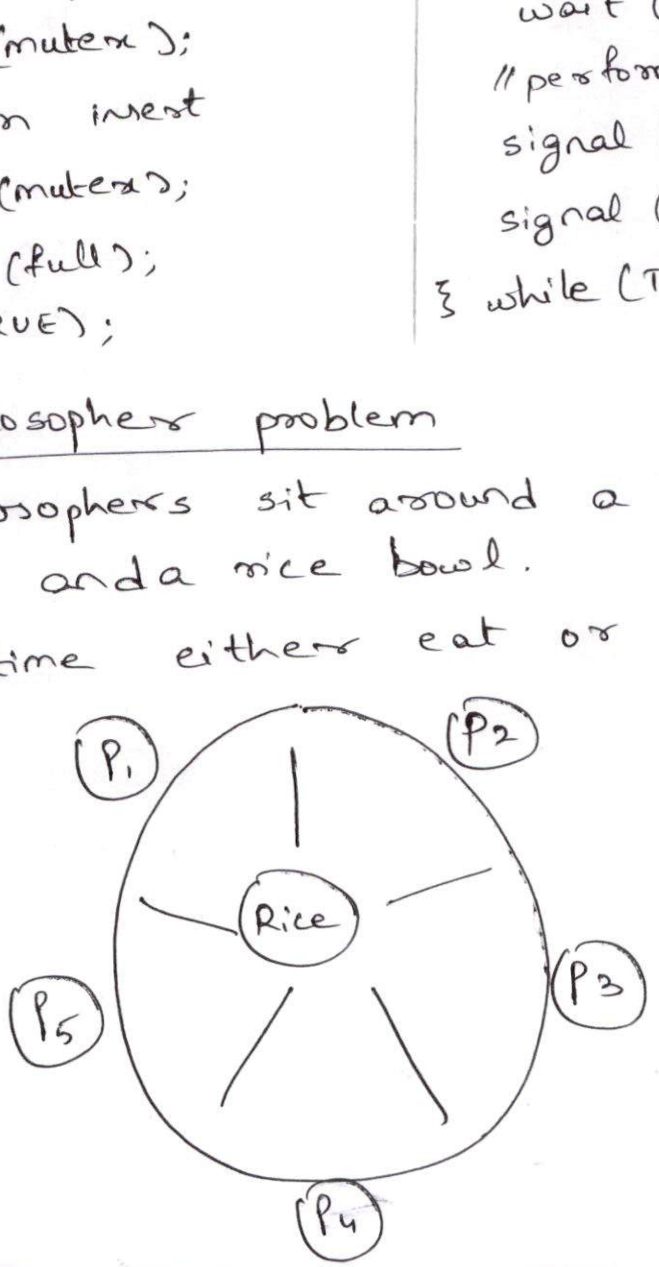
### Consumer operation

```
do {
   wait (full);
   wait (mutex);
 //perform remove
 signal (mutex);
 signal (empty);
} while (TRUE);
```

## Dining Philosopher problem

→ Five philosophers sit around a table with 5 chopsticks and a rice bowl.

→ At a time either eat or think



```
while (TRUE ) {
   wait (stick [i]);

   wait (stick[(i+1) % 5]);
    //eat
    signal (stick [i ]);
    signal (stick [(i+1)% 5]);
   //think
}
```

Readers Writers problem

→ Any number of readers can read from the shared resource, only one writer can write into resource.

for writer process

```
while (TRUE ) {

    wait (w);
   //perform write
   signal(w);
}
```

for readers process

```
while (TRUE) {
    //acquire lock
    wait (m);
    read_count++;
    if (read_count==1)
         wait (w);
    //release lock

    signal(m);

    //perform read

    // acquire lock

    wait(m);

    read_count--;

    if(read_count==0)
         signal (w);
     //release lock
     signal(m);
}
```

## 6.B (i) Critical Section Problem

Each process has a segment of code called critical section.

When one process is in its critical section, no other process is allowed to execute in its critical section.

general structure

```
do {
    entry section
        critical section
    exit section
        remainder section
} while (true);
```

Solution to CSP must satisfy:

1. Mutual Exclusion
2. Progress
3. Bounded waiting

Any solution to CSP requires a lock.

Race conditions are prevented by requiring that ~~cs4~~ critical regions be protected by locks.

```
do {
    acquire lock
        critical section
    release lock
        remainder section
} while (true);
```

7.A

## Paging

Memory management scheme that permits the physical address space of a process to be non contiguous.

Avoids external fragmentation and need for computation

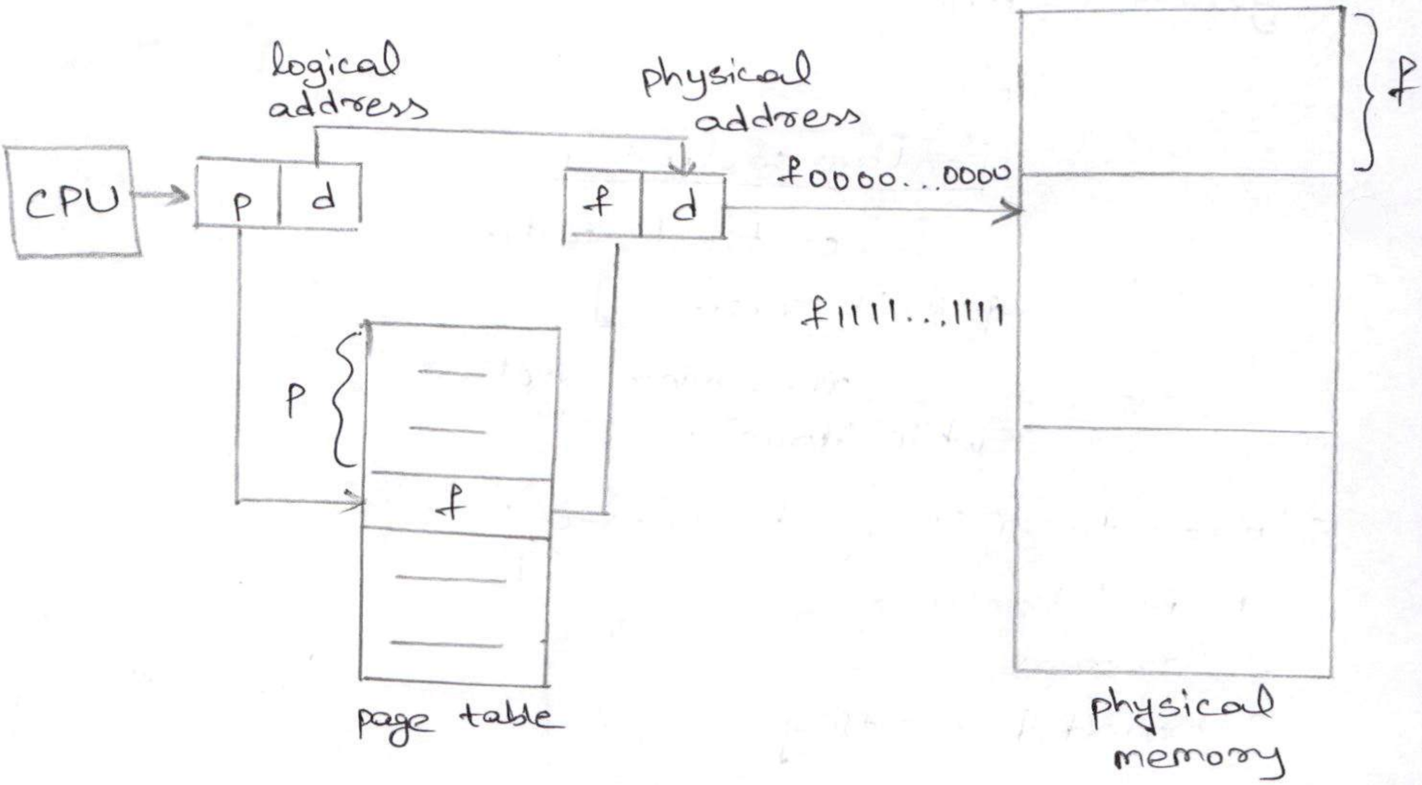


Fig: Paging hardware

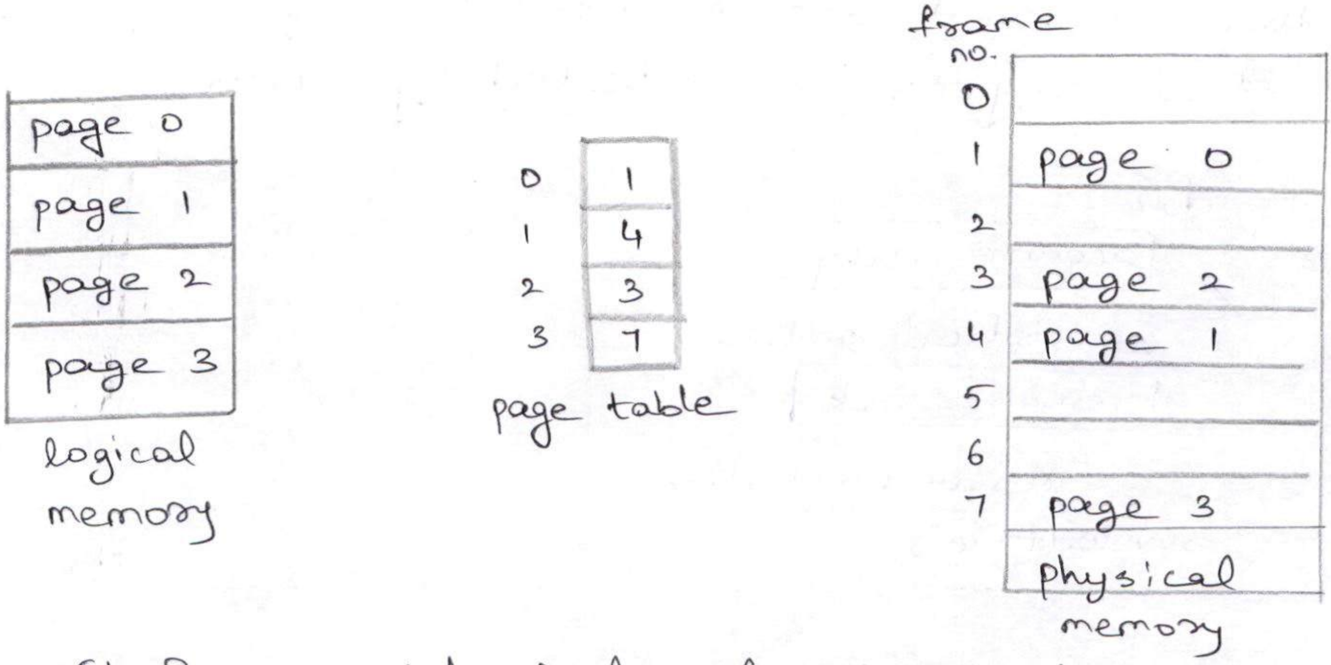


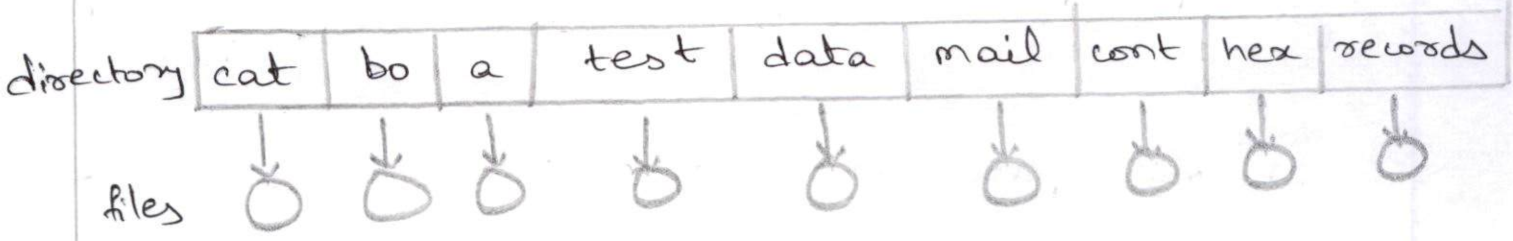
Fig: Paging model of logical and physical memory

| page numbers | page offset |
|---|---|
| p | d |
| m-n | n |

where p is an index into the page table and d is the displacement within the page.

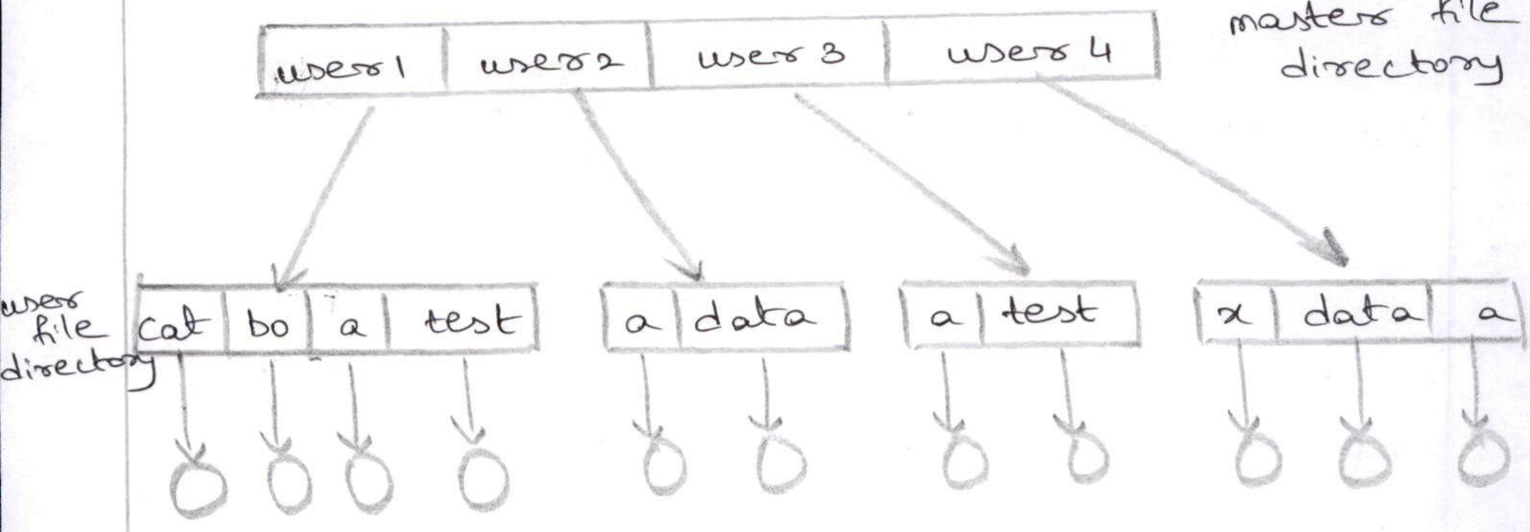
7.B (i) <u>Single level Directory structure</u>

All files are contained in the same directory.



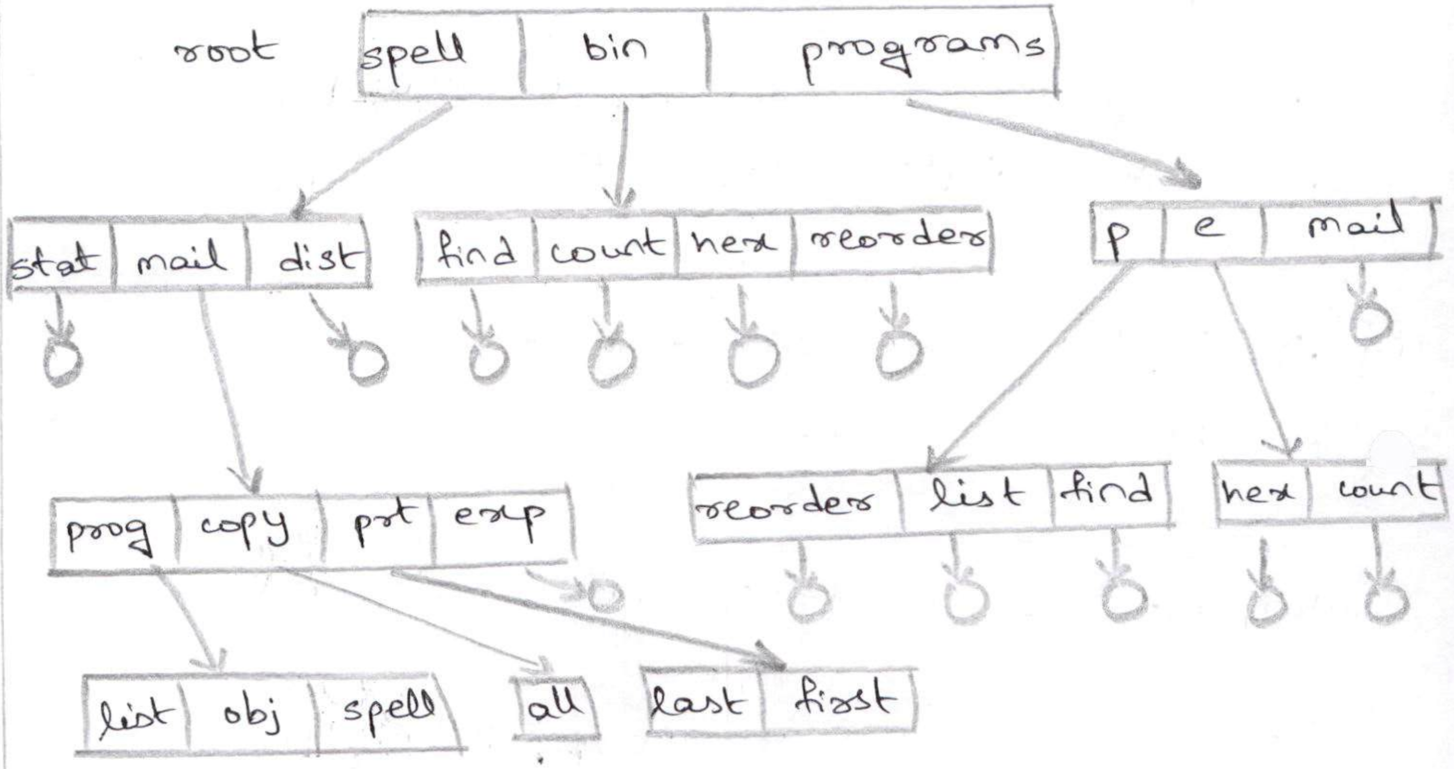
(ii) <u>Two level directory structure</u>

Each user has his own user file directory (UFD)



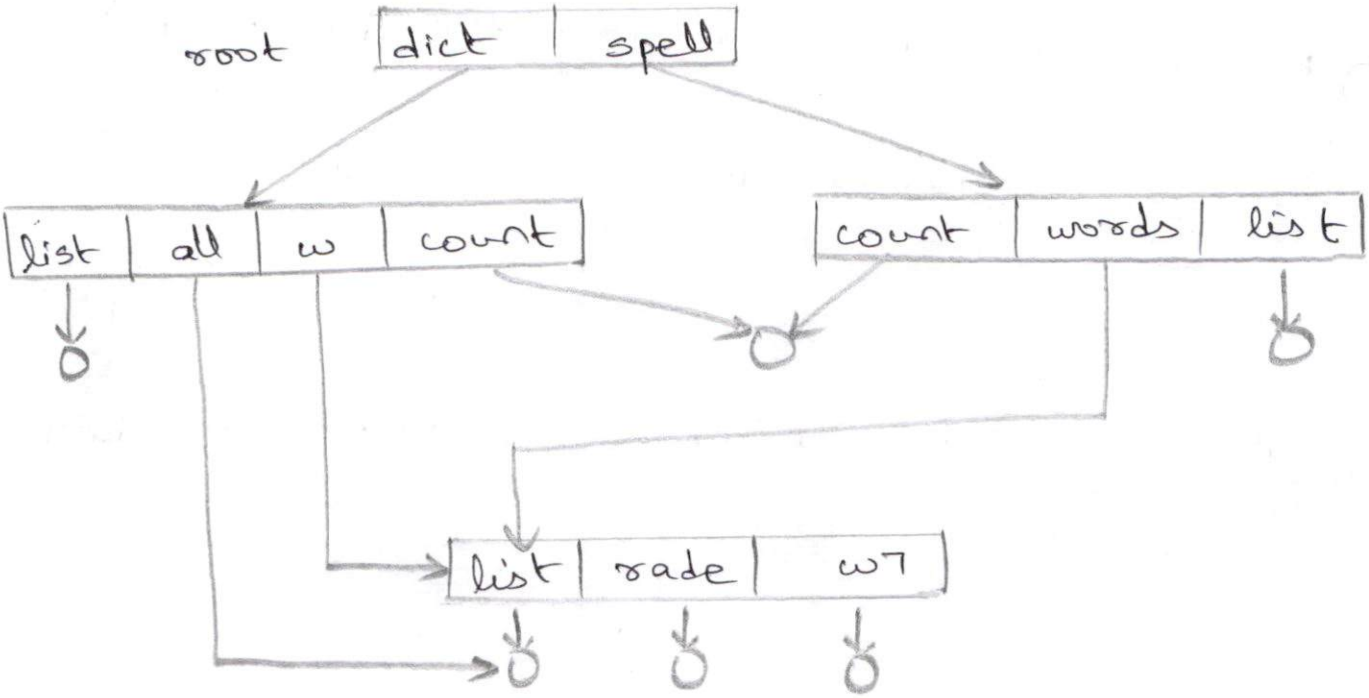
### (iii) Tree structure

Allows users to create their own subdirectories and to organise their files accordingly.



### (iv) Acyclic Graph structure

Allows directories to share subdirectories and files.



8. A (i) Free Space Management

Disk space is limited, need to use space from deleted files for new files.

To keep track of free disk space, system maintains free space list.

(1) Bit Vector

Free space list is implemented as a bit map or bit vector. Each block is represented by 1 bit. if block free then 1, if allocated then 0.

Ex.

disk blocks: 2, 3, 4, 5, 8, 9, 10, 11, 12, 13, 17, 18, 25, 26, 27

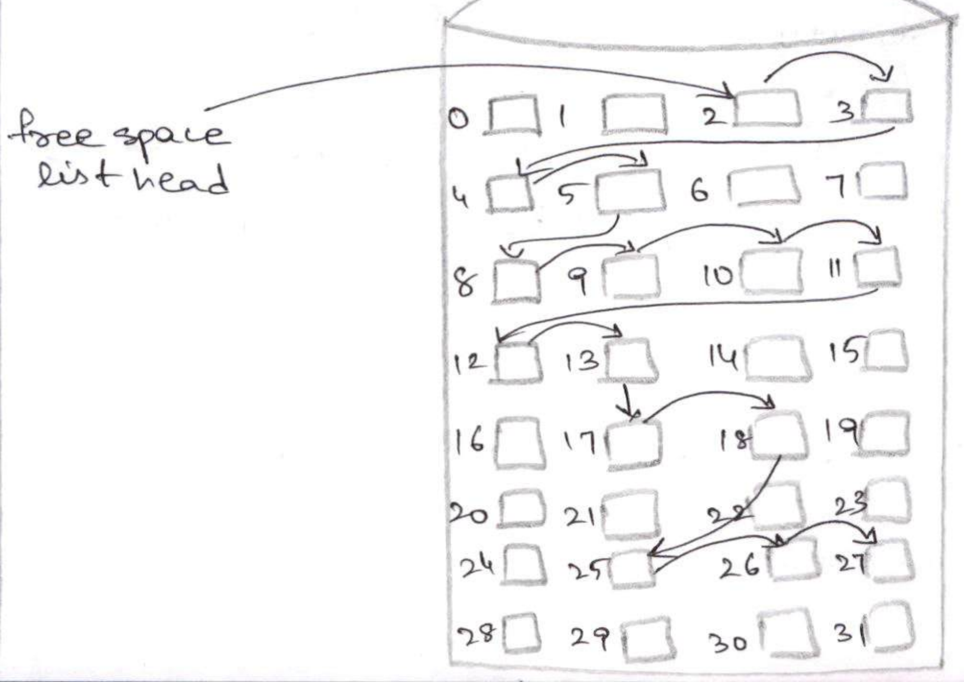
001111001111110001100000011100000...

Adv.

Relative simplicity and its efficiency in finding the first free block ~~of~~ or n consecutive free blocks on the disk.

(2) Linked list

Links together all the free blocks, keeps a pointer to the first free block.



Disadv.

Not efficient ; to traverse the list, we must read each block , requires substantial I/O time.

(3) Grouping

Stores the addresses of n free blocks in the first free block.

Adv.

Addresses of a large number of free blocks can be found quickly.

(4) Counting

Each entry in free space list contains a disk address and a count.

Disadv.

each entry requires more space than a simple disk address.

Adv.

Overall list is shorter.

(ii) File System Structure

File Systems provide efficient and convenient access to the disk by allowing data to be stored, located , and retrieved easily.

File system is composed of many layers or levels.

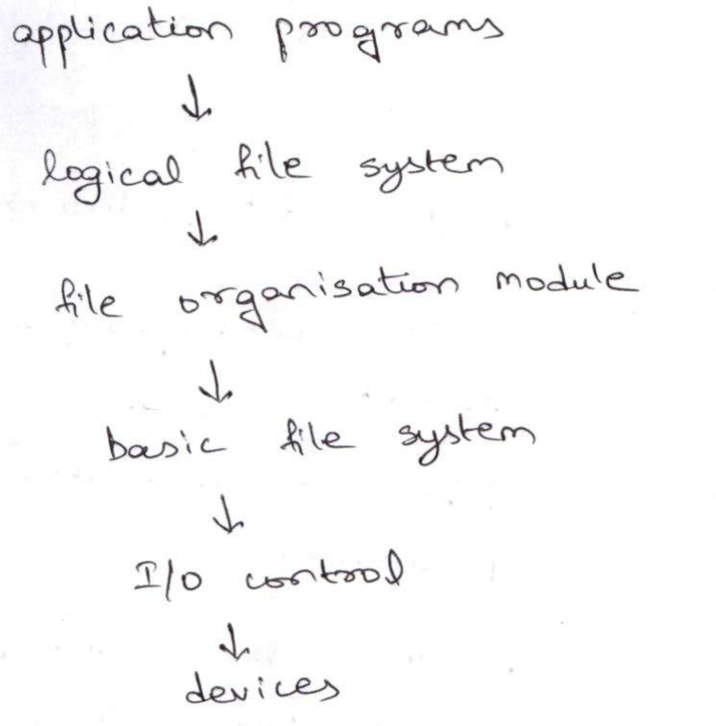


Fig: Layered file system

I/O control levels consists of device drivers and interrupt handlers to transfer info. b/w the main memory and the disk system.

The basic file system needs only to issue generic commands to the appropriate device driver.

Caches : used to hold frequently used file system metadata to improve performance.

File Organisation module knows about files and their logical blocks.

Logical file system manages metadata information.

8.B) (i) Types of files

| File type | usual extension | function |
|---|---|---|
| executable | exe, com, bin or none | ready to run machine language program |
| object | obj, o | compiled, machine language, not linked |
| source code | c, cc, java, pas, asm, a | source code in various languages. |

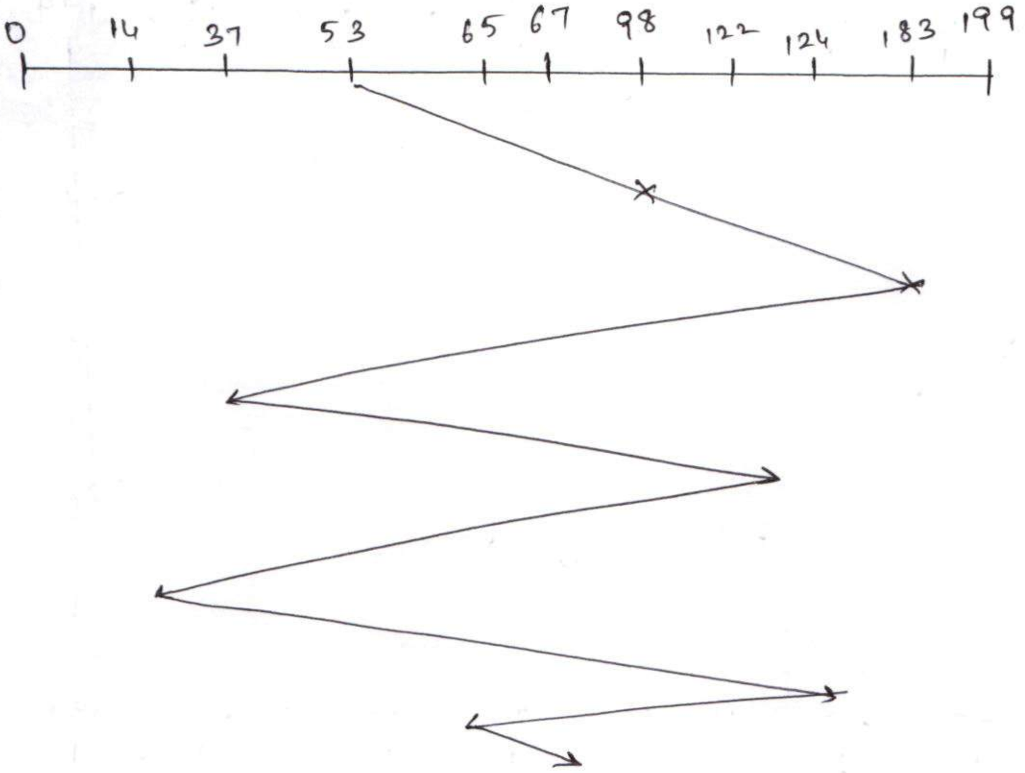
| | | |
|---|---|---|
| batch | bat, sh | Commands to the command interpreter |
| text | txt, doc | textual data, documents |
| word processor | wp, tex, rtf, doc | various word-processor formats. |
| library | lib, a, so, dll | libraries of routines for programmers. |
| print or view | ps, pdf, jpg | ASCII or binary file in a format for printing o viewing |
| archive | arc, zip, tar | related files grouped into one file, sometimes compressed, for archiving or storage. |
| multimedia | mpeg, mov, rm, mp3, avi | binary file containing audio or A/V information |

Fig: File Types.

(ii) FCFS

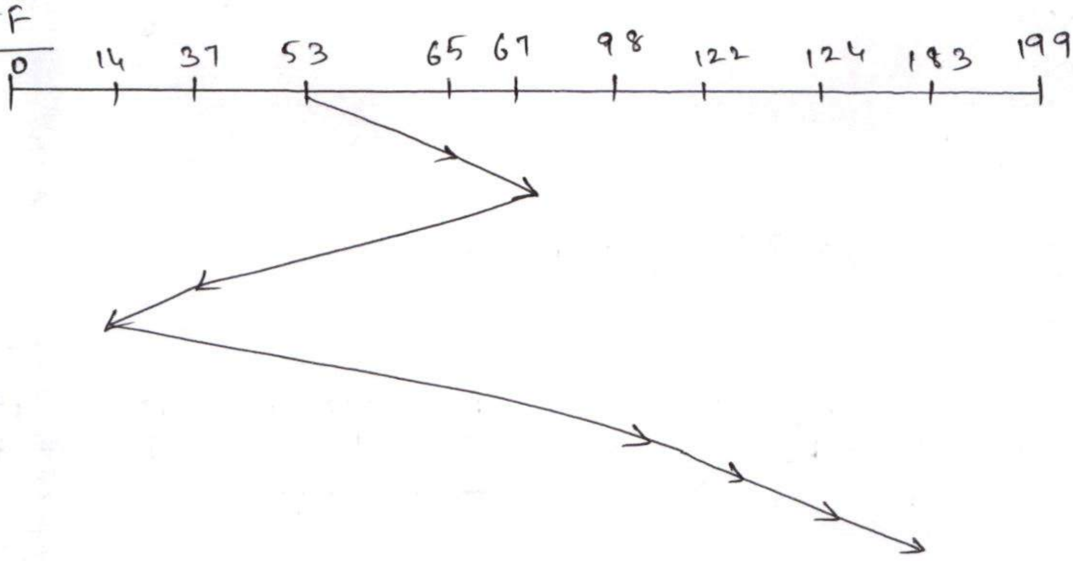
Queue : 98, 183, 37, 122, 14, 124, 65, 67

= 640 cylinders.



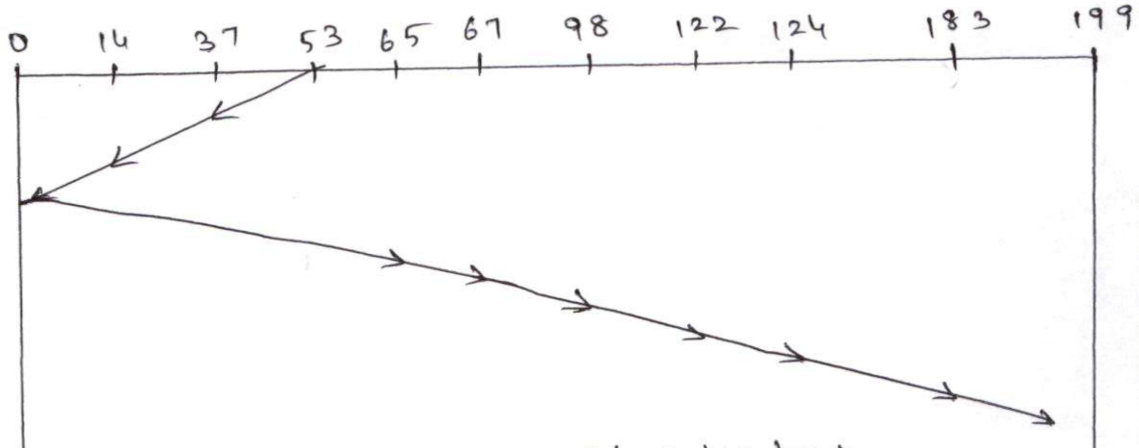
= 640 cylinders.

SSTF



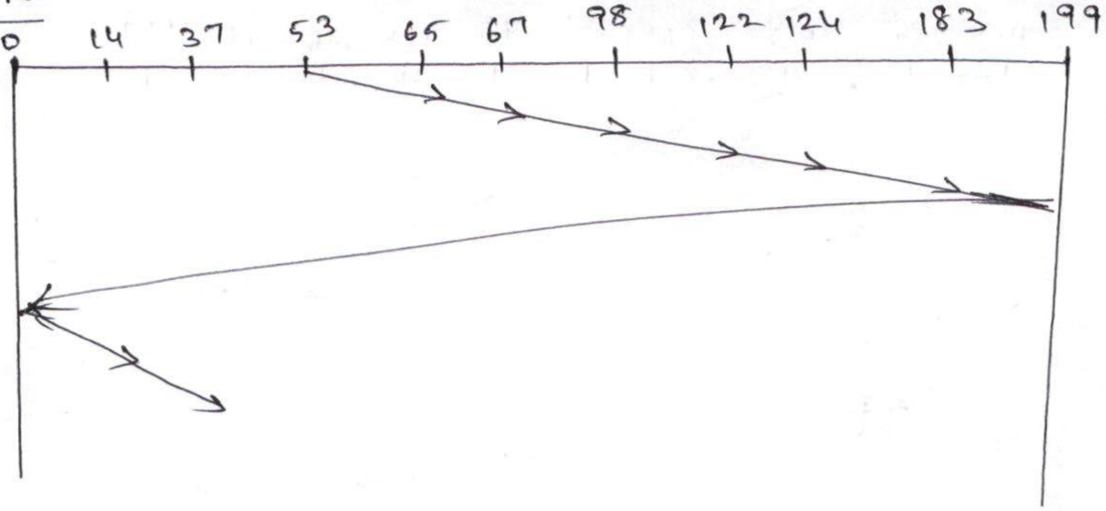
= 236 cylinders.

SCAN



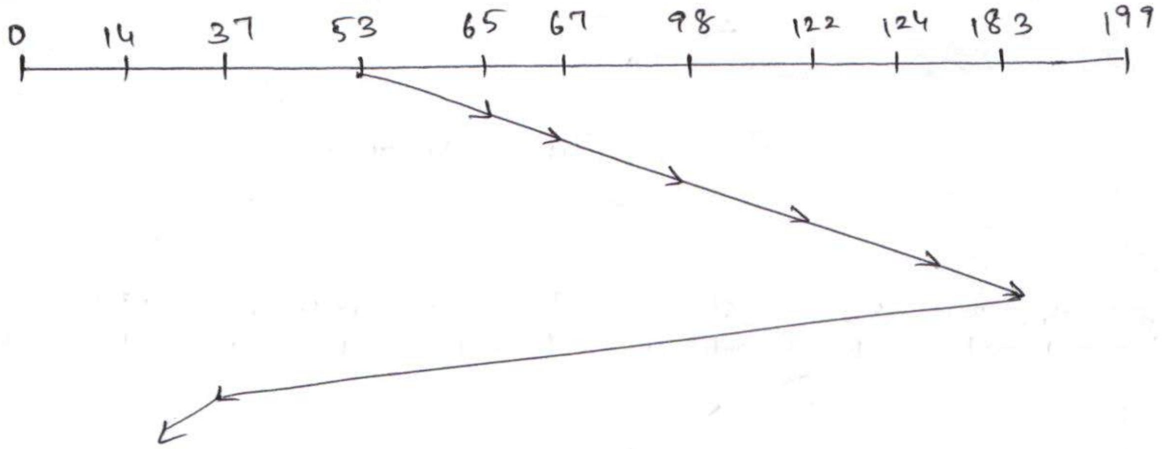
= 236 cylinders.

C-SCAN



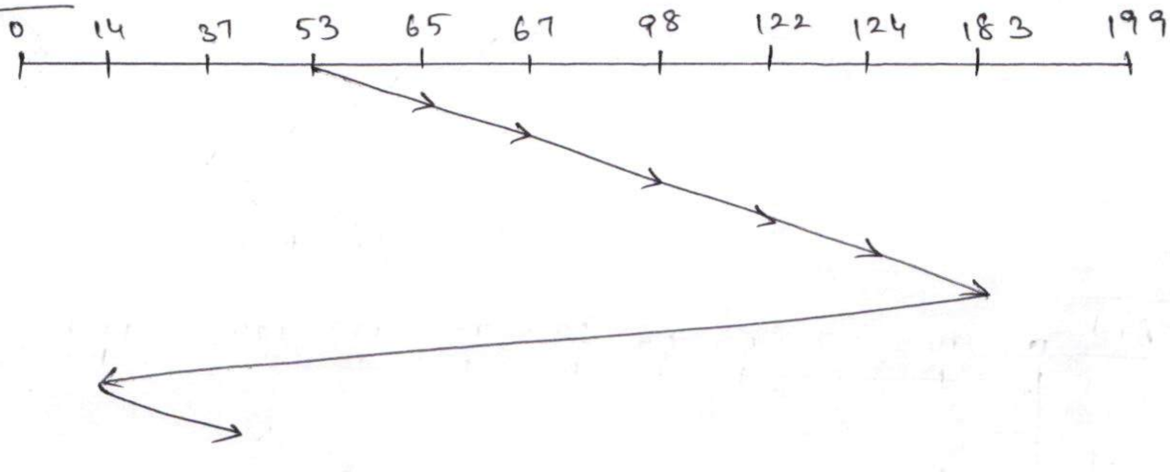
= 382 cylinders.

LOOK



= 299 cylinders.

C-LOOK



= 322 cylinders.

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(UGC AUTONOMOUS)

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD-501 401

ASSESSMENT OF PROGRAMME OUTCOMES & PROGRAMME SPECIFIC OUTCOMES

**PROGRAMME** **B.TECH (CSE)**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| YEAR | III | SEM | V | Academic Year | 2021-2022 | *BATCH* | *2019-202* |
| Course Code | A30516 | | | Course Name | OPERATING SYSTEMS | | |

## ARTICULATION

| S.No | COs | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PO12 | PSO1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | **CO1** | **3** | **1** | **1** | - | - | - | - | - | - | - | - | **1** | - |
| 2 | **CO2** | **3** | **3** | **2** | **2** | - | - | - | - | - | - | - | **2** | **1** |
| 3 | **CO3** | **3** | **3** | **2** | **1** | - | - | - | - | - | - | - | **1** | **1** |
| 4 | **CO4** | **3** | **1** | - | - | - | - | - | - | - | - | - | **1** | - |
| 5 | **CO5** | **3** | **3** | **2** | - | - | - | - | - | - | - | - | **1** | **1** |
| **Average** | | **3** | **3** | **2** | **2** | | | | | | | | **1** | **1** |

## FINAL ATTAINMENT (70% of External marks + 30% of Internal marks)

| Description | CO1 | C02 | C03 | CO4 |
|---|---|---|---|---|
| External Examinations Attainment | **3.00** | **3.00** | **3.00** | **3.00** |
| Internal Examinations Attainment | **2.00** | **2.00** | **3.00** | **3.00** |
| 70% of External Examinations Attainment | **2.10** | **2.10** | **2.10** | **2.10** |
| 30% of Internal Examinations | **0.60** | **0.60** | **0.90** | **0.90** |
| Final Attainment (70% of Ext + 30% of Int) | **2.70** | **2.70** | **3.00** | **3.00** |

## ATTAINMENT OF POs & PSOs THROUGH THE COURSE OUTCOMES

| **COs** | **Attain ment** | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | P07 | PO8 | PO9 | PO10 | PO11 | PO12 | PSO1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CO1 | **2.70** | 3 | 1 | 1 | - | - | - | - | - | - | - | - | 1 | - |
| CO2 | **2.70** | 3 | 3 | 2 | 2 | - | - | - | - | - | - | - | 2 | 1 |
| CO3 | **3.00** | 3 | 3 | 2 | 1 | - | - | - | - | - | - | - | 1 | 1 |
| CO4 | **3.00** | 3 | 1 | - | - | - | - | - | - | - | - | - | 1 | - |
| CO5 | **3.00** | 3 | 3 | 2 | - | - | - | - | - | - | - | - | 1 | 1 |
| **Attainment** | | **2.88** | **3.18** | **2.87** | **2.80** | - | - | - | - | - | - | - | **2.85** | **2.90** |

(Course Coordinator)

(Programme Coordinator

BOOKLET NUMBER :

CMR
GROUP OF INSTITUTIONS
EXPLORE TO INVENT

College Stamp

R18

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS)

CSE-A

Kandlakoya, Medchal, Hyderabad - 501 401.

## MID SEMESTER EXAMINATION ANSWER BOOK

Registered No. 2 1 H 5 1 A 0 5 3 3

FIRST / SECOND SEMESTER EXAMINATION B.Tech./M.Tech./MBA V th Semester Dec/2023
(Month and year)

Subject : Operating System

Date : 29-12-2023

29/12/2023
**Signature of the Invigilator with date**

## INSTRUCTIONS TO THE CANDIDATES

1. This booklet contains 16 pages. Candidates must ensure it before writing and in case a defective answer book is issued it must be returned to the invigilator and a new and defect free booklet must be obtained.
2. Before the candidate begins to answer, registered number, particulars of year, semester, subject etc., are to be filled in. Failure to enter all or any of these particulars may disqualify the paper from valuation.
3. Candidate is prohibited from
   (a) Writing.
   ☞ anything addressing the examiner in any manner whatsoever, in their answer book.
   ☞ Objectionable/obscene language in the answer book.
   ☞ anything other than their Registered Number on the question paper.
   (b) either seeking or providing any assistance to the fellow candidates in the exam.
   (c) possessing a manuscript or a printed matter, in any form, in the examination hall.
   (d) bringing loose sheets or paper into the examination hall and detaching any paper from the answer book.
   (e) carrying Mobile Phone to Exam Hall.
   **Violation of these instructions will be viewed as a case of malpractice, which is a punishable offence.**
4. Before beginning to answer any question, candidates must write the correct question number, in the margin only and should not write anything else in the margin.
5. Answers must be written legibly on both sides of the paper. There shall be about 25 lines in each page. It is not necessary to begin each answer on a fresh page. Candidates should not use any other ink, except BLACK or BLUE ink.
6. Rough work, if any, must be separated, from the subject matter, by a line and noted as rough work.
7. The answer book, at the end of the examination, must be handed over to the Assistant Superintendent (Invigilator) by the candidate **This responsibility lies with the candidate only.**
8. Candidates should maintain absolute silence during the time of examination. Misbehavior, in any form, by the candidate, in the examination hall, will attract severe punishment.
9. Candidates are permitted to leave the examination hall only after the expiry of half of the allotted time and candidates will be permitted to carry the question paper only when they are leaving the exam hall in the last half-an-hour.
10. **No additional answer books will be supplied.**

**To be filled in by the Examiner only**

PART - A / PART - B
MARKS SLIP

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PART-A | Q.No. | 1 | 2 | 3 | 4 | 5 | — | — | — | — | — | **Part-A Total** |
| | Marks | 2 | 1 | 2 | 2 | 2 | | | | | | 9 |

| | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PART-B | Q.No. | 6 | | 7 | | 8 | | — | — | — | — | — | — | — | — | — | — | **Part-B Total** |
| | | A | B | A | B | A | B | | | | | | | | | | | |
| | Marks | | 3 | 4 | | | 4 | | | | | | | | | | | 11 |

| | |
|---|---|
| **Grand Total in Words :** | **GRAND TOTAL** 20 |

Signature of the Scrutinizer with Date

Signature of the Examiner with Date

## Part-A

1. Ans:-

### process Synchronization

→ The process Synchronization :-

→ Coordination of process to avoid conflicts and data inconsistancy.

→ Race Condition:-

→ concurrent execution leading to unpredicatable results due to shared resources.

2 Ans:-

### Swapping

→ Swapping involves moving a process between main memory and secondary storage.

key points:-

1. purpose:-

→ Alleviates memory constrains by temporarily transfering less-used processed to disk.

2. performance impact:-

may cause delays due to the time required for swapping processes.

3. prevention:-

- Efficient memory management strategies aim to minimize the need for swapping.

4. Disk I/O Consideration:-

Swapping involves dis I/O operations impacting overall system performance.

3Ans

→ Loading pages into memory only when needed reducing initial loading time.

→ Additional points:-

1. optimization:-

→ Enhances system efficiency by loading only necessary pages into memory.

2. page Fault Handling:-

→ the system responds to pages faults by loading the required page on demand.

3. Resource utilization:-

· Minimizes wastages of memory by loading pages based on program Execution.

4Ans:-

Common Attributes:-

1. Name:-
 · unique identifier.

2. Type:
 → indicates the nature of the file.

3. Size:-
 → Amount of data stored.

· Additional points:-

4. Timestamps:
 → Record file-related timestamps such as creation modification, and acess times.

1. permissions:-

 → Specify who can read, write or execute the file,

2. Location:-

Stores the physical or logical location of the file in the file system.

3. Owner information:-

→ Identifies the user who owns the file, influencing access control.

5Ans:-

(i) Open():

- opens a file and returns a file descriptor.

(ii) Read():

- Reads data from an open file descriptor.

(iii) write():

- write data to an open file descriptor.

(iv) Create();

- Creates a new file or opens an Existing one.

## Part - B

6B

### (ii) process Synchronization

→ process synchronization refers to the coordination and orderly execultion of multipole processes or threads to ensure data consistency and avoid race conditions in a concurrent computing environment.

### Objective:-

→ the primary goal is to establish a set of rules or mechanisms that enable processes to work together without interfering with each other's data

### Synchronization Hardware:-

3

1. Mutex (Mutual Exclusion):

→ Auti~~x~~ is a synchronization primitive theat ensure only one process or theard can access a shared resouces at a time.

Example:-

· imagine a scenario where multiple proces need to write to a share file. A muten would be employed to ensure that only one proces can write to the file at any given moment, preventing data corruption.

2. Semaphores:

· Semaphores are integer variable used for signaling b/w processes to achieve process synchronization.

Example:-

in a scenario when multiple proces are producing and consuming items from a shared buffer semaphores can be employed to control access to the buffer the "empty" and "full" Semaphores. regulate the production and consumption, ensuring the buffer is not overfilled or underfilled.

3. Condition variables:

→ condition variables are synchronization primitives used for signaling between thread of processes to coordinate their activity.

Example:-

· Consider a situation where multipule threads are waiting for a specific condition to be true before proceeding. A conditio variable would be used to notifi waiting threads when the condition is met allowing them to continue their execution.

Conclusion:

Synchronization hardware and mechanism play a crucial role in ensuring the orderly and coordinated execution of processes in concurrent computing environments, preventing data inconsistence and race conditions.

=x=

8B

ii)

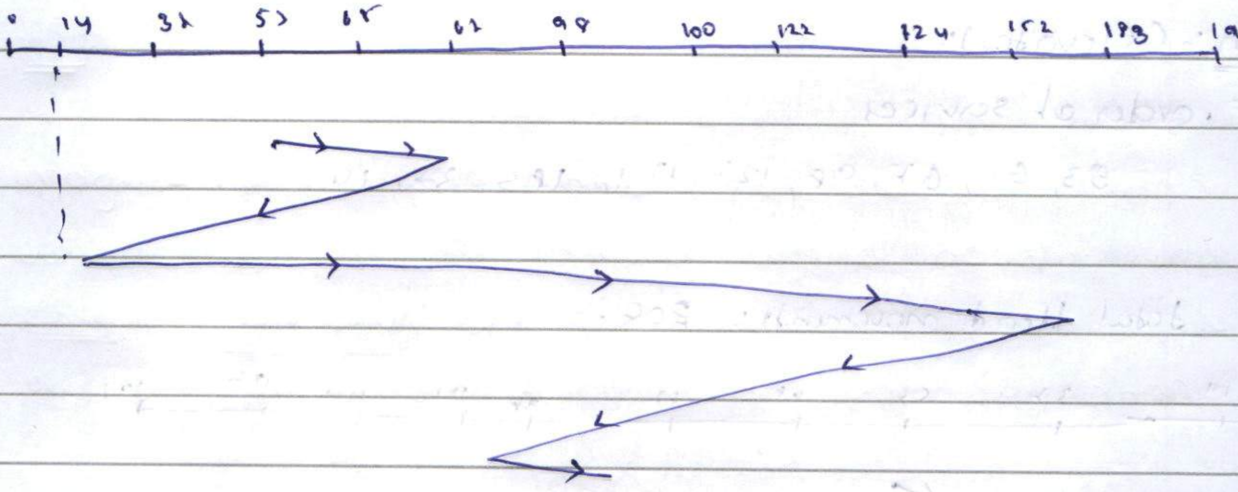
disk scheduling algorithms

Example Queue:- [98, 183, 37, 122, 14, 124, 65, 67]
Head start at: 53

ii] FCFS (First-come-First-Serve):
- order of service:

53, 98, 183, 37, 122, 14, 124, 65, 67

Total Head movements = 640

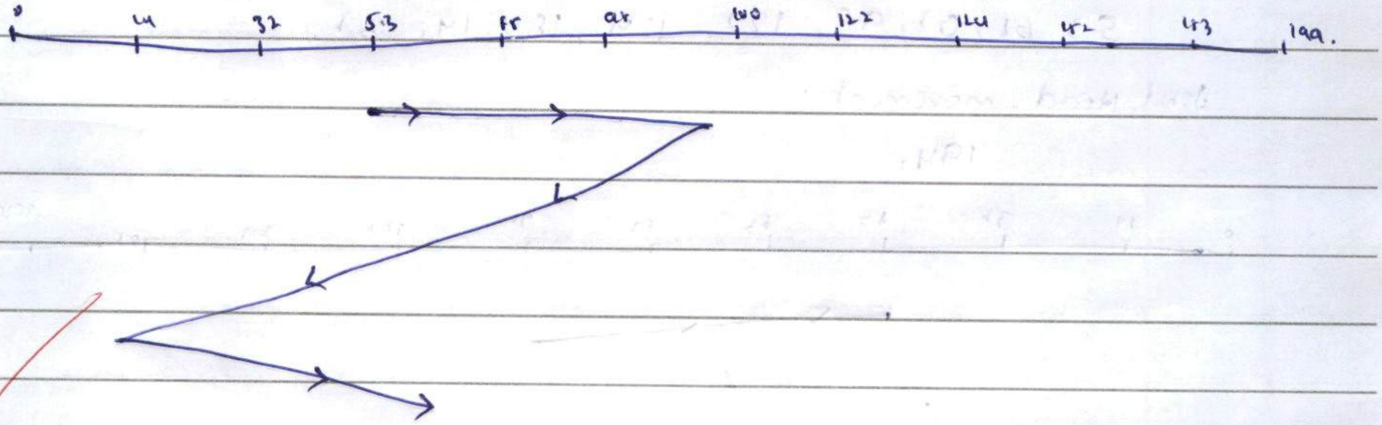


(ii) SSTF (Shortest seek time first):

.order of service.

53, 65, 32, 14, 98, 122, 124, 183

•total Head count is=236

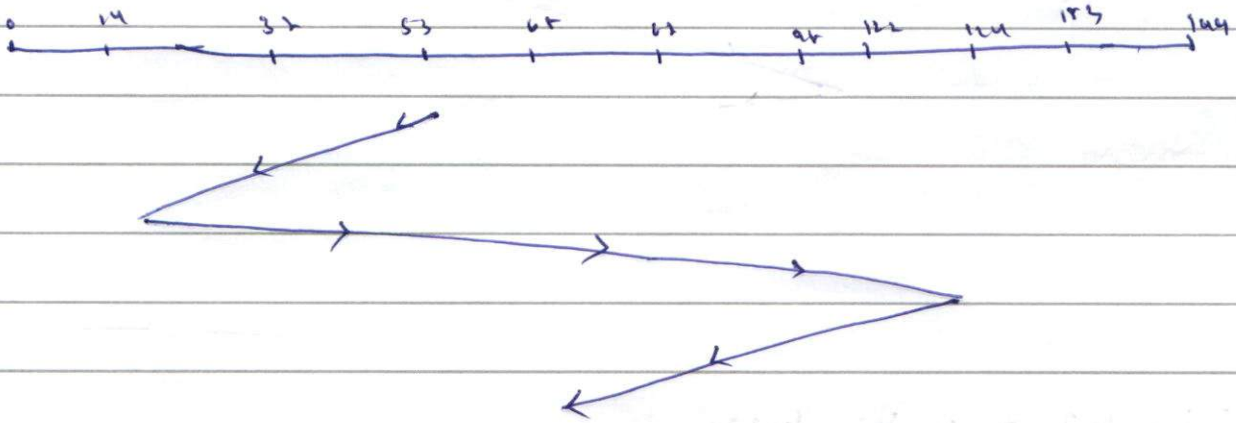


(iii) Scan:- (Elevator):

- order of services

  53, 65, 67, 98, 122, 124, 183, 37, 14

Total Head movement: 208.



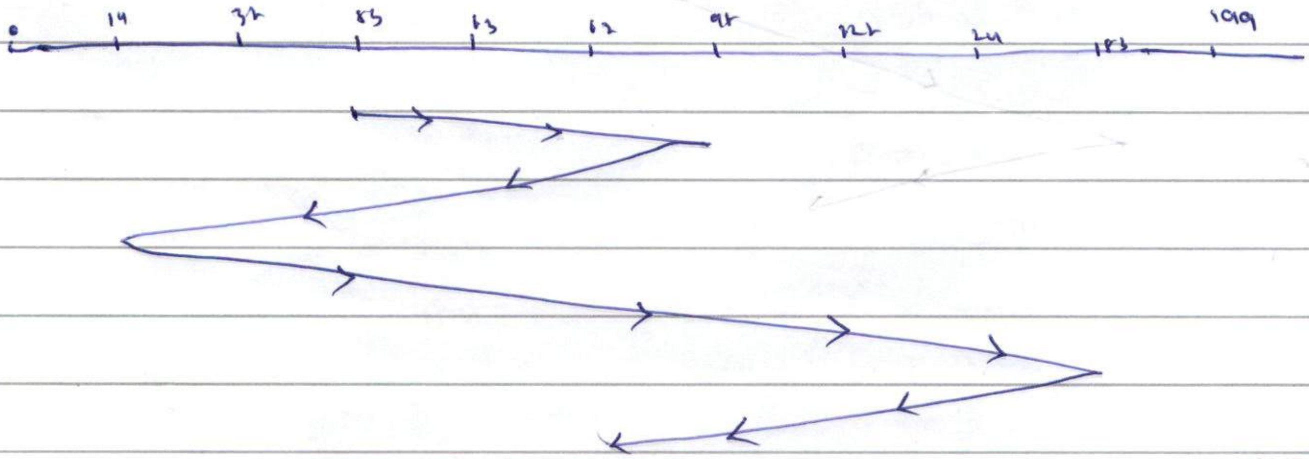
(iv) C-Scan (circular SCAN):

- order of services

  53, 65, 67, 98, 122, 124, 183, 14, 37,

Total Head movement.

194.



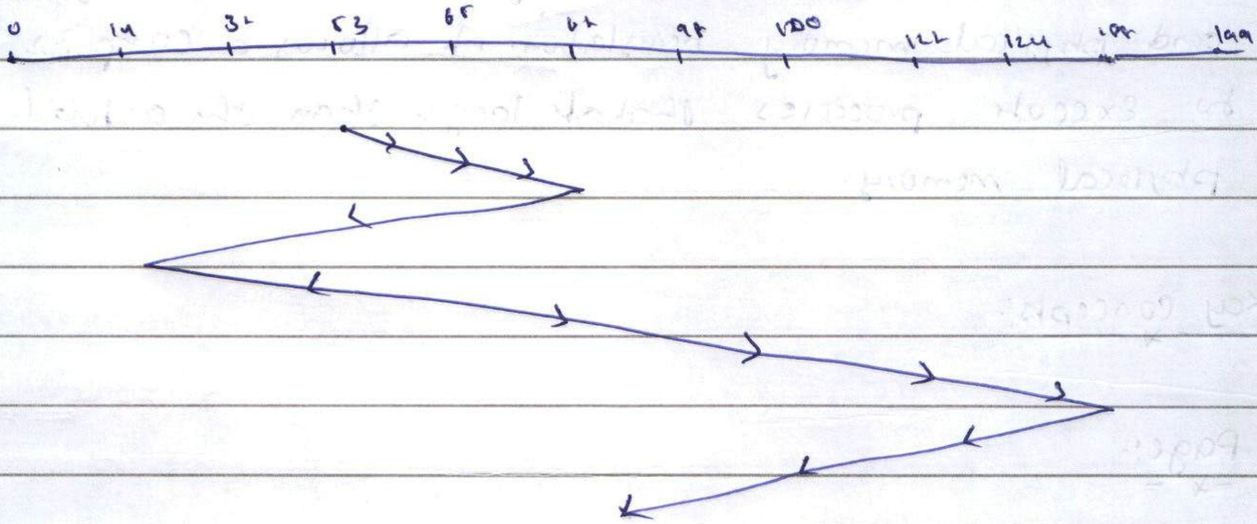
5. Look

Order of service

53, 65, 67, 98, 122, 124, 183, 37, 14

Total Head movements:

184.



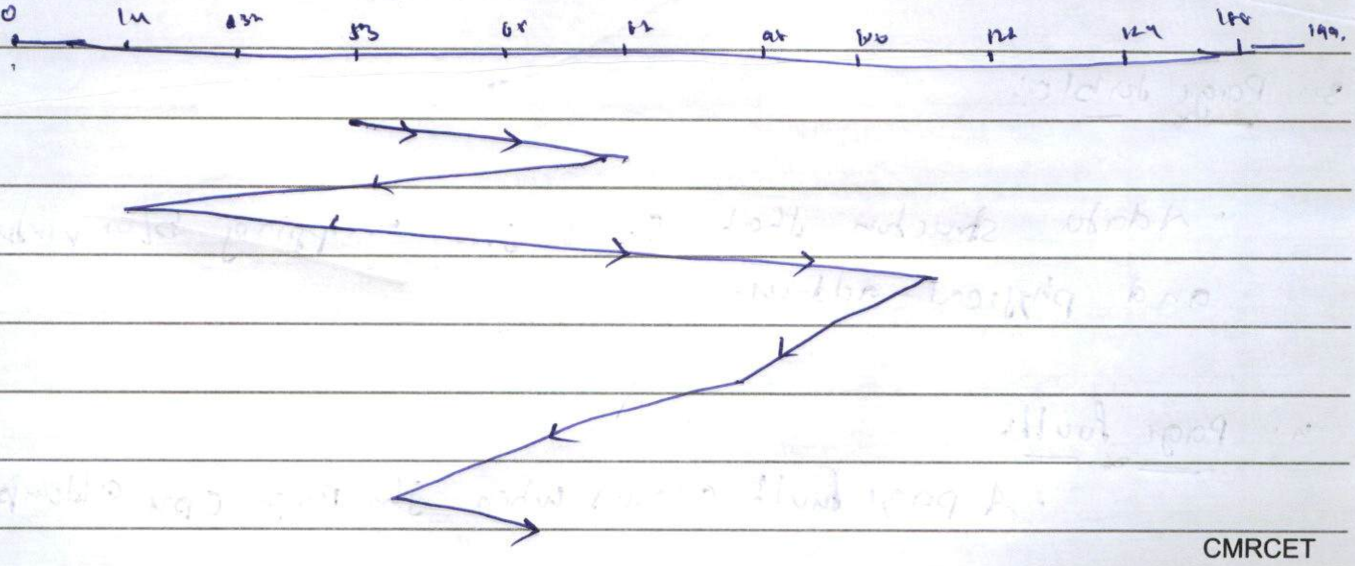
6. C-Look (circular look):

· order of service

53, 65, 67, 98, 122, 124, 183, 37, 14

· Total Head movements

184.



7A

## paging memory management Technique:-

paging is a memory management technique used in modern computer system to manage the virtual memory and physical memory translation. It allows a computer to execute processes that are larger than the actual physical memory.

### Key concepts:-

1. Page:-

   · A fixed-size contigous block of virtual memory.

2. Page Frame:-

   · A Fixed-size contiguous block of physical memory.

3. Page table:-

   · A data structure that stores the mapping b/w virtual and physical address.

4. Page fault:-

   · A page fault occurs when the page cpu attempts to

access a page that is not currently in physical memory.
(a page not resident.

5. Backing Store (Swap space):

An area on the disk used to store pages that are not current in physical memory.

Working of paging:-

1. memory Division:

2. Page table Creation.

3. Address translation

4. Page Fault Handing:

5. Page Replacement.

Advantages:-

1. Simplifies memory management.

2. Efficient use of memory.

3. Supports virtual memory.

BOOKLET NUMBER :

CMR
GROUP OF INSTITUTIONS
EXPLORE TO INVENT

College Stamp

R18

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(AUTONOMOUS)

Kandlakoya, Medchal, Hyderabad - 501 401.

CSE - A

## MID SEMESTER EXAMINATION ANSWER BOOK

Registered No. 2 1 H 5 1 A 0 5 1 5

FIRST / SECOND SEMESTER EXAMINATION B.Tech./M.Tech./MBA V<sup></sup>th Semester Nov 2023
(Month and year)

Subject : OS

Date : 02/11/2023

2/11/23
Signature of the Invigilator with date

## INSTRUCTIONS TO THE CANDIDATES

1. This booklet contains 16 pages. Candidates must ensure it before writing and in case a defective answer book is issued it must be returned to the invigilator and a new and defect free booklet must be obtained.
2. Before the candidate begins to answer, registered number, particulars of year, semester, subject etc., are to be filled in. Failure to enter all or any of these particulars may disqualify the paper from valuation.
3. Candidate is prohibited from
   (a) Writing.
   ☞ anything addressing the examiner in any manner whatsoever, in their answer book.
   ☞ Objectionable/obscene language in the answer book.
   ☞ anything other than their Registered Number on the question paper.
   (b) either seeking or providing any assistance to the fellow candidates in the exam.
   (c) possessing a manuscript or a printed matter, in any form, in the examination hall.
   (d) bringing loose sheets or paper into the examination hall and detaching any paper from the answer book.
   (e) carrying Mobile Phone to Exam Hall.
   **Violation of these instructions will be viewed as a case of malpractice, which is a punishable offence.**
4. Before beginning to answer any question, candidates must write the correct question number, in the margin only and should not write anything else in the margin.
5. Answers must be written legibly on both sides of the paper. There shall be about 25 lines in each page. It is not necessary to begin each answer on a fresh page. Candidates should not use any other ink, except BLACK or BLUE ink.
6. Rough work, if any, must be separated, from the subject matter, by a line and noted as rough work.
7. The answer book, at the end of the examination, must be handed over to the Assistant Superintendent (Invigilator) by the candidate **This responsibility lies with the candidate only.**
8. Candidates should maintain absolute silence during the time of examination. Misbehavior, in any form, by the candidate, in the examination hall, will attract severe punishment.
9. Candidates are permitted to leave the examination hall only after the expiry of half of the allotted time and candidates will be permitted to carry the question paper only when they are leaving the exam hall in the last half-an-hour.
10. **No additional answer books will be supplied.**

**To be filled in by the Examiner only**

PART - A / PART - B

MARKS SLIP

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PART-A | Q.No. | 1 | 2 | 3 | 4 | 5 | — | — | — | — | — | **Part-A Total** |
| | Marks | 2 | 1 | 2 | 2 | 1 | | | | | | 8 |

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PART-B | Q.No. | 6 | | 7 | | 8 | | — | — | — | — | — | — | — | **Part-B Total** |
| | | A | B | A | B | A | B | | | | | | | | |
| | Marks | 4 | | 5 | | | 5 | | | | | | | | 14 |

| | | |
|---|---|---|
| **Grand Total in Words :** | **GRAND TOTAL** | 22 |

Signature of the Scrutinizer with Date

Signature of the Examiner with Date

PART-A

1) Simple Batch Systems:

In Simple Batch Systems the processor depends upon the user and the system process is done by the users.

→ Simple Batch Systems are not dependent on the time.

Real Time Systems:

Real Time Systems depend on time itself only. The user is not dependent on RealTime Systems.

→ It only variants on the time not on users.

Operating System:

Operating system is defined as well developed process to work completely fine without any tasks or errors.

2) Various Components of Operating System:

1) Kernal Interface
2) User Interface
3) Device drivers
4) File Systems
5) Application programs

These are the various components of operating system.

3) Process Controll Block (PCB) organizes the process to complete the task without any errors. The steps of process is called as Process Control Block (PCB).

| |
|---|
| Pointers |
| Process Numbers |
| Process System |
| Process Calls |
| Registers |
| Microkernal Approach |
| Operating System |
| Module Approach |

Process Control Block (PCB)

4) Schedulling is the process to schedule the system to participate the task of a system.
It handles all the process by schedulling in a proper way is called as schedulling.

Types of Schedulers:

1) Throughput
2) Waiting Time
3) Turn Around Time (T.A.T)
4) Deadline
5) Priority
6) Burst Time
7) Need

2

5) Deadlock :

-) The process needs requirement and the availability of resources are less then the process enters inta waiting Time.

-) After entering into waiting stage the process does not change it's state, this is called as Deadlock in operating system.

PART-B

7. A) Shortest Job First (SJF) Algorithm :

In this algorithm, first the burst time is checked in an ascending order and then placed in the table

-) The process starts with the least Burst time and waiting time and turn around time is calulated.

-) By drawing Gantt chart we could calculate average waiting time and average turn around time.

Priority CPU Scheduling Algorithm:

-) In this algorithm the priorities are checked for the processes. The first priority is checked and then goes to second.

-) Here the waiting and Turn around time are calculated from the Gantt chart.

-) Turn Around Time = Waiting Time + Burst time.

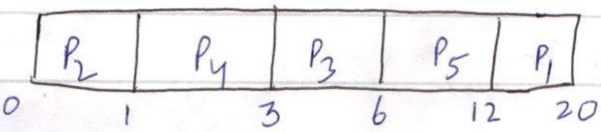
5

| Process id | Arrival time | Burst time(ms) | Priority |
|---|---|---|---|
| P1 | 0 | 8 | 3 |
| P2 | 1 | 1 | 1 |
| P3 | 2 | 3 | 2 |
| P4 | 3 | 2 | 3 |
| P5 | 4 | 6 | 4 |

(i) Gantt chart for Preemptive SJF:

| Process id | Burst time | Priority |
|---|---|---|
| P2 | 1 | 1 |
| P4 | 2 | 3 |
| P3 | 3 | 2 |
| P5 | 6 | 4 |
| P1 | 8 | 3 |

Gantt chart:



(ii) Gantt chart for preemptive priority:

| Process id | Priority | Burst time |
|---|---|---|
| P2 | 1 | 1 |
| P3 | 2 | 3 |
| P1 | 3 | 8 |
| P4 | 3 | 2 |
| P5 | 4 | 6 |

| $P_2$ | $P_3$ | $P_1$ | $P_4$ | $P_5$ |
|---|---|---|---|---|

0 1 4 12 14 20

(ii) Preamptive SJF:

| Process id | Waiting Time | Turn Around Time |
|---|---|---|
| $P_2$ | 0 | 1 |
| $P_4$ | 1 | 3 |
| $P_3$ | 3 | 6 |
| $P_5$ | 6 | 12 |
| $P_1$ | 12 | 20 |

Average Waiting Time = $\frac{0+1+3+6+12}{5} = \frac{22}{5} = 4.4$

Average Turn around Time = $\frac{1+3+6+12+20}{5} = 8.4$

Preamptive Priority:

| Process id | Waiting Time | Turn around Time |
|---|---|---|
| $P_2$ | 0 | 1 |
| $P_3$ | 1 | 4 |
| $P_1$ | 4 | 12 |
| $P_4$ | 12 | 14 |
| $P_5$ | 14 | 20 |

Average Waiting Time = $\frac{0+1+4+12+14}{5} = 6.2$

Average Turn Araund Time = $\frac{1+4+12+14+20}{5} = 10.2$

8. B) Deadlock :

-) The process needs requirement and the availability of resources are less than the process enters into a waiting stage.

-) After entering into waiting stage the process does not change it's state , this is called as Deadlock.

Bankers Algorithm:

Bankers Algorithm defines as the safest way of processing the process and keeping the system safe.

a) Data Structure :

These are the data structures used in bankers algorithm. They are,

1) Allocation

2) Need

3) Max

4) Available

1) Allocation: The resources which we have with us is called as Allocation.

2) Need: The resources which are required to us to complete the requirement is called as need.

3) Max: The resources which are already completed their needs are called as Max.

4) Available: The resources which are available to us and required is called as Available.

b) Safety Algorithm :

Step 1: First we have to assume all the states as False.

Final = [F, F, F, F, F]

Step2: Consider the available as work.

Work = Available.

Step3: By assuming all the proces as false, assume $P_0$ as false and check the condition of $P_0$ that need ≤ Work.

Need ≤ Work

Step4: If false, check the other proces id, if true then replace the work by adding the allocation of that proces id to the work will be our new work.

Allocation+Work = Work

Step5: Check all the proces id's and safest state is the one which will be true in ascending order.

This is the safety Algorithm.

c) Resource Request Algorithm:

There are 3 main steps to remember in resource Request Algorithm. They are:

i) Need ≤ Request

Our need should always be less than or equal to the Request. If not then the process would diedly fail. If true then it goes to second step.

ii) Need ≤ Available

Here if need ≤ Available then it goes to third step, if not then it enters into a waiting stage.

iii) Here,

allocation : +req;
Need : -req;
Available : -req;

These are the 3 main steps in resource request algorithm.

Given,

| | Allocation | | | Max | | | Available | | |
|---|---|---|---|---|---|---|---|---|---|
| | A | B | C | A | B | C | A | B | C |
| $P_0$ | 0 | 1 | 0 | 7 | 5 | 3 | 3 | 3 | 2 |
| $P_1$ | 2 | 0 | 0 | 3 | 2 | 2 | | | |
| $P_2$ | 3 | 0 | 2 | 9 | 0 | 2 | | | |
| $P_3$ | 2 | 1 | 1 | 2 | 2 | 2 | | | |
| $P_4$ | 0 | 0 | 2 | 4 | 3 | 3 | | | |

$\boxed{\text{Need} = \text{Max} - \text{Allocation}}$

Need

| | A | B | C |
|---|---|---|---|
| $P_0$ | 7 | 4 | 3 |
| $P_1$ | 1 | 2 | 2 |
| $P_2$ | 6 | 0 | 0 |
| $P_3$ | 0 | 1 | 1 |
| $P_4$ | 4 | 3 | 1 |

Step 1: Final [F, F, F, F, F]

Step 2: Assume $P_0 = F$ , work = Available = 3 3 2

$7\ 4\ 3 \leq 3\ 3\ 2$

$\therefore P_0 = F$

, Assume $P_1 = F$

$1\ 2\ 2 < 3\ 3\ 2$

$\boxed{\therefore P_1 = T}$ as need ≤ Work

Allocation + work = work

200 + 332 = 532

∴ Work = 532

step 3: Assume , $P_2$ = F

600 ≤ 532

∴ $P_2$ = F

Assume , $P_3$ = F

011 ≤ 532

$\boxed{∴ P_3 = \text{True}}$

∴ Allocation + work = work

011 + 532 = 743

∴ Work = 743

step 4: Assume $P_4$ = F

431 ≤ 743

$\boxed{∴ P_4 = \text{True}}$

Allocation + work = Work

~~[illegible]~~ 002 + 743 = 745

∴ Work = 745

$P_0$ = F

By checking $P_0$ with work = 545

743 ≤ 745

~~[illegible]~~ $\boxed{P_0 = T}$

~~[illegible]~~ Allocation + work = work

010 + 745 = 755

Work = 755

∴ $P_2$ = F

600 ≤ 755

$\boxed{\therefore P_2 = T}$

Allocation + work = work

$200 + 755 = 955$

Work = 955

$\therefore$ The safest path is $P_1 \rightarrow P_3 \rightarrow P_4 \rightarrow P_0 \rightarrow P_2$

6. A) Operating System Structures:

As the system will be large and complex we have to engineer them carefully as long as it should be well developed.

The four structures in operating system are:

i) Simple Structures

ii) Layered Approach

iii) Microkernel Approach

iv) Module Approach

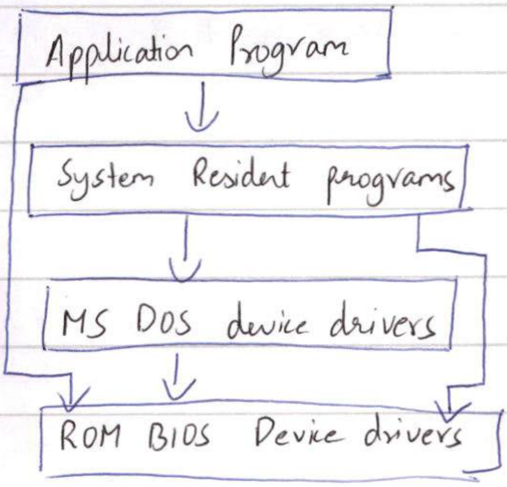
i) Simple Structure:

-> Many systems are not well functioned. Simple structures are small, finite and limited

-> The best example is UNIX OS. It have 2 parts. They are

i) Kernel

ii) System programs.



4

-) MS-DOS device driver is used in simple structures.

-) If one system gets an error then the whole system gets crash in the simple structure.

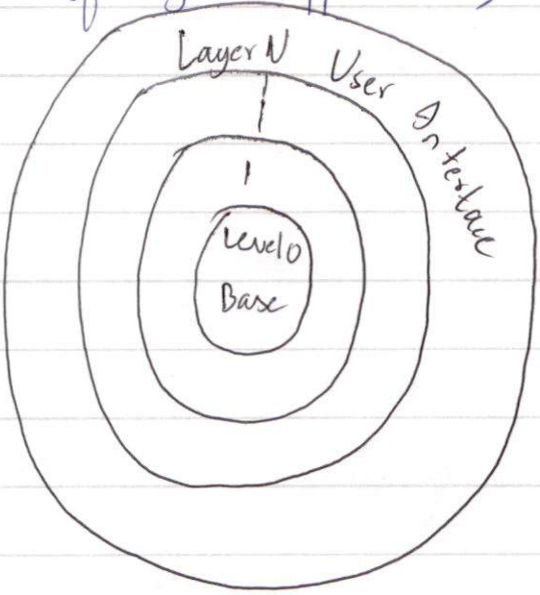
ii) Layered Approach:

-) In this approach, the layers are built and levels are divided.

-) Level 0 is called as Base

-) Level N is called as User Interface.

-) The structure of Layered Approach is;



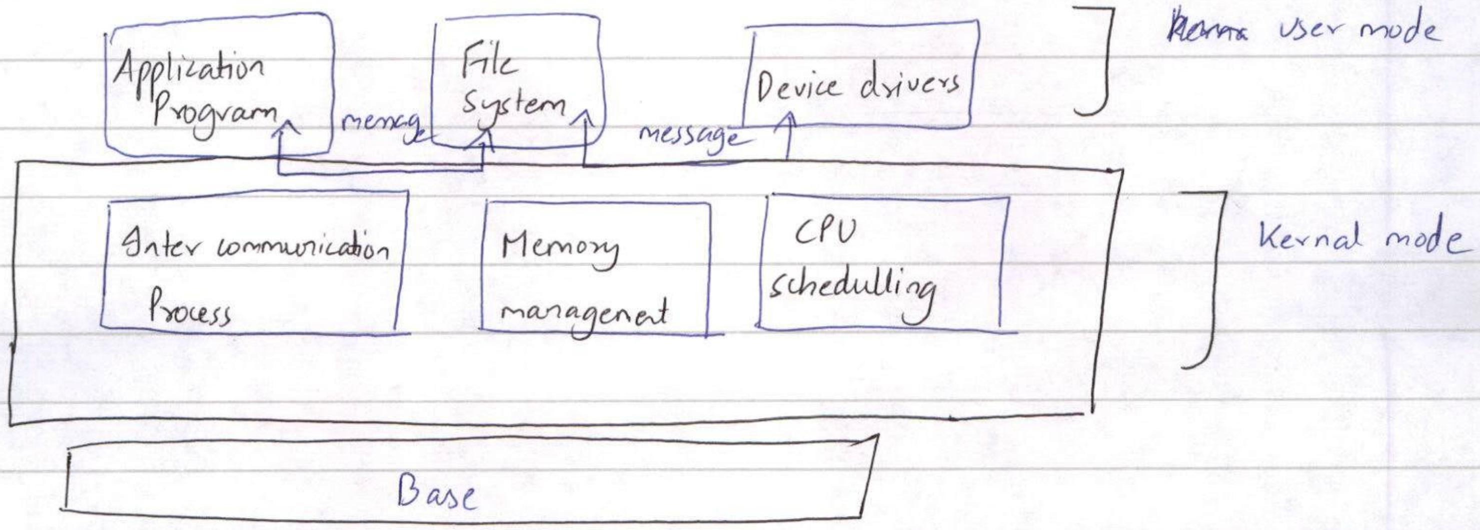
-) If one layer gets debugged and errors are found, then that level is only terminated.

-) Layer N can introvoke operations on lower levels.

-) Every layer gets debugged each after the other.

iii) Micro kernal Approach

-) This is a multiple level Approach.

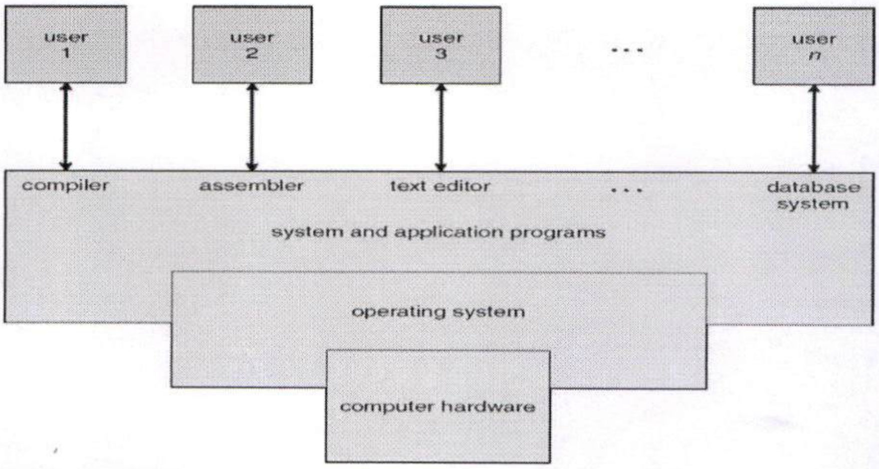


Micro kernal Approach

-) It gives the best security to the operating system.

-) If one system gets failed then all the other system work except that system.

### Components of a Computer System



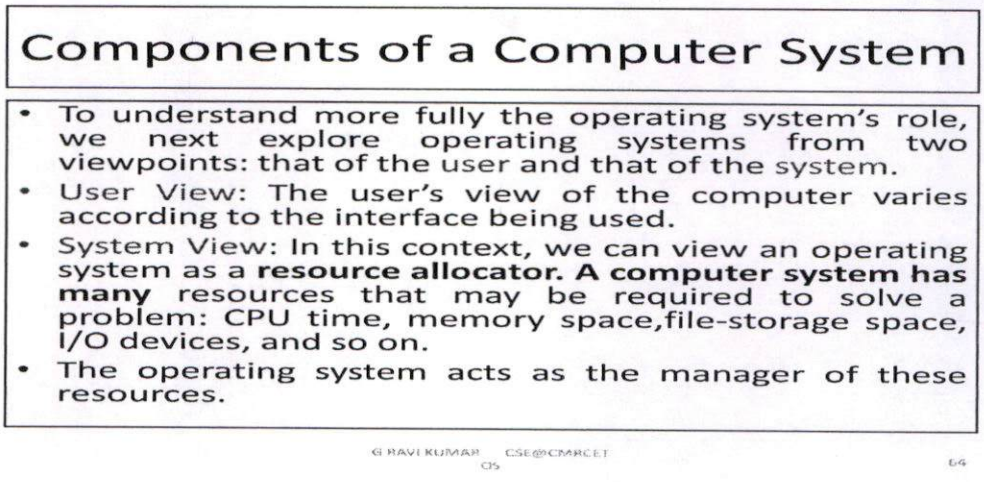
**Computer system can be divided into four components**

- Hardware – provides basic computing resources
  CPU, memory, I/O devices

- Operating system
  Controls and coordinates use of hardware among various applications and users

- Application programs – define the ways in which the system resources are used to solve the computing problems of the users
  Word processors, compilers, web browsers, database systems, video games

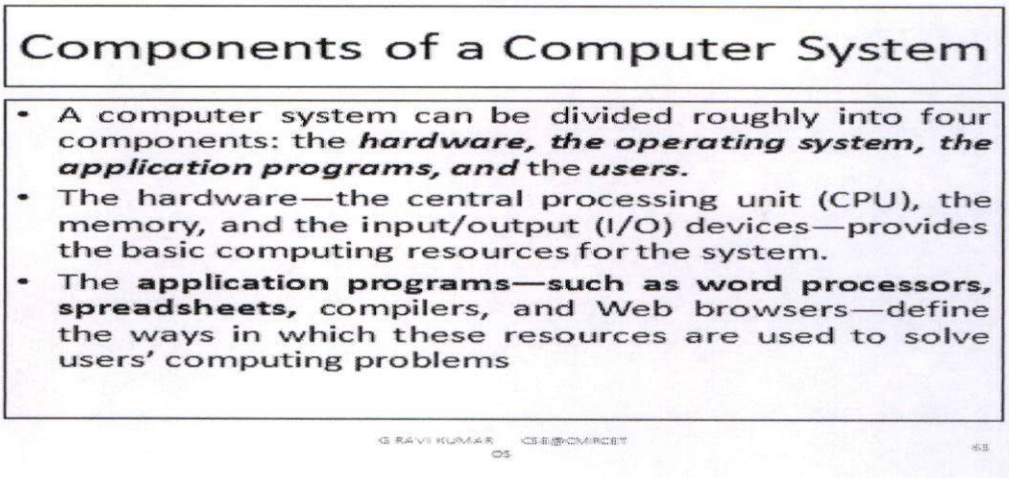
- Users
  People, machines, other computers



## Components of a Computer System

- To understand more fully the operating system's role, we next explore operating systems from two viewpoints: that of the user and that of the system.
- User View: The user's view of the computer varies according to the interface being used.
- System View: In this context, we can view an operating system as a **resource allocator. A computer system has many** resources that may be required to solve a problem: CPU time, memory space,file-storage space, I/O devices, and so on.
- The operating system acts as the manager of these resources.

G RAVI KUMAR CSE@CMRCET
OS
64



## Components of a Computer System

- A computer system can be divided roughly into four components: the ***hardware, the operating system, the application programs, and*** the ***users.***
- The hardware—the central processing unit (CPU), the memory, and the input/output (I/O) devices—provides the basic computing resources for the system.
- The **application programs—such as word processors, spreadsheets,** compilers, and Web browsers—define the ways in which these resources are used to solve users' computing problems

G RAVI KUMAR CSE@CMRCET OS 53

**Computer Startup**

- **Bootstrap program** is loaded at power-up or reboot
- Typically stored in ROM or EPROM, generally known as **firmware**
- Initializes all aspects of system
- Loads operating system kernel and starts execution

.

**Operating-System Structure**

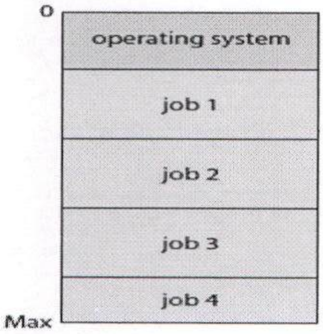


**Figure 1.9** Memory layout for a multiprogramming system.

One of the most important aspects of operating systems is the ability to multiprogram. The idea is as follows: The operating system keeps several jobs in memory simultaneously (Figure 1.9).

In general, Main Memory is too small to accommodate all jobs, the jobs are kept initially on the disk is called **Job Pool.**

This pool consists of all processes residing on disk and waiting allocation of main memory.

The operating system picks and begins to execute one of the jobs in memory. Eventually, the job may have to wait for some task, such as an I/O operation, to complete.

In a Non-Multiprogrammed system, the CPU would sit idle.

In a Multiprogrammed system, the operating system simply switches to and executes another job. When ***that*** job needs to wait, the CPU switches to ***another*** job, and so on.

Eventually, the first job finishes waiting and gets the CPU back. As long as at least one job needs to execute, the CPU is never idle.

Multiprogrammed system provides an environment in which various system resources(CPU, MEMORY, Peripheral Devices) are utilized effectively, but they don't provide, the user interaction with computer system.
Multiprogramming works on the concept of context switching.

Multitasking or Time Sharing: Multitasking is an logical extension of multiprogramming. In these systems, the CPU executes multiple jobs, by switching among them, but the switches occur so frequently that the users can interact with each program while it is running.

Multitasking is based on the time sharing alongside the concept of context switching.

In Time sharing system, each process is assigned some specific quantum of time for a process to execute. As soon as time quantum or one process expires, another process begins its execution.

Here also a context switch is occurring but it is so fast that the user is able to interact with each program separately while it is running.
But actually only one process/task is executing at a particular instant of time.

For example 4 processes and 5 nano seconds.

**Multiprocessing**

Multiprocessing is basically executing multiple processes at the same time on multiple processors, whereas multiprogramming is keeping several programs in main memory and executing them concurrently using a single CPU only.

A program loaded into memory and executing is called a process.

If several jobs are ready to be brought into memory and if there is no enough room for all of them, then the system must choose among them. Making this decision is called "Job Scheduling".

If several jobs are ready to run at the same time, the system must choose among them. Making this decision is called "CPU Scheduling"

- **The Evolution Operating System**
    - ➢ Simple Batch Systems.
    - ➢ Multiprogrammed Systems
    - ➢ Time Shared Systems
    - ➢ Personal Computers
    - ➢ Parallel systems
    - ➢ Distributed Systems
    - ➢ Real time systems

**i)Simple Batch Systems.**

- Early computers are very expensive.
- And therefore it was important to maximize processor utilization.
- To improve utilization the concept of Batch OS was developed.
- The first Batch OS was developed in the mid 1950's by General Motors for use on an IBM 701.
- It is refined & implemented on the IBM704.
- IBM 700 Series computers

- 
- Early 1960's , a no. of vendors had developed batch OS for their computer systems.
- Such as IBMSYS,IBM OS for 7090/7094 Computers.
- The idea behind this simple batch processing scheme is the use of piece of a s/w known as Monitor.
- With this type of OS, the user no longer has direct access to the processor.
- Instead, the user submits the job on cards or tape to a computer operator.
- Who batches the jobs together sequentially & places the entire batch on an input device, for use by the monitor.
- To understand how this scheme works?
- Let us look at it from two points of view.
- The monitor
- The processor

- 
- A **punched card** or **punch card** is a piece of stiff paper that can be used to contain digital data represented by the presence or absence of holes in predefined positions.[1]
- Digital data can be used for data processing applications or used to directly control automated machinery.
- The monitor point of view:
- The monitor controls the sequence of events.
- For this , monitor must be in main memory& available for the execution.
- The portion is referred as Resident Monitor.
- The rest of the monitor consists of utilities and common functions that are loaded as subroutines to the user program at the beginning of any job that requires them.

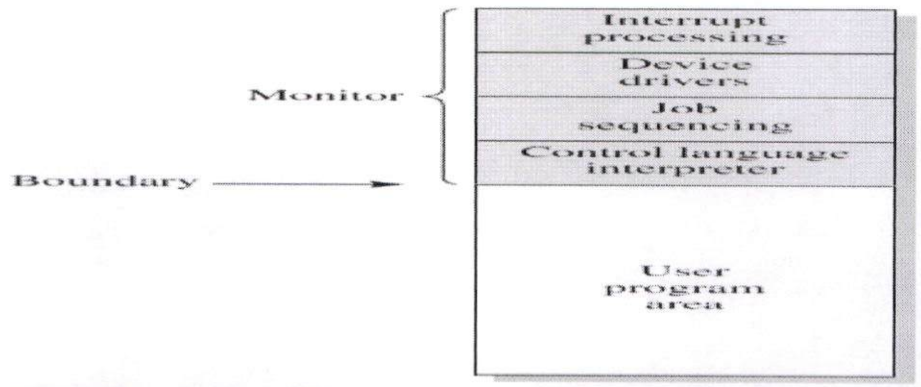


Figure 2.3 Memory Layout for a Resident Monitor

- The monitor reads in jobs one at time form input device(Tape or Card reader).
- As it is read in, the current job is placed in the user program area and control is passed to this job.
- When the job is completed it returns controls to the monitor.
- Which immediately reads in the next job?
- The result of each job is sent to an o/p device such as a printer for delivery to the user.
- 2) Processor point of view:-
- The processor is executing instructions from the portion of main memory containing the monitor.
- The processor will then execute the instructions in the user program until it encounters an ending or error condition.
- The monitor performs a scheduling function.
- A batch of jobs is queued up and jobs are executed.
- With each job, instructions are in a primitive form of JCL (Job control Language).
- This is a special type of programming language used to provide instructions to the monitor.
- For ex. FORTRAN PLUS
- All FORTRAN instructions and data are on a separate punched cards or tape.
- In addition the job includes Job Control Instructions, which are denoted by the beginning $.

```
$JOB
$FTN
•
•   }      FORTRAN instructions
•
$LOAD
$RUN
•
•   }         Data
•
$END
```

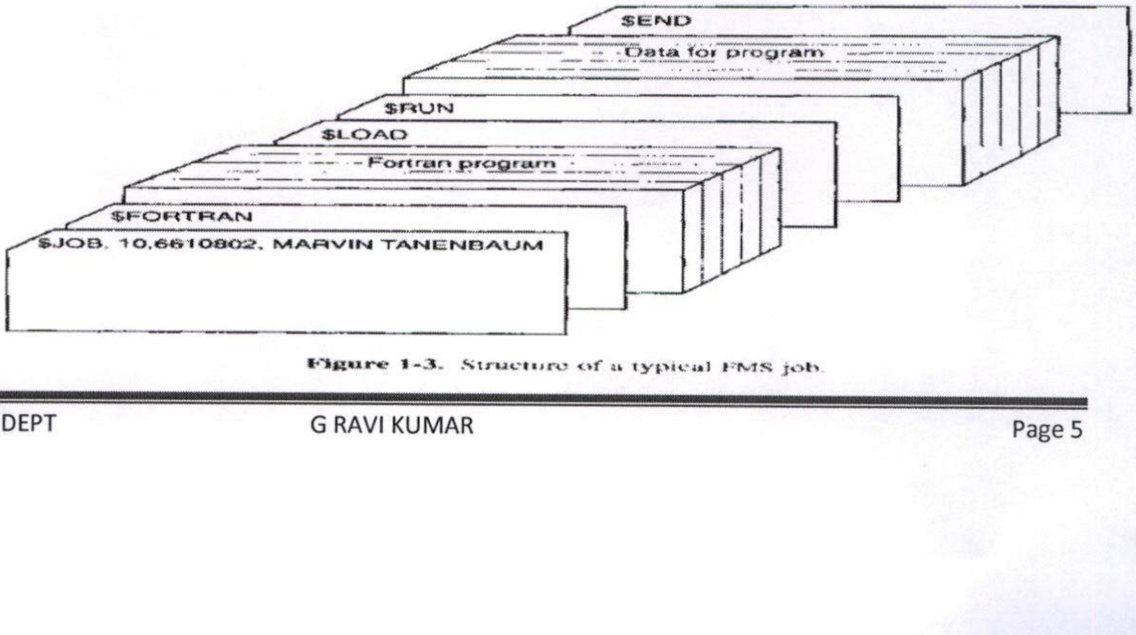
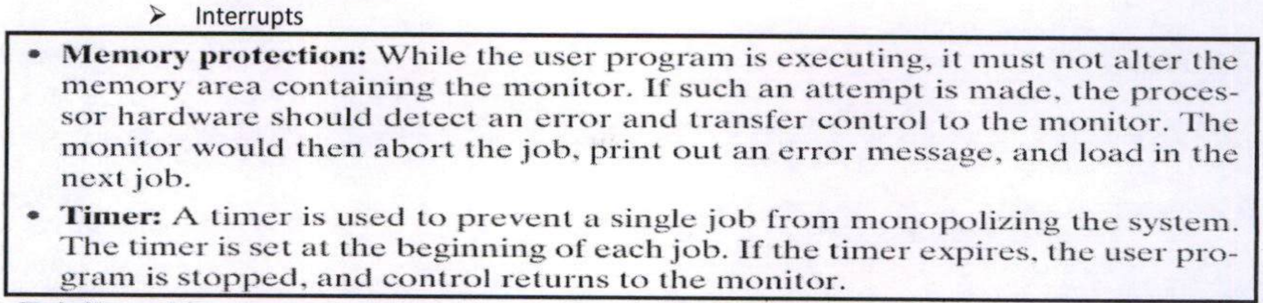
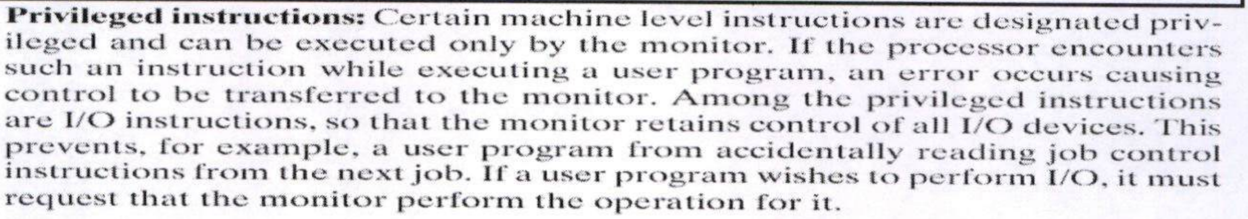


Figure 1-3. Structure of a typical FMS job.

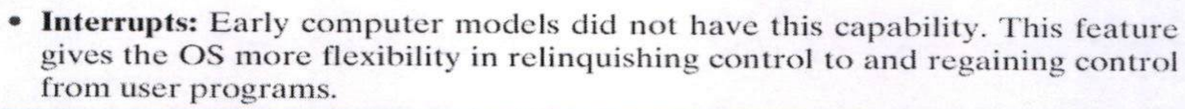
- To execute this job, the monitor reads the $FTN line and loads the appropriate language compiler from its mass storage(tape).
- The compiler translates the user's program into object code, which is stored in memory or storage.
- If it is stored in memory, the operation is referred to as "compile, load and go".
- If it is stored on tape, then the $LOAD instructions is required.
- The monitor invokes the loader, which loads the object program into memory and transfers control to it.
- During the execution of the user program, any input instruction causes one line of data to be read.
- The input instruction in the user program causes an input routine that is part of the OS to be invoked.
- The monitor or batch OS is simply a computer program.
- It relies on the ability of the processor to fetch instructions from various portions of main memory.
- Certain other hardware features are also desirable.
    - ➢ Memory protection
    - ➢ Timer
    - ➢ Privileged instructions
    - ➢ Interrupts



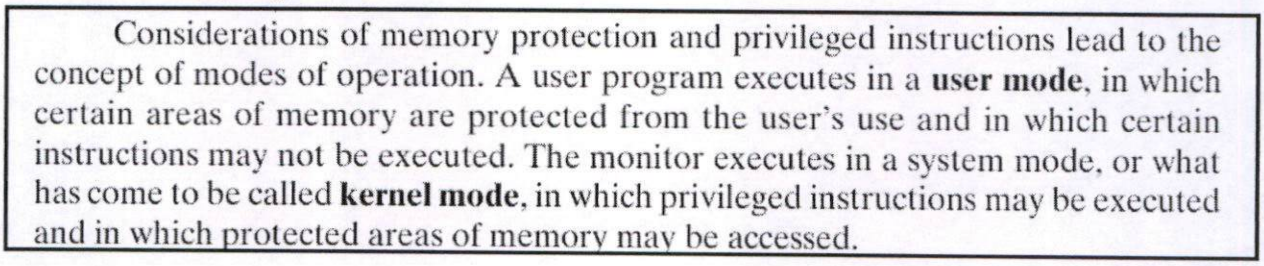
- **Memory protection:** While the user program is executing, it must not alter the memory area containing the monitor. If such an attempt is made, the processor hardware should detect an error and transfer control to the monitor. The monitor would then abort the job, print out an error message, and load in the next job.
- **Timer:** A timer is used to prevent a single job from monopolizing the system. The timer is set at the beginning of each job. If the timer expires, the user program is stopped, and control returns to the monitor.



**Privileged instructions:** Certain machine level instructions are designated privileged and can be executed only by the monitor. If the processor encounters such an instruction while executing a user program, an error occurs causing control to be transferred to the monitor. Among the privileged instructions are I/O instructions, so that the monitor retains control of all I/O devices. This prevents, for example, a user program from accidentally reading job control instructions from the next job. If a user program wishes to perform I/O, it must request that the monitor perform the operation for it.



- **Interrupts:** Early computer models did not have this capability. This feature gives the OS more flexibility in relinquishing control to and regaining control from user programs.



Considerations of memory protection and privileged instructions lead to the concept of modes of operation. A user program executes in a **user mode**, in which certain areas of memory are protected from the user's use and in which certain instructions may not be executed. The monitor executes in a system mode, or what has come to be called **kernel mode**, in which privileged instructions may be executed and in which protected areas of memory may be accessed.

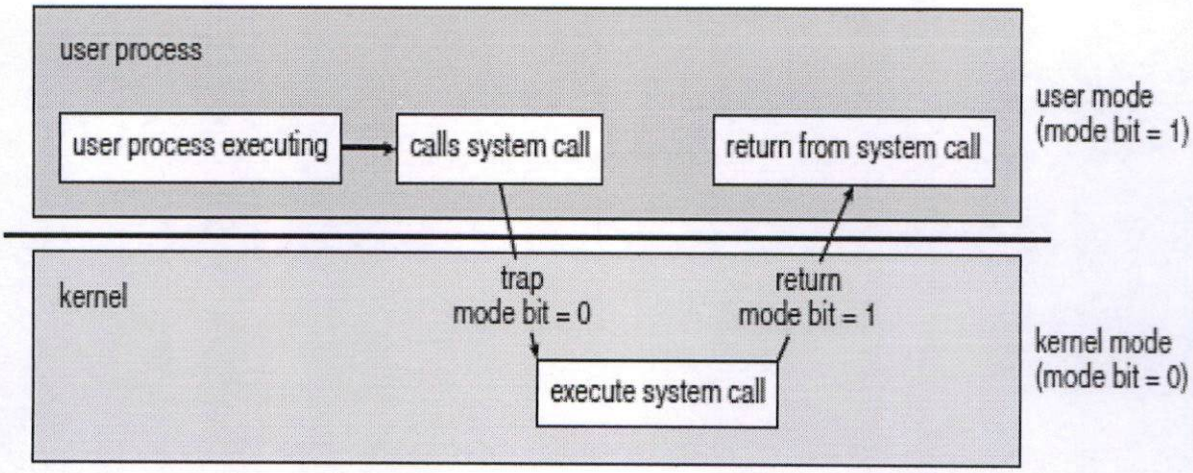


**Figure 1.10** Transition from user to kernel mode.

## OS Operations

1. Dual mode operation
2. Timer

### 1. Dual mode operation

In order to ensure the proper execution of the operating system, we must be able to distinguish between the execution of operating-system code and userdefined code. The approach taken by most computer systems is to provide hardware support that allows us to differentiate among various modes of execution.

For that we need two separate ***modes*** of operation: user mode and kernel mode (also called supervisor mode, system mode, or privileged mode).

A bit, called the mode bit, is added to the hardware of the computer to indicate the current mode: kernel (0) or user (1)

With the mode bit, we can distinguish between a task that is executed on behalf of the operating system and one that is executed on behalf of the user. When the computer system is executing on behalf of a user application, the system is in user mode. However, when a user application requests a service from the operating system (via a system call), the system must transition from user to kernel mode to fulfill the request.

Whenever a trap or interrupt occurs, the hardware switches from user mode to kernel mode (that is, changes the state of the mode bit to 0). Thus, whenever the operating system gains control of the computer, it is in kernel mode. The system always switches to user mode (by setting the mode bit to 1) before passing control to a user program.

The dual mode of operation provides us with the means for protecting the operating system from errant users

### 2.Timer

We must ensure that the operating system maintains control over the CPU. We cannot allow a user program to get stuck in an infinite loop or to fail to call system services and never return control to the operating system. To accomplish this goal, we can use a timer.

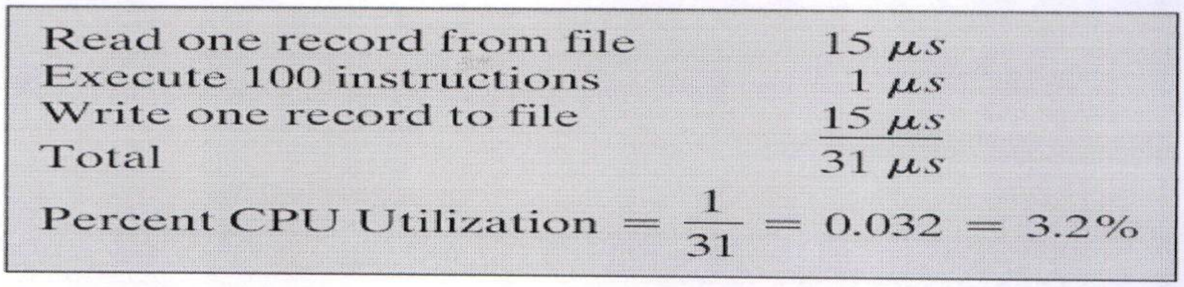
A timer can be set to interrupt the computer after a specified period. The period may be fixed (for example, 1/60 second) or variable (for example, from 1 millisecond to 1 second). A variable timer is generally implemented by a fixed-rate clock and a counter. The operating system sets the counter. Every time the clock ticks, the counter is decremented. When the counter reaches 0, an interrupt occurs.

We can use the timer to prevent a user program from running too long. A simple technique is to initialize a counter with the amount of time that a program is allowed to run.

A program with a 7-minute time limit, for example, would have its counter initialized to 420. Every second, the timer interrupts, and the counter is decremented by 1. As long as the counter is positive, control is returned to the user program. When the counter becomes negative, the operating system terminates the program for exceeding the assigned time limit.

**ii)Multiprogrammed Batch Systems**

- With the automatic job sequencing provided by a simple Batch OS, the processor is often idle.
- The problem is that I/O Devices are slow compared to the processor.
- In the below example, the computer spends 96% of its time waiting for I/O Devices to finish transferring data to and from the file.

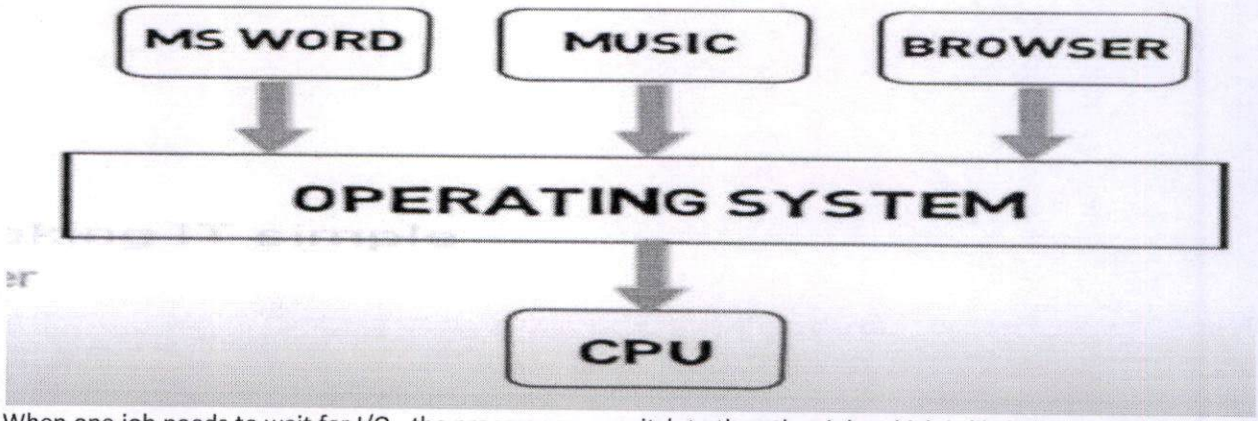


| | |
|---|---|
| Read one record from file | 15 $\mu s$ |
| Execute 100 instructions | 1 $\mu s$ |
| Write one record to file | 15 $\mu s$ |
| Total | 31 $\mu s$ |

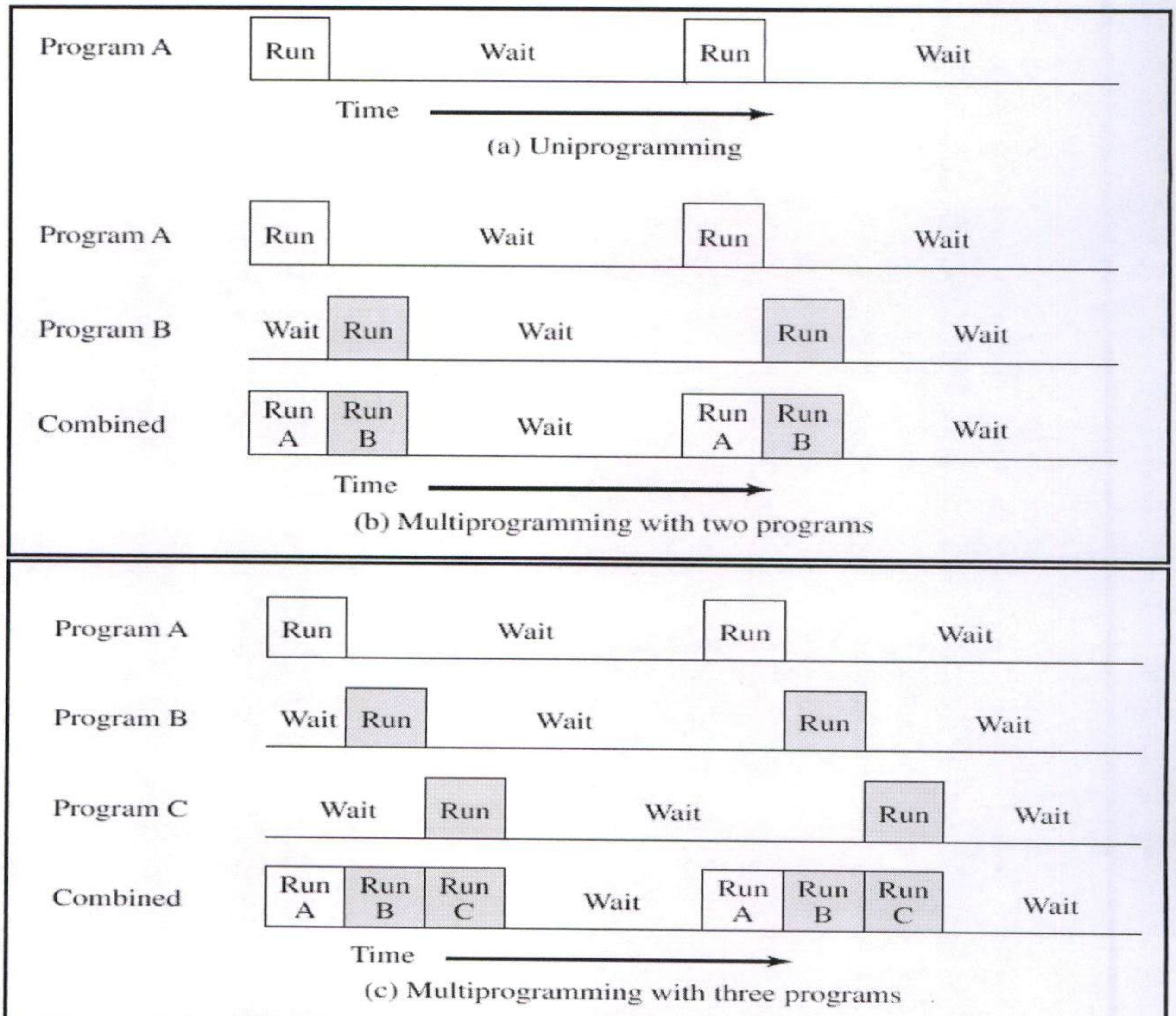
$$\text{Percent CPU Utilization} = \frac{1}{31} = 0.032 = 3.2\%$$

**Figure 2.4 System Utilization Example**

- This situation ,where we have a single program, referred to as uniprogramming.
- The processor spends a certain amount of time executing, until it reaches an I/O instruction, it must then wait until that I/O instruction concludes before proceeding.
- This efficiency is not necessary.
- Suppose that there is room for OS and two user programs.



- When one job needs to wait for I/O , the processor can switch to the other job, which is likely not waiting for I/O.
- Furthermore, we might expand memory to hold three, four or more programs and switch among all of them.
- This approach is known is multiprogramming or multitasking.
- It is the central theme of modern OS.



(a) Uniprogramming

(b) Multiprogramming with two programs

(c) Multiprogramming with three programs

**Figure 2.5 Multiprogramming Example**

To illustrate the benefit of multiprogramming, we give a simple example. Consider a computer with 250 Mbytes of available memory (not used by the OS), a disk, a terminal, and a printer. Three programs, JOB1, JOB2, and JOB3, are submitted for execution at the same time, with the attributes listed in Table 2.1. We assume minimal processor requirements for JOB2 and JOB3 and continuous disk and printer use by JOB3. For a simple batch environment, these jobs will be executed in sequence. Thus, JOB1 completes in 5 minutes. JOB2 must wait until

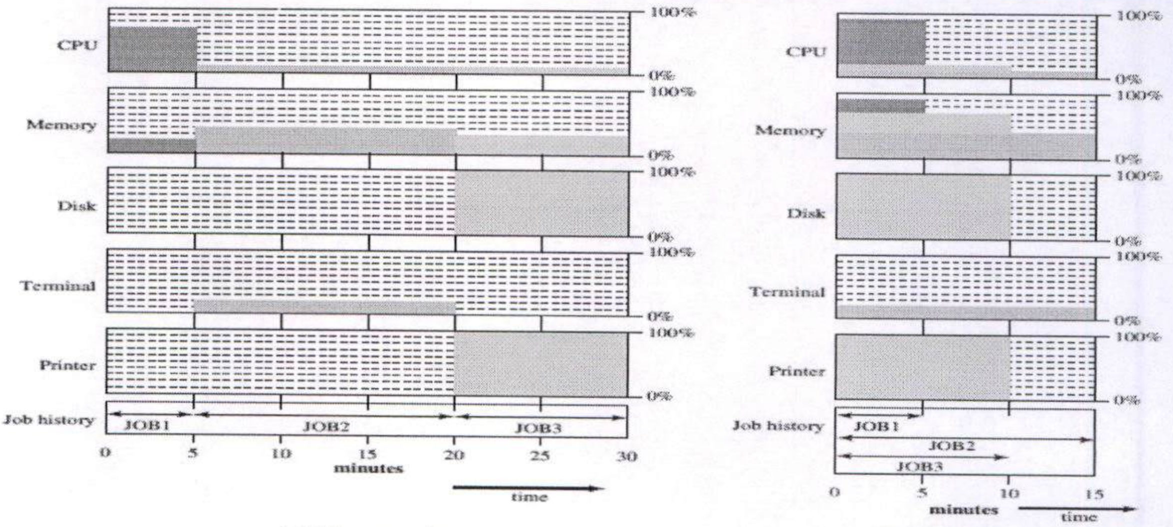
**Table 2.1** Sample Program Execution Attributes

| | JOB1 | JOB2 | JOB3 |
|---|---|---|---|
| **Type of job** | Heavy compute | Heavy I/O | Heavy I/O |
| **Duration** | 5 min | 15 min | 10 min |
| **Memory required** | 50 M | 100 M | 75 M |
| **Need disk?** | No | No | Yes |
| **Need terminal?** | No | Yes | No |
| **Need printer?** | No | No | Yes |

the 5 minutes are over and then completes 15 minutes after that. JOB3 begins after 20 minutes and completes at 30 minutes from the time it was initially submitted. The average resource utilization, throughput, and response times are shown in the uniprogramming column of Table 2.2. Device-by-device utilization is illustrated in Figure 2.6a. It is evident that there is gross underutilization for all resources when averaged over the required 30-minute time period.

**Table 2.2** Effects of Multiprogramming on Resource Utilization

| | Uniprogramming | Multiprogramming |
|---|---|---|
| **Processor use** | 20% | 40% |
| **Memory use** | 33% | 67% |
| **Disk use** | 33% | 67% |
| **Printer use** | 33% | 67% |
| **Elapsed time** | 30 min | 15 min |
| **Throughput** | 6 jobs/hr | 12 jobs/hr |
| **Mean response time** | 18 min | 10 min |



(a) Uniprogramming

(b) Multiprogramming

**Figure 2.6** **Utilization Histograms**

# UNIT-2

**(A30516) OPERATING SYSTEMS**

**UNIT-I**

Operating System Introduction, Structures - Simple Batch, Multi-programmed, Time-shared, Personal Computer, Parallel, Distributed Systems, Real-Time Systems, System components, Operating System services, System Calls.

**UNIT –II**

**Process and CPU Scheduling -** Process concepts and scheduling, Operations on processes, Cooperating Processes, Threads, and Interposes Communication, Scheduling Criteria, Scheduling Algorithms, Multiple -Processor Scheduling.
System call interface for process management-fork, exit, wait, waitpid, exec

## PROCESS CONCEPTS

## I)THE PROCESS:

A Process is program in execution.

For example: A user may run program, word processor, a web browser and email.

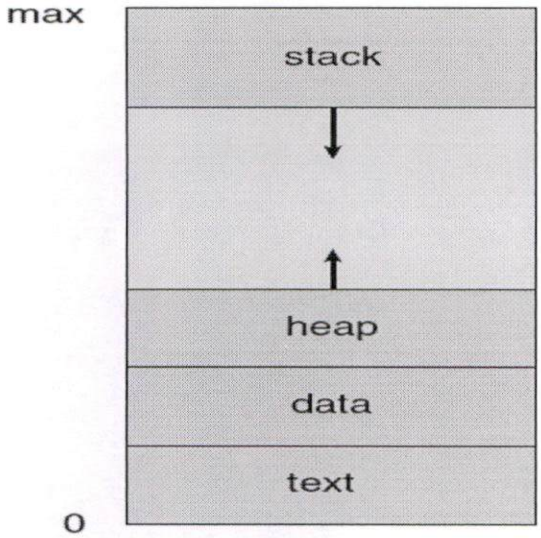


**Figure 3.1** Process in memory.

A process generally also includes the process stack, which contains temporary data(such as function parameters, return address and local variables) and data section which contains global variables.

A process may also include a Heap, which is memory that is dynamically allocated during process runtime.

A program is passive entity such as a file containing a list of instructions stored on disk(executables files).

Whereas a process is an active entity, with a program counter and a set of associated resources.

A program becomes a process when an executable file is loaded into memory.

There are two common techniques for loading executable files are double clicking icon representing the executable file and entering the name of the executable file on command line.

## II)PROCESS STATE DIAGRAM:

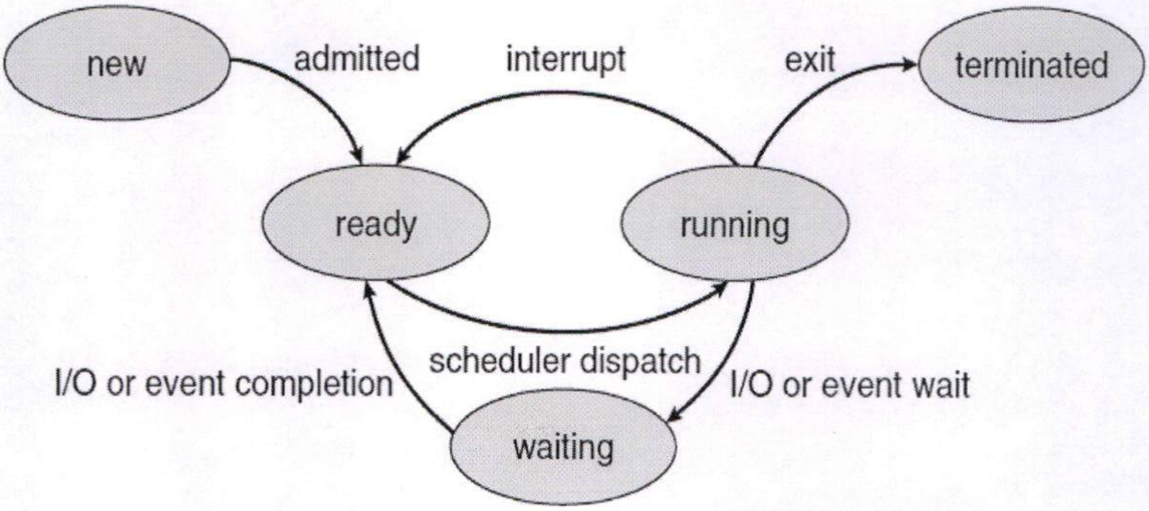


**Figure 3.2** Diagram of process state.

- **New.** The process is being created.
- **Running.** Instructions are being executed.
- **Waiting.** The process is waiting for some event to occur (such as an I/O completion or reception of a signal).
- **Ready.** The process is waiting to be assigned to a processor.
- **Terminated.** The process has finished execution.

As a process executes, it changes state. The state of a process is defined in part by the current activity of that process.

It is important to realize that only one process can be running on any processor at any instant. Many processes may be ready and waiting state.

## PROCESS CONTROL BLOCK.

Each process is represented in the OS by a Process Control Block(PCB), also called as "Task Control Block".

It contains many pieces of information associated with specific process.

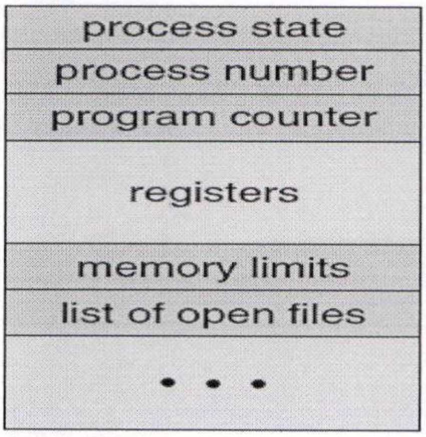


**Figure 3.3** Process control block (PCB).

1. Process state

2. Program Counter

3. CPU Registers

4. CPU Scheduling information

5. Memory management information

6. Accounting information

7. I/O Status information

**Process state**. The state may be new, ready, running, waiting, halted, and so on.

- **Program counter**. The counter indicates the address of the next instruction to be executed for this process.
- **CPU registers**. The registers vary in number and type, depending on the computer architecture. They include accumulators, index registers, stack pointers, and general-purpose registers, plus any condition-code information.
- **CPU-scheduling information**. This information includes a process priority, pointers to scheduling queues, and any other scheduling parameters.
- **Memory-management information**. This information may include such items as the value of the base and limit registers and the page tables, or the segment tables, depending on the memory system used by the operating system .

**Accounting information**. This information includes the amount of CPU and real time used, time limits, account numbers, job or process numbers, and so on.

**I/O status information**. This information includes the list of I/O devices allocated to the process, a list of open files, and so on.

# THREADS

A process is program that performs a single thread of execution.

A thread is a basic unit of CPU utilization; it comprises a thread id, a program counter, a register set and a stack.

It shares with other threads belonging to the same process its code section, data section and other OS resources.

For ex when a process is running a word processor program, a single thread of instructions is being executed. This single thread of control allows the process to perform only one task at a time.

The user cannot simultaneously type in characters and run the spell checker within the same process.

Many modern OS's have extended the process concept to allow a process to have multiple threads of execution and thus to perform more than one task at a time. On a system that supports threads, the pcb is expanded to include information for each thread.

## PROCESS SCHEDULING

## SCHEDULING QUEUES

**Job Queue**: As processes enter the system they are put into the job queue. This consists of all processes in the system.

**Ready Queue**: The processes that are residing in the main memory and they are Ready and waiting to execute are kept on a list called the Ready Queue.

**Device Queue**: The list of processes waiting for a particular I/O device is called a Device Queue.

Each Device has its own Device Queue.

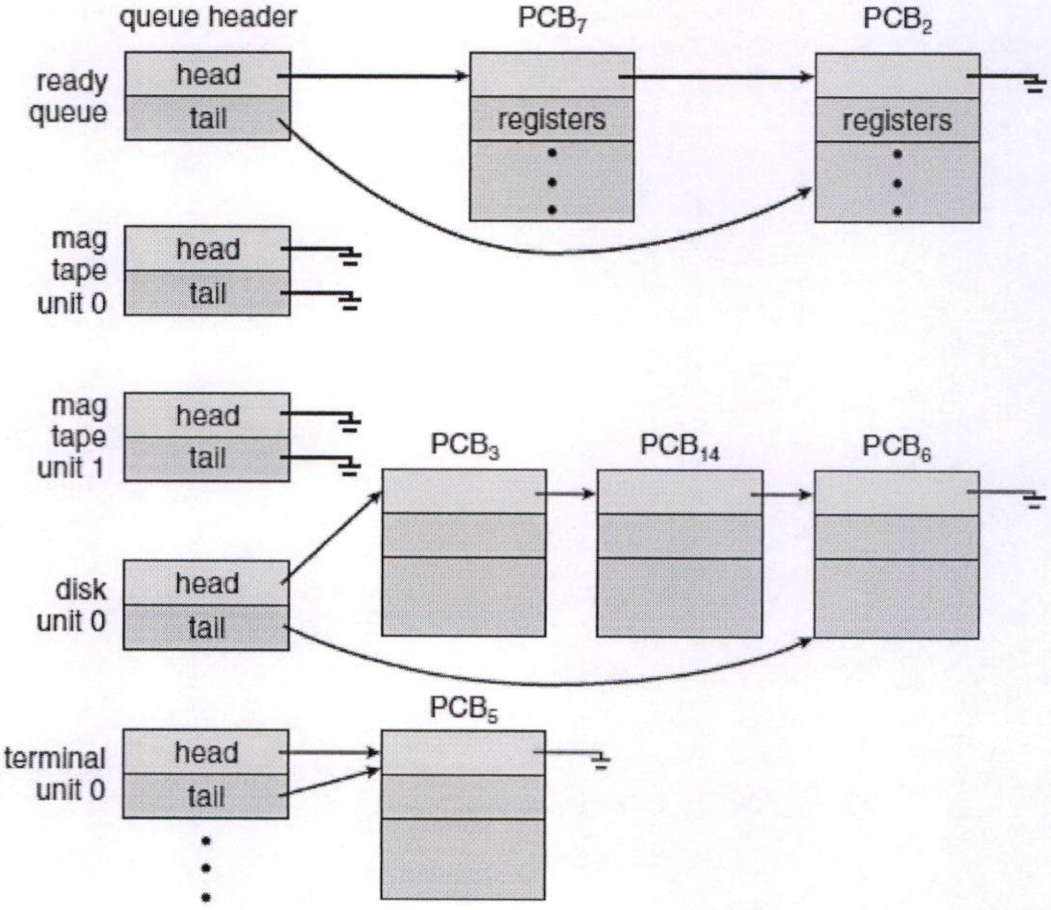


**Figure 3.5** The ready queue and various I/O device queues.

The queue is generally stored as a linked list.

A ready queue header contains pointers to the first and final PCB's in the list.

Each PCB includes a pointer field that points to the next PCB in the Ready Queue.

**Queuing Diagram.**

A new process is initially put in the Ready Queue. And it waits until it is selected for its execution.

Once the process is allocated the CPU and is executing, one of the several events could occur.

A common representation of process scheduling is queuing diagram.

In this each rectangular box represents a queue(Ready and I/O queue).

The circle represents the resources that serve the queues and arrows indicate the flow of process in the system.

1. The process could issue I/O request and then be placed in I/O queue.
2. The process could create a new sub process and wait for the sub process termination.
3. The process could be removed forcefully from the CPU as a result of an interrupt and then be put back in the Ready Queue.

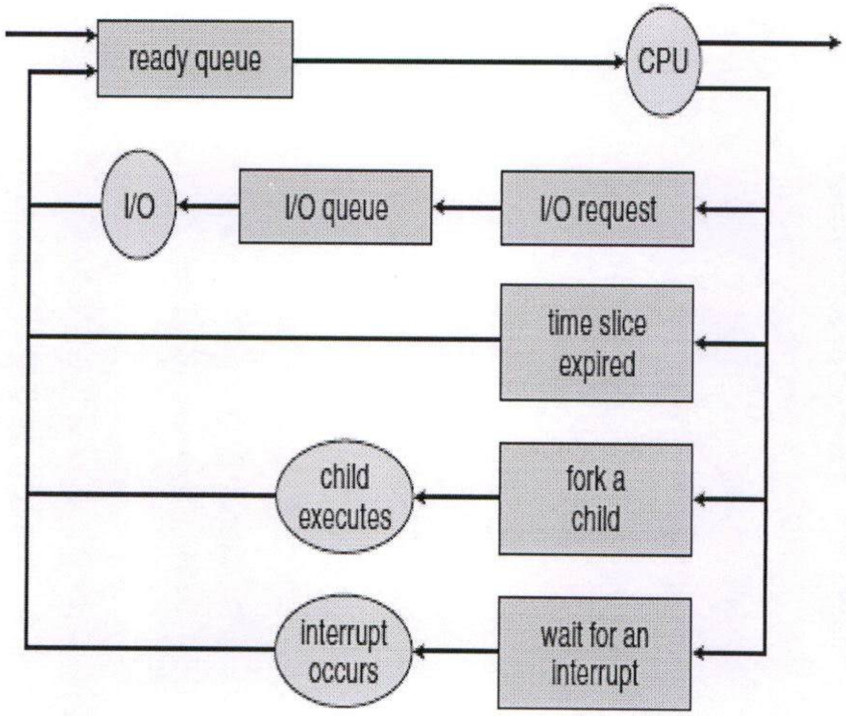
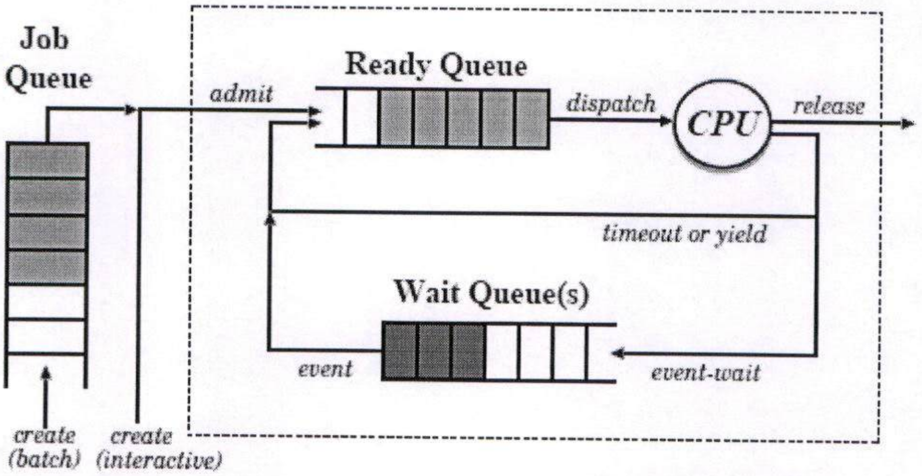


**Figure 3.6** Queueing-diagram representation of process scheduling.



In the first two cases, the process eventually switches from the waiting state to the ready state and is then put back in the ready queue. A process continues this cycle until it terminates, at which time it is removed from all queues and has its PCB and resources deallocated.

## SCHEDULERS

The operating system must select, for scheduling purposes, processes from these queues in some fashion. The selection process is carried out by the appropriate **scheduler**.

1. **LONG TERM SCHEDULERS**
2. **SHORT TERM SCHEDULERS**
3. **MEDIUM TERM SCHEDULERS**

1. Long Term Schedulers Or Job Scheduler

   Which selects processes form this pool and loads them into memory for execution?

2. Short term scheduler or CPU Scheduler:

   Which selects from among the process that is ready to execute and allocate the CPU to one of them?

   The short term scheduler must select a new process for the CPU frequently.

   The long term scheduler will execute much less frequently.

   It is important long term scheduler makes a careful selection.

   In general most processes can be either I/O Bound process or CPU Bound process.

   I/O Bound process is one that spends more of its time doing I/O.

   CPU Bound process more of its time doing computations.

   It is important that the long term scheduler select a good process mix of I/O bound and CPU Bound processes.

# UNIT-III

# Dead locks

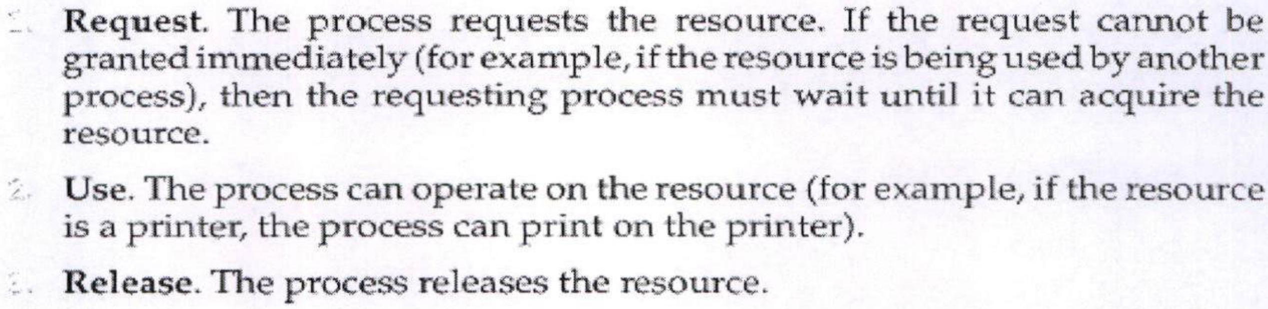
### Definition:

A process requests resources, if the resources are not available at that time, the process enters a waiting state. Sometimes a waiting process is never again able to change state, because the resources it has requested are held by other waiting processes . This situation is called a Dead Lock.

### System model

A system consists of a finite number of resources to be distributed among a number of competing processes.

A process may utilize a resource in only in the following sequence.



1. **Request**. The process requests the resource. If the request cannot be granted immediately (for example, if the resource is being used by another process), then the requesting process must wait until it can acquire the resource.
2. **Use**. The process can operate on the resource (for example, if the resource is a printer, the process can print on the printer).
3. **Release**. The process releases the resource.

A process must request a resource before using it and must release the resource after using it.

## Dead lock characterization

A deadlock situation can arise if the following four conditions hold simultaneously in a system.

1. **Mutual Exclusion**: At least one resource must be held in a non sharable mode, that is, only one process at a time can use the resource. If another process requests that resource, the requesting process must be delayed until the resource has been released.

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

3. **No preemption.** Resources cannot be preempted; that is, a resource can be released only voluntarily by the process holding it, after that process has completed its task.

4. **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 a resource held by $P_2$, ..., $P_{n-1}$ is waiting for a resource held by $P_n$, and $P_n$ is waiting for a resource held by $P_0$.

## Resource Allocation Graph.

Dead locks can be described more precisely in terms of a directed graph called a System resource allocation graph.

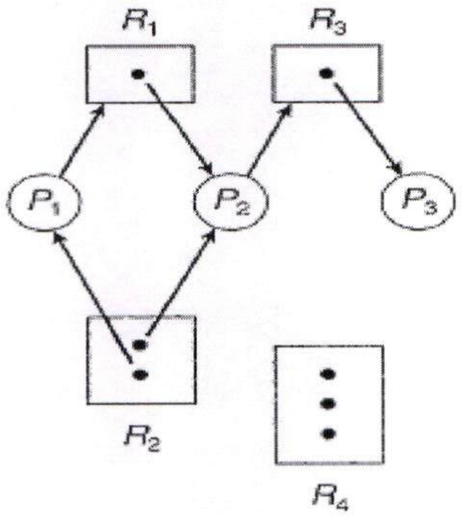


**Figure 7.2** Resource-allocation graph.

A directed edge from process $P_i$ to resource type $R_j$ is deonted $P_i$ -> $R_{j.}$

It signifies that process $P_i$ has requested an instance of resource type $R_j$ and it is called “Request Edge”.

The directed edge from resource type $R_j$ to process $P_i$ is denoted by $R_j$ -> $P_i$.

It signifies that an instance of resource type Rj has been allocated to process Pi and it is called as "Assignment Edge".

We represent each process as a circle. And each resource type as rectangle.

Each resource type may have more than one instance. We represent each instance as a dot(.) with in a rectangle.

Note that a request edge points to only the rectangle Rj where as assignment edge must also designate one of the dots in rectangle.

- The sets *P*, *R*, and *E*:
  - $P = \{P_1, P_2, P_3\}$
  - $R = \{R_1, R_2, R_3, R_4\}$
  - $E = \{P_1 \rightarrow R_1, P_2 \rightarrow R_3, R_1 \rightarrow P_2, R_2 \rightarrow P_2, R_2 \rightarrow P_1, R_3 \rightarrow P_3\}$
- Resource instances:
  - One instance of resource type $R_1$
  - Two instances of resource type $R_2$

  - One instance of resource type $R_3$
  - Three instances of resource type $R_4$
- Process states:
  - Process $P_1$ is holding an instance of resource type $R_2$ and is waiting for an instance of resource type $R_1$.
  - Process $P_2$ is holding an instance of $R_1$ and an instance of $R_2$ and is waiting for an instance of $R_3$.
  - Process $P_3$ is holding an instance of $R_3$.

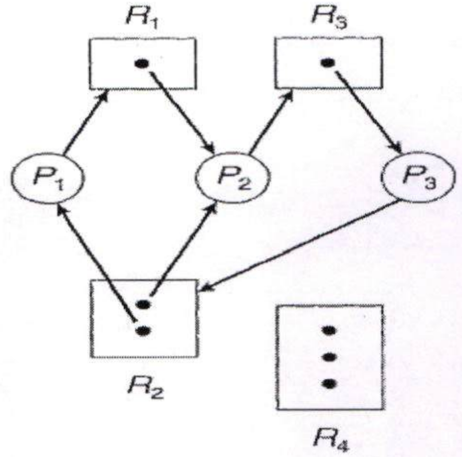


**Figure 7.3** Resource-allocation graph with a deadlock.

$$P_1 \rightarrow R_1 \rightarrow P_2 \rightarrow R_3 \rightarrow P_3 \rightarrow R_2 \rightarrow P_1$$
$$P_2 \rightarrow R_3 \rightarrow P_3 \rightarrow R_2 \rightarrow P_2$$

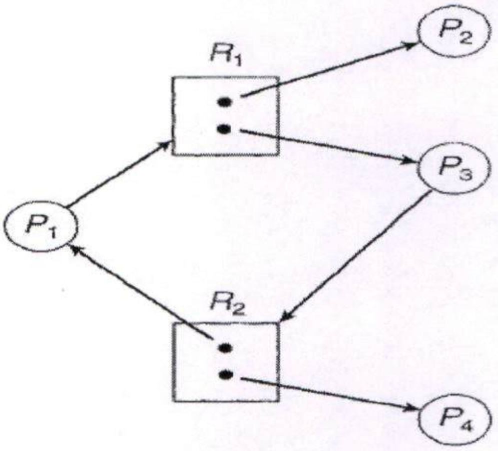


**Figure 7.4** Resource-allocation graph with a cycle but no deadlock.

$$P_1 \rightarrow R_1 \rightarrow P_3 \rightarrow R_2 \rightarrow P_1$$

Now consider this graph, we have a cycle . However there is no dead lock, observe that p4 may release its instance of resource type R2, that resource can then be allocated to P3 breaking the cycle.

If resource allocation graph does not have a cycle, then the system is not dead locked state. If there is a cycle then the system may or may not be in a dead locked state.

## Methods for Handling Dead locks

1. DEAD LOCK PREVENTION OR DEAD LOCK AVOIDANCE
2. DEAD LOCK DETECTION OR RECOVERY FROM DEAD LOCK
3. IGNORE THE DEAD LOCKS

We can deal with the dead lock problem in one of 3 ways.

1. We can use a protocol to prevent or Avoid dead locks ensuring that the system will never enter a dead locked state.
2. We can allow the system to enter a dead locked state, Detect it and Recover
3. We can ignore the problem all together and pretend that Dead locks occur in the system.

   Ex: Unix and Windows

# Dead Lock Prevention

For a dead lock to occur each of the four necessary conditions must hold. By ensuring that atleast one of these conditions cannot hold. We can prevent the occurrence of a Dead lock.

i) **Mutual Exclusion**: This condition must hold non-sharable resources. Example a printer cannot be simultaneously shared by several processes.

Sharable resources do not require mutual exclusive access and thus can not be involved in a dead lock. Example Read only files are a good example of a sharable resources.

ii) **Hold and Wait**: To ensure that, this condition never occurs in systems we must guarantee that whenever a process requests a resource, it does not hold any other resources.

There are 2 protocols

One protocol that can be used requires each process to request and be allocated all its resources before it begins execution. Disadvantage low resource utilization.

An alternative protocol allows a process to request resources only when it has none. A process may request some resources and use them before it can request any additional resources, it must release all the resources that it is currently allocated.

Ex. DVD, HARD DISK,PRINTER

Consider a process that copies data from DVD to a file on disk,sorts the file and then prints the results through a printer. Disadvantage we require all resources, it leads to starvation.

iii) **No Preemption**: This condition is there be no preemption of resources that have , already been allocated. To ensure that this condition does not hold, we can use the following protocol.

If a process is holding some resources and request another resource that can not be immediately allocated to it. Then all resources the process is currently holding are preempted.

If a process requests some resources we first check whether they are available. If they are then allocate them. If they are not, then we check whether they are allocated to some other process. Ie,. Waiting for additional resources, if so we preempt the desired resources from waiting process and allocate them to the requesting process.

iv) **Circular wait**: that this condition never holds is to impose a total ordering of all resource types and to require that each process request resources in an increasing order of enumeration. To illustrate let R={R1,R2,..Rm} be the set of resource types, we assign to each resource type a unique integer number , which allows us to compare 2 resources and to determine whether one precedes another in our ordering . we define a one-one function

F: R->N where N is natural number

F(TAPE DRIVE)=1

F(DISK DRIVE)=5

F(PRINTER)=12

A process can initially request any number of instances of a resource type say $R_i$. After that The process can request instances of resources types $R_j$, iff

$F(R_j) > F(R_i)$

For example a process that wants to use the tape drive and printer at the same time must first request the tape drive and then request the printer.

## Dead lock Avoidance

The various algorithms that we use in this approach differ in the amount and type of information required. The simplest and most useful model requires that each process declare the maximum number of resources of each type that it may need.

Given this a priori information it is possible to construct an algorithm that ensures that the system will never enter the dead lock state.

The dead lock avoidance algorithm dynamically examines the resource allocation state. This resource allocation state is defined by the number of available and allocated resources and the maximum demands of processes.

## i) Safe State

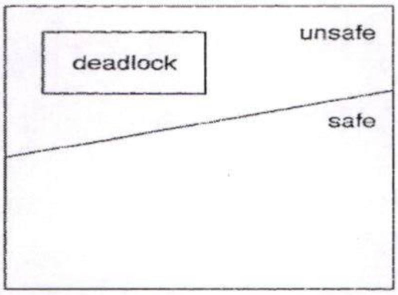


**Figure 7.5** Safe, unsafe, and deadlocked state spaces.

A state is safe if the system can allocate resources to each process in some order and still avoid a dead lock. A system is in a safe state only if there exists a safe sequence. If no such sequence exists then the system state is said to be unsafe. An unsafe state may lead to a dead lock.

For example there are total 12 magnetic tape drives and 3 processes($P_0$,$P_1$,$P_2$)

| | Maximum Needs | Current Needs |
|---|---|---|
| $P_0$ | 10 | 5 |
| $P_1$ | 4 | 2 |
| $P_2$ | 9 | 2 |

Suppose at time zero (to) process P0 is holding 5 tape drives, P1 holds 2 tape drives, P2 holds 2 tape drives.

P1 can immediately be allocated all its tape drives and then return them(5 Available).

Then process P0 can get all its tape drives and return them(10 available)

Finally P2 can get all its tape drives and return them ( the system have 12 tape drives available)

At time t0, the system is in a safe state. The sequence <P1,P0,P2> satisfies the safety condition.

Suppose that at time t1, process P2 request and is allocated one more tape drive. Ie., 2 tape drives given to P1 and 1 tape drive given to P2 then P1 releases only 4 tape drives. That is not sufficient to any of the process nor P0(5 tape drives required) and P2(required 6 tape drives), so we get unsafe condition Ie., deadlock.

Whenever a process requests a resource that is currently available , the system must decide whether the resource can allocated immediately or whether the process must wait. The request is granted only if the allocation leaves the system in a safe state.

## ii) Resource Allocation Graph Algorithm

If we have a resource allocation system with only one instance of each resource type, we can use a variant of the resource allocation graph.

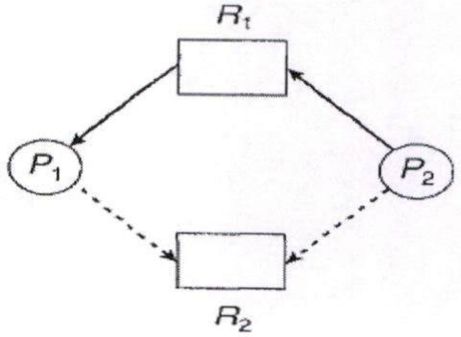


**Figure 7.6** Resource-allocation graph for deadlock avoidance.

We introduce a new type edge called a claim edge.

A claim edge $P_i$->$R_j$ indicates that process Pi may request resource $R_j$ at some time in the future.

This edge similar to a request edge in direction but is represented in the graph by a dashed line.

The claim edge $P_i$-> $R_j$ is converted to a request edge.

Similarly , when a resource $R_j$ is released by $P_i$ , the assignment edge

$R_j$->$P_i$ is reconverted to claim edge $P_i$->$R_j$.

We note that the resources must be claimed a priori in the system.

Before process Pi starts executing, all its claim edges must already appear in the resource allocation graph.

# UNIT-4

## MEMORY MANAGEMENT.

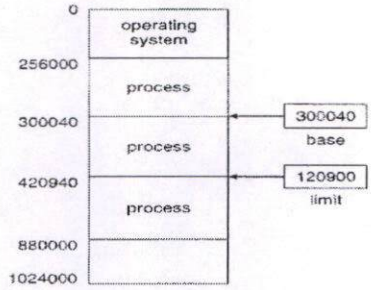


Figure 8.1 A base and a limit register define a logical address space.

Memory consists of a large array of words or bytes.

Each word has its own address.

The main memory and the registers built into the processor itself are the only storage that the cpu can access directly.

Any instruction in execution and any data being used by the instructions must be in one of these direct access storage devices.

If data is not in memory , they must be moved there before the cpu can operate.
We need to make sure that each process has a separate memory space.
We can provide this protection by using two registers as base register and limit registers.
The base register holds the smallest legal physical memory address.
The limit register specifies the size of the range.
The base and limit registers can be loaded only by os, which uses a special privileged instruction.
These instructions can be executed in kernel mode.

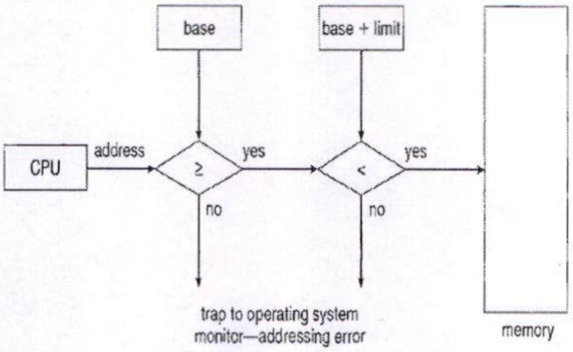


Figure 8.2 Hardware address protection with base and limit registers.

Protection of memory space is accomplished by having the CPU hardware compare every address generated in user mode with registers. An attempt by a program executing in user mode to access OS memory or others user's memory results in a trap to OS, which treats the attempt as a fatal error.

**Address binding**

Usually a program resides on a disk as a binary executable file.
To be executed the program must be brought in to memory and placed within a process.
Most systems allow a user process to reside in any part of the physical memory.
Before being executed a user program, the address may be represented in different ways.
Addresses in the source program are generally symbolic such as count.
A compiler will bind these symbolic addresses to relocatable addresses.
The linkage editor or loader will inturn point the relocatable addresses to absolute addresses .

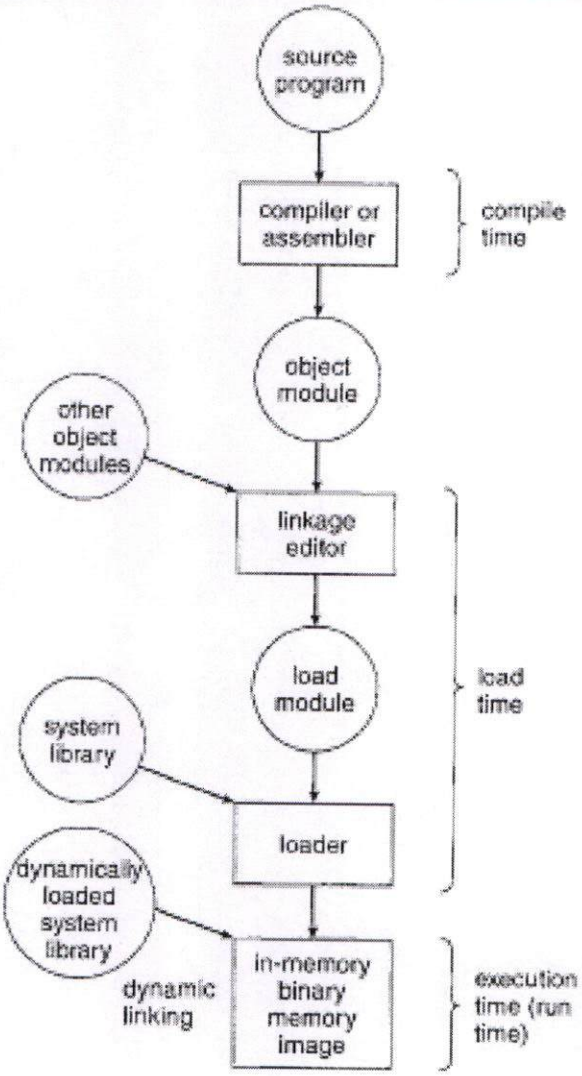


Figure 8.3 Multistep processing of a user program.

The binding of instructions and data to memory addresses can be done in the following way

i) Compile time

ii) Load time

iii) Execution time

Logical and physical address space

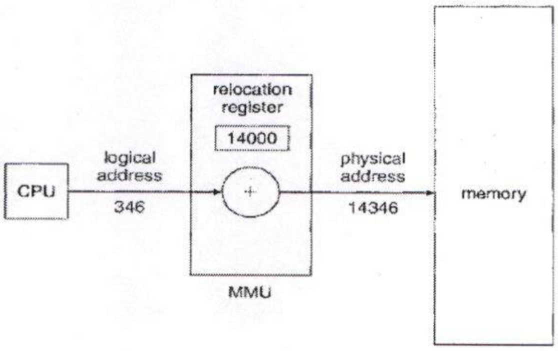


**Figure 8.4** Dynamic relocation using a relocation register.

An address generated by CPU is referred to as a logical address or virtual address.

Where as an address seen by memory unit is referred to as a physical address.

The set of all logical addresses generated by a program is a logical address space.

The set of all physical addresses corresponding to logical addresses is a physical address space.

The run time mapping from virtual to physical address is done by a h/w device called the memory management unit.

The base register is called a relocation register ,the value in the relocation register is added to every address generated by a user process at the time the address is sent to memory.

For example if the base at 14000, then an attempt by the user to address location 0 is dynamically relocated to location 14000.

An access to location 346 is mapped to location 14346.

## Swapping

A process must be in memory to be executed

However it can be swapped temporarily out of memory to a backing store and then brought back into memory for continued execution.

For example

Assume a multiprogramming environment with a round robin cpu scheduling algorithm.

When a quantum expires, the memory manager will start to swap out the process that just finished and to swap another process into memory space that has been fired.

When each process finishes its quantum, it will be swapped with another process.

A variant of this swapping policy is used for priority based scheduling algorithms.

If a higher priority process arrives and wants service, the memory manager can swap out the lower priority process and then load and execute the higher priority process.

When the higher priority process finishes, the lower priority process can be swapped back in and continued.

This variant of swapping is sometimes called roll out, roll in.

Swapping requires a backing store.

The backing store is commonly a fast disk.

It must be large enough to accommodate copies of all memory images for all users, and it must provide direct access to these memory images.

Swapping is used in many versions of UNIX.

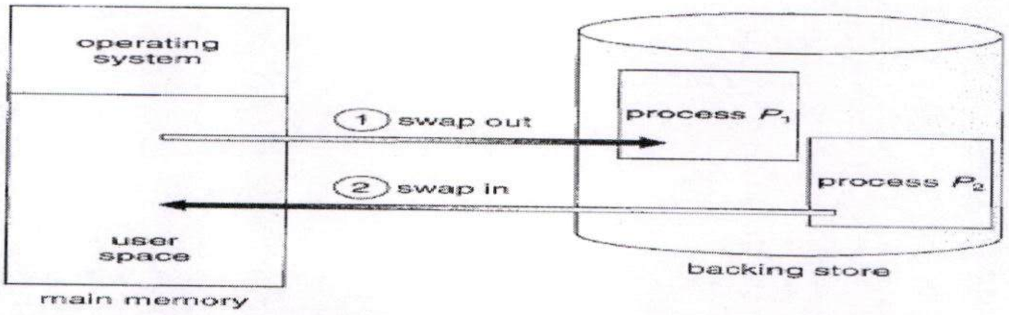


Figure 8.5 Swapping of two processes using a disk as a backing store.

## I.Contiguous memory allocation

The main memory must accommodate both the os and the various user processes.

The memory is divided into two partitions

One for the resident os and for the user processes.

How to allocate available memory to the processes that are in the input queue waiting to be brought into memory in Contiguous memory allocation, each process in a single contiguous section of memory.

i) **Memory mapping and protection**

We can provide by using a Relocation Register and Limit Register.

The relocation register contains the value of the smallest physical address

The limit register contains the range of logical addresses.

(For example relocation register=100040 and limit=74600)

logical address must be less than the limit register.

When the CPU scheduler selects a process for execution, the dispatcher
loads the relocation and limit registers with the correct values.

Because every address generated by a CPU is checked against these registers, we can protect both the operating system and the other users'
programs and data from being modified by this running process.

The MMU maps the logical address dynamically by adding the value in the relocation register.

The mapped address is sent to memory.

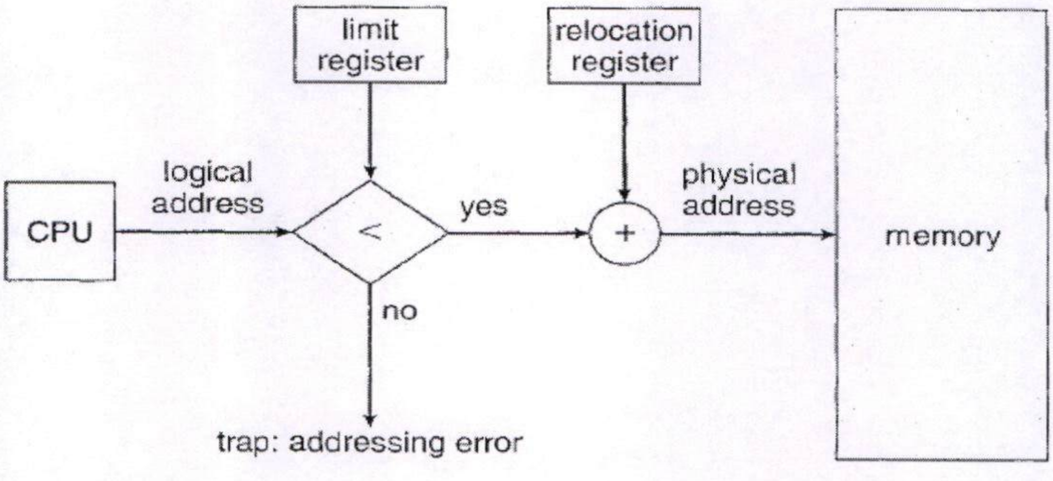


**Figure 8.6** Hardware support for relocation and limit registers.

### ii) Memory Allocation

One of the simplest methods for allocating memory is to divide memory into several **fixed sized partitions.**

Each partition may contain exactly one process.

Thus the degree of multiprogramming is bound by the number of partitions.

In this multiple partition method, when a partition is free, a process is selected from the input queue and is loaded into the free partition.

When the process terminates the partition becomes available for another process.

In the **variable partition** scheme, the OS keeps a table indicating which parts of memory are available and which are occupied.

Initially, all memory is available for user processes and is considered one large block of available of memory, a **hole.**

Eventually, as you will see memory contains a set of holes of various sizes.

Memory is allocated to processes until, finally the memory requirements of the next process cannot be satisfied.

That is no available block of memory (or hole) is large enough to hold that process.

The OS can then wait until a large enough block is available or it can skip down the input queue to see whether the smaller memory requirements of some other process can be met.

The system may need to check whether there are processes waiting for memory could satisfy the demands of any of these waiting processes.

This procedure is **Dynamic storage allocation problem.**

Which concerns how to satisfy a request of size n from a list of free holes?

There are many solutions to this problem.

The First fit, Best fit, Worst fit strategies are used to select a free hole from the set of available holes.

**First fit**: allocate the first hole that is big enough. Searching can start either at the beginning of the set of holes or at the location where the previous first fit search ended.

We can stop searching as soon as we find a free hole that is large enough.

**Best fit**: allocate the smallest hole that is big enough. We must search the entire list, unless the list is ordered by size.

This strategy produces the smallest leftover hole.

**Worst Fit**: allocate the largest hole. Again we must search the entire list, unless it is sorted by size.

This strategy produces the largest leftover hole, which may be more useful than the smaller leftover hole from a best fit approach.

Simulations have shown that both first fit and best fit are better than worst fit in terms of decreasing time and storage utilization. Neither first fit nor best fit is clearly better than the other in terms of storage utilization, but first fit is generally faster.

**Fragmentation**

After some time that processes can not be allocated to memory because of small size and memory blocks remains unused. This problem is called **fragmentation.**

Memory fragmentation can be **internal as well as external.**

First fit and best fit strategies for memory allocation suffer from external fragmentation.

As processes are loaded and removed from memory, the free memory space is broken into little pieces.

**External fragmentation** exists when there is enough total memory space to satisfy a request but the available spaces are not contiguous, storage is fragmented into a large number of small holes.

**Internal fragmentation**

The memory allocated to a process may be slightly larger than the requested memory. The difference between these two numbers is **internal fragmentation.**

Unused memory that is internal to a partition.

One solution to the problem of external fragmentation is **compaction.**

The goal is to shuffle the memory contents so as to place all free memory together in one large block

Another solution is to the external fragmentation problem is to permit the logical address space of the processes to be non contiguous.

# 2.PAGING

It is a Memory management scheme that permits the physical address space of a process to be non contiguous.

It avoids external fragmentation and the need for compaction.

The basic method for implementing paging involves breaking physical memory into fixed sized blocks called frames(F) and breaking logical memory into blocks of same size called pages (P).

When a process to be executed, its pages are loaded into any available memory frames from their source(Backing Store).

The backing store is divided into fixed sized blocks that are of same size as the memory frames.

The hardware supports for paging is described in the following figure.

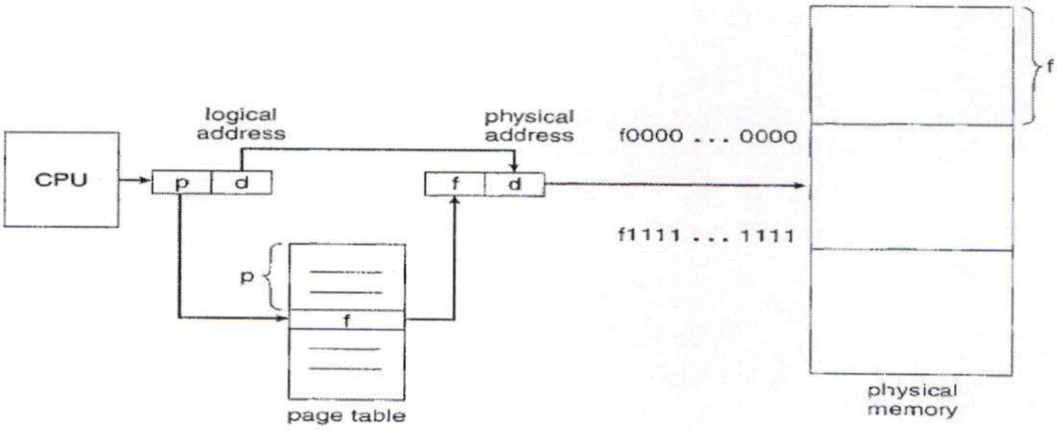


Figure 8.7 Paging hardware.

Every address generated by CPU is divided into two parts.

i) Page number(P)

ii) Page offset (d)

The page number is used as an index into page table .

The page table contains the base address of each page in physical memory.

This base address is combined with the page offset to define the physical memory address ie.. sent to memory unit.

The paging model of memory is shown in figure.

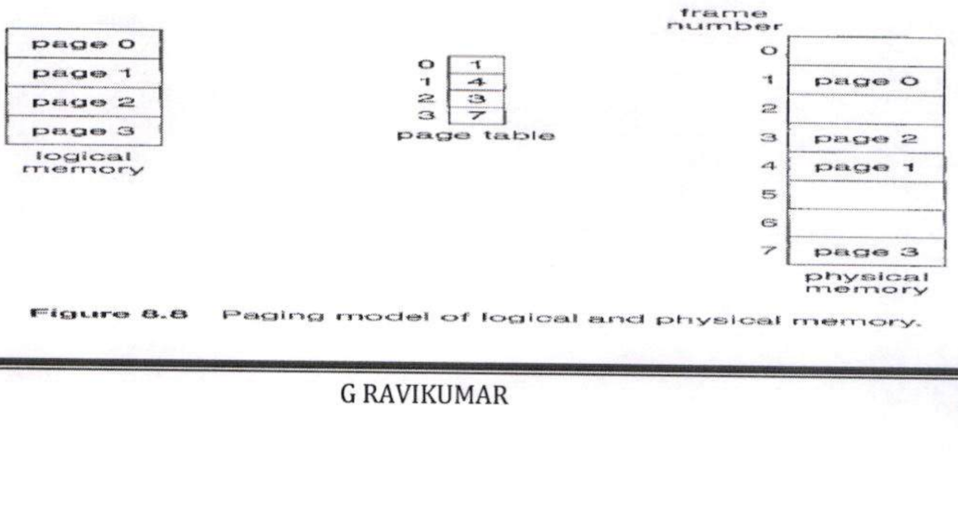
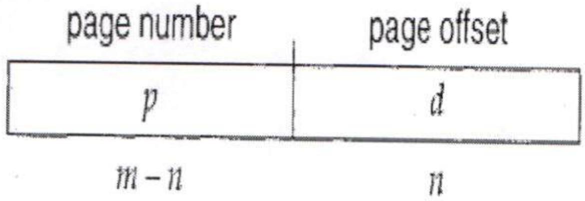


Figure 8.8 Paging model of logical and physical memory.

The page size or frame size is defined by the hardware.

The size of a page is typically a power of 2 varying between 512 bytes and 16 MB per page depending on computer architecture.

The selection of a power of 2 as a page size makes the translation of a logical address into a page number and page offset particularly easy. If the size of the logical address space is $2^m$, and a page size is $2^n$ addressing units (bytes or words) then the high-order *m*- *n* bits of a logical address designate the page number, and the *n* low-order bits designate the page offset. Thus, the logical address is as follows:



where $p$ is an index into the page table and $d$ is the displacement within the page.

As a concrete (although minuscule) example, consider the memory in Figure 8.9. Here, in the logical address, $n$= 2 and $m$ = 4. Using a page size of 4 bytes and a physical memory of 32 bytes (8 pages), we show how the user's view of memory can be mapped into physical memory. Logical address 0 is page 0, offset 0. Indexing into the page table, we find that page 0 is in frame 5. Thus, logical address 0 maps to physical address 20 [= (5 × 4) + 0]. Logical address 3 (page 0, offset 3) maps to physical address 23 [= (5 × 4) + 3]. Logical address 4 is page 1, offset 0; according to the page table, page 1 is mapped to frame 6. Thus, logical address 4 maps to physical address 24 [= (6 × 4) + 0]. Logical address 13 maps to physical address 9.

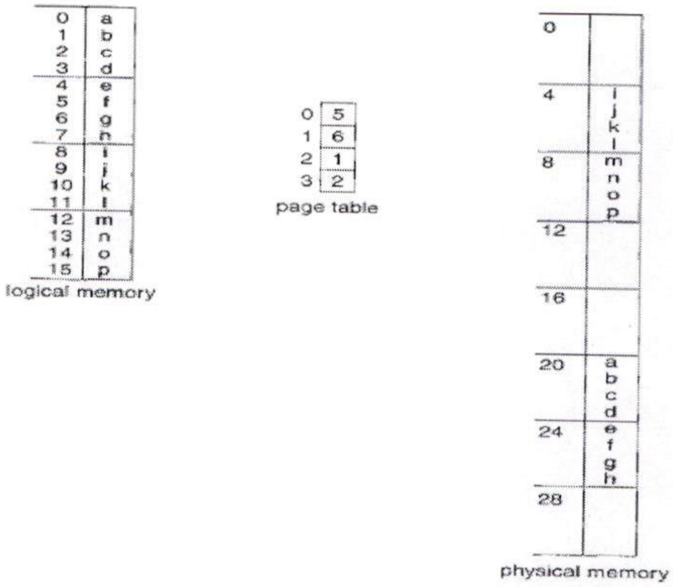


**Figure 8.9** Paging example for a 32-byte memory with 4-byte pages.

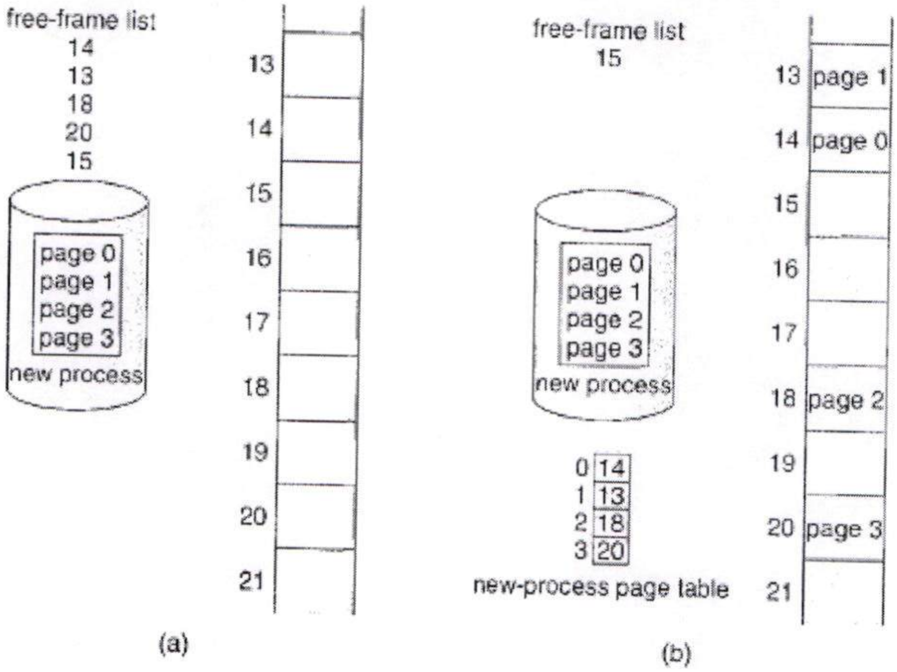


**Figure 8.10** Free frames (a) before allocation and (b) after allocation.

# UNIT - 5

# File-System Implementation

### 12.1 File-System Structure

- Hard disks have two important properties that make them suitable for secondary storage of files in file systems: (1) Blocks of data can be rewritten in place, and (2) they are direct access, allowing any block of data to be accessed with only ( relatively ) minor movements of the disk heads and rotational latency. ( See Chapter 12 )
- Disks are usually accessed in physical blocks, rather than a byte at a time. Block sizes may range from 512 bytes to 4K or larger.
- File systems organize storage on disk drives, and can be viewed as a layered design:
    - At the lowest layer are the physical devices, consisting of the magnetic media, motors & controls, and the electronics connected to them and controlling them. Modern disk put more and more of the electronic controls directly on the disk drive itself, leaving relatively little work for the disk controller card to perform.
    - ***I/O Control*** consists of ***device drivers***, special software programs ( often written in assembly ) which communicate with the devices by reading and writing special codes directly to and from memory addresses corresponding to the controller card's registers. Each controller card ( device ) on a system has a different set of addresses ( registers, a.k.a. ***ports*** ) that it listens to, and a unique set of command codes and results codes that it understands.
    - The ***basic file system*** level works directly with the device drivers in terms of retrieving and storing raw blocks of data, without any consideration for what is in each block. Depending on the system, blocks may be referred to with a single block number, ( e.g. block # 234234 ), or with head-sector-cylinder combinations.
    - The ***file organization module*** knows about files and their logical blocks, and how they map to physical blocks on the disk. In addition to translating from logical to physical blocks, the file organization module also maintains the list of free blocks, and allocates free blocks to files as needed.
    - The ***logical file system*** deals with all of the meta data associated with a file ( UID, GID, mode, dates, etc ), i.e. everything about the file except the data itself. This level manages the directory structure and the

mapping of file names to ***file control blocks, FCBs***, which contain all of the meta data as well as block number information for finding the data on the disk.

- The layered approach to file systems means that much of the code can be used uniformly for a wide variety of different file systems, and only certain layers need to be filesystem specific. Common file systems in use include the UNIX file system, UFS, the Berkeley Fast File System, FFS, Windows systems FAT, FAT32, NTFS, CD-ROM systems ISO 9660, and for Linux the extended file systems ext2 and ext3 ( among 40 others supported. )

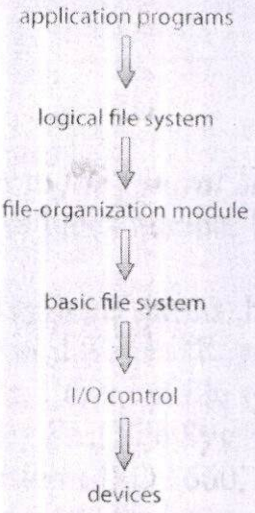


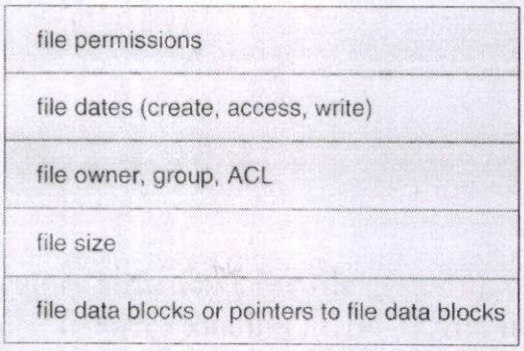
**Figure 12.1 - Layered file system.**

### 12.2 File-System Implementation

#### 12.2.1 Overview

- File systems store several important data structures on the disk:
  - A ***boot-control block***, ( per volume ) a.k.a. the ***boot block*** in UNIX or the ***partition boot sector*** in Windows contains information about how to boot the system off of this disk. This will generally be the first sector of the volume if there is a bootable system loaded on that volume, or the block will be left vacant otherwise.
  - A ***volume control block,*** ( per volume ) a.k.a. the ***master file table*** in UNIX or the ***superblock*** in Windows, which contains information such as the partition table, number of blocks on each filesystem, and pointers to free blocks and free FCB blocks.

- A directory structure ( per file system ), containing file names and pointers to corresponding FCBs. UNIX uses inode numbers, and NTFS uses a ***master file table.***
- The ***File Control Block, FCB,*** ( per file ) containing details about ownership, size, permissions, dates, etc. UNIX stores this information in inodes, and NTFS in the master file table as a relational database structure.



| |
|---|
| file permissions |
| file dates (create, access, write) |
| file owner, group, ACL |
| file size |
| file data blocks or pointers to file data blocks |

**Figure 12.2 - A typical file-control block.**

- There are also several key data structures stored in memory:
  - An in-memory mount table.
  - An in-memory directory cache of recently accessed directory information.
  - ***A system-wide open file table***, containing a copy of the FCB for every currently open file in the system, as well as some other related information.
  - ***A per-process open file table,*** containing a pointer to the system open file table as well as some other information. ( For example the current file position pointer may be either here or in the system file table, depending on the implementation and whether the file is being shared or not. )
- Figure 12.3 illustrates some of the interactions of file system components when files are created and/or used:
  - When a new file is created, a new FCB is allocated and filled out with important information regarding the new file. The appropriate directory is modified with the new file name and FCB information.
  - When a file is accessed during a program, the open( ) system call reads in the FCB information from disk, and stores it in the system-wide open file table. An entry is added to the per-process open file table referencing the system-wide table, and an index into the per-process table is returned by the open( ) system call.

UNIX refers to this index as a ***file descriptor***, and Windows refers to it as a ***file handle***.

- If another process already has a file open when a new request comes in for the same file, and it is sharable, then a counter in the system-wide table is incremented and the per-process table is adjusted to point to the existing entry in the system-wide table.
- When a file is closed, the per-process table entry is freed, and the counter in the system-wide table is decremented. If that counter reaches zero, then the system wide table is also freed. Any data currently stored in memory cache for this file is written out to disk if necessary.

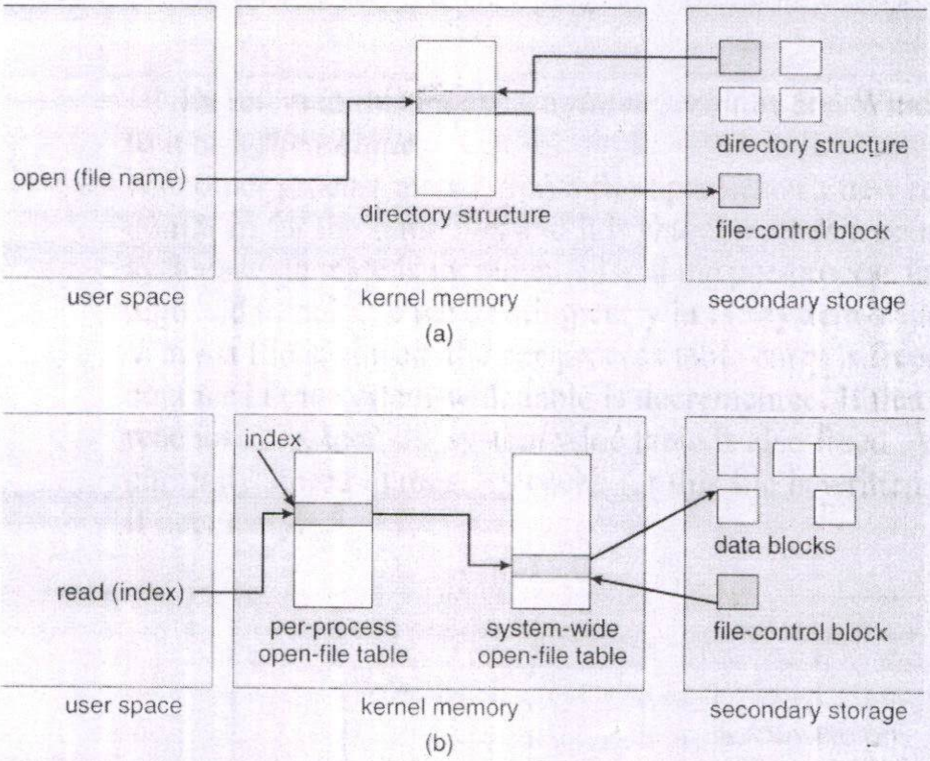


**Figure 12.3 - In-memory file-system structures. (a) File open. (b) File read.**

### 12.2.2 Partitions and Mounting

- Physical disks are commonly divided into smaller units called partitions. They can also be combined into larger units, but that is most commonly done for RAID installations and is left for later chapters.
- Partitions can either be used as raw devices ( with no structure imposed upon them ), or they can be formatted to hold a filesystem ( i.e. populated with FCBs and initial directory structures as appropriate. ) Raw partitions are generally used for swap space, and may also be used

for certain programs such as databases that choose to manage their own disk storage system. Partitions containing filesystems can generally only be accessed using the file system structure by ordinary users, but can often be accessed as a raw device also by root.

- The boot block is accessed as part of a raw partition, by the boot program prior to any operating system being loaded. Modern boot programs understand multiple OSes and filesystem formats, and can give the user a choice of which of several available systems to boot.
- The ***root partition*** contains the OS kernel and at least the key portions of the OS needed to complete the boot process. At boot time the root partition is mounted, and control is transferred from the boot program to the kernel found there. ( Older systems required that the root partition lie completely within the first 1024 cylinders of the disk, because that was as far as the boot program could reach. Once the kernel had control, then it could access partitions beyond the 1024 cylinder boundary. )
- Continuing with the boot process, additional filesystems get mounted, adding their information into the appropriate mount table structure. As a part of the mounting process the file systems may be checked for errors or inconsistencies, either because they are flagged as not having been closed properly the last time they were used, or just for general principals. Filesystems may be mounted either automatically or manually. In UNIX a mount point is indicated by setting a flag in the in-memory copy of the inode, so all future references to that inode get re-directed to the root directory of the mounted filesystem.

### 12.2.3 Virtual File Systems

- ***Virtual File Systems, VFS***, provide a common interface to multiple different filesystem types. In addition, it provides for a unique identifier ( vnode ) for files across the entire space, including across all filesystems of different types. ( UNIX inodes are unique only across a single filesystem, and certainly do not carry across networked file systems. )
- The VFS in Linux is based upon four key object types:
  - The ***inode*** object, representing an individual file
  - The ***file*** object, representing an open file.
  - The ***superblock*** object, representing a filesystem.
  - The ***dentry*** object, representing a directory entry.
- Linux VFS provides a set of common functionalities for each filesystem, using function pointers accessed through a table. The same functionality is accessed through the same table position for all filesystem types, though the actual functions pointed to by the pointers may be filesystem-

specific. See /usr/include/linux/fs.h for full details. Common operations provided include open( ), read( ), write( ), and mmap( ).

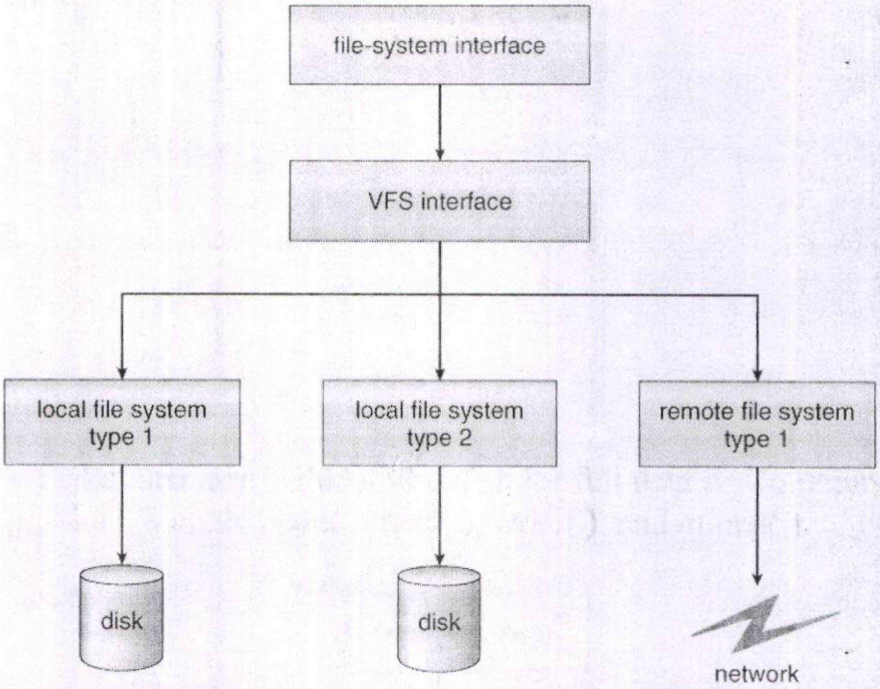


**Figure 12.4 - Schematic view of a virtual file system.**

## 12.3 Directory Implementation

- Directories need to be fast to search, insert, and delete, with a minimum of wasted disk space.

### 12.3.1 Linear List

- A linear list is the simplest and easiest directory structure to set up, but it does have some drawbacks.
- Finding a file ( or verifying one does not already exist upon creation ) requires a linear search.
- Deletions can be done by moving all entries, flagging an entry as deleted, or by moving the last entry into the newly vacant position.
- Sorting the list makes searches faster, at the expense of more complex insertions and deletions.
- A linked list makes insertions and deletions into a sorted list easier, with overhead for the links.
- More complex data structures, such as B-trees, could also be considered.

### 12.3.2 Hash Table

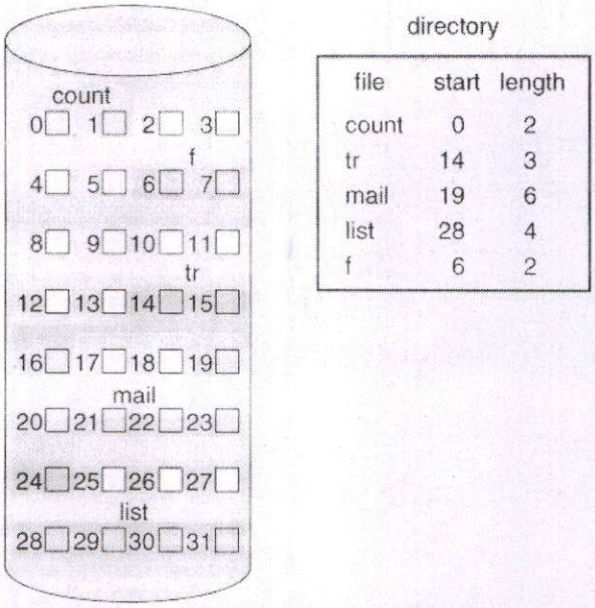
- A hash table can also be used to speed up searches.
- Hash tables are generally implemented ***in addition to*** a linear or other structure

## 12.4 Allocation Methods

- There are three major methods of storing files on disks: contiguous, linked, and indexed.

### 12.4.1 Contiguous Allocation

- ***Contiguous Allocation*** requires that all blocks of a file be kept together contiguously.
- Performance is very fast, because reading successive blocks of the same file generally requires no movement of the disk heads, or at most one small step to the next adjacent cylinder.
- Storage allocation involves the same issues discussed earlier for the allocation of contiguous blocks of memory ( first fit, best fit, fragmentation problems, etc. ) The distinction is that the high time penalty required for moving the disk heads from spot to spot may now justify the benefits of keeping files contiguously when possible.
- ( Even file systems that do not by default store files contiguously can benefit from certain utilities that compact the disk and make all files contiguous in the process. )
- Problems can arise when files grow, or if the exact size of a file is unknown at creation time:
  - Over-estimation of the file's final size increases external fragmentation and wastes disk space.
  - Under-estimation may require that a file be moved or a process aborted if the file grows beyond its originally allocated space.
  - If a file grows slowly over a long time period and the total final space must be allocated initially, then a lot of space becomes unusable before the file fills the space.
- A variation is to allocate file space in large contiguous chunks, called ***extents.*** When a file outgrows its original extent, then an additional one is allocated. ( For example an extent may be the size of a complete track or even cylinder, aligned on an appropriate track or cylinder boundary. ) The high-performance files system Veritas uses extents to optimize performance.



| file | start | length |
|---|---|---|
| count | 0 | 2 |
| tr | 14 | 3 |
| mail | 19 | 6 |
| list | 28 | 4 |
| f | 6 | 2 |

**Figure 12.5 - Contiguous allocation of disk space.**

### 12.4.2 Linked Allocation

- Disk files can be stored as linked lists, with the expense of the storage space consumed by each link. ( E.g. a block may be 508 bytes instead of 512. )
- Linked allocation involves no external fragmentation, does not require pre-known file sizes, and allows files to grow dynamically at any time.
- Unfortunately linked allocation is only efficient for sequential access files, as random access requires starting at the beginning of the list for each new location access.
- Allocating *clusters* of blocks reduces the space wasted by pointers, at the cost of internal fragmentation.
- Another big problem with linked allocation is reliability if a pointer is lost or damaged. Doubly linked lists provide some protection, at the cost of additional overhead and wasted space.

Disk files can be stored as linked lists, with the expense of the storage space consumed by each link. ( E.g. a block may be 512 bytes instead of 512. )
Linked allocation involves no external fragmentation, does not require pre-known file sizes, and allows files to grow dynamically at any time.
Unfortunately linked allocation is only efficient for sequential access files, as random access requires starting at the beginning of the list for each new location access.
Allocating clusters of blocks reduces the space wasted by pointers, at the cost of internal fragmentation.
Another big problem with linked allocation is reliability if a pointer is lost or damaged. Doubly linked lists provide some protection, at the cost of additional overhead and wasted space.

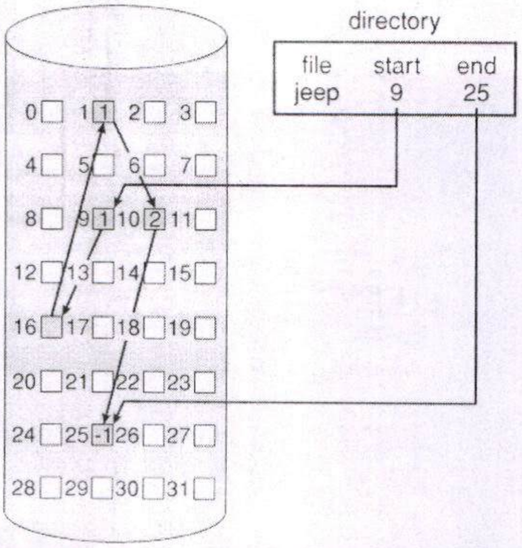


**Figure 12.6 - Linked allocation of disk space.**

- The ***File Allocation Table, FAT,*** used by DOS is a variation of linked allocation, where all the links are stored in a separate table at the beginning of the disk. The benefit of this approach is that the FAT table can be cached in memory, greatly improving random access speeds.

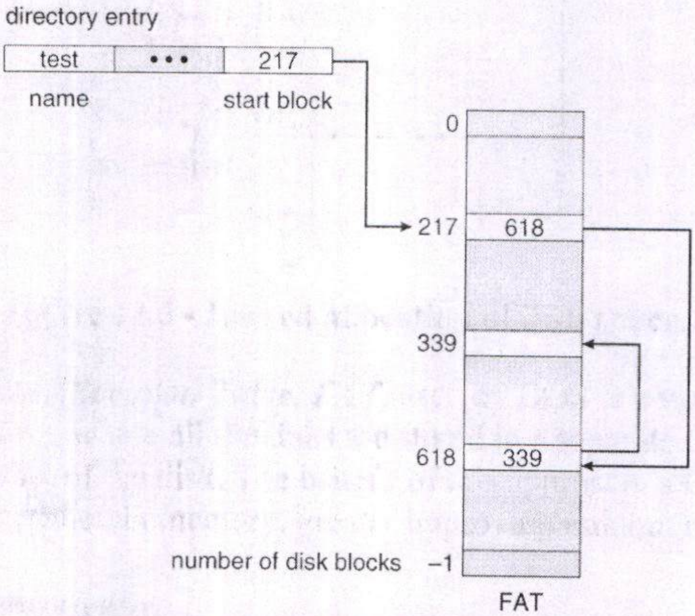


**Figure 12.7 File-allocation table.**

**12.4.3 Indexed Allocation**

- ***Indexed Allocation*** combines all of the indexes for accessing each file into a common block ( for that file ), as opposed to spreading them all over the disk or storing them in a FAT table.

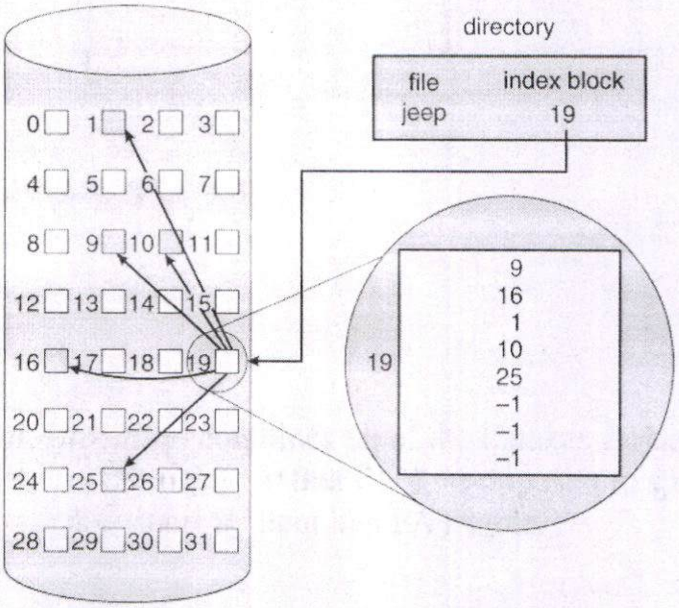
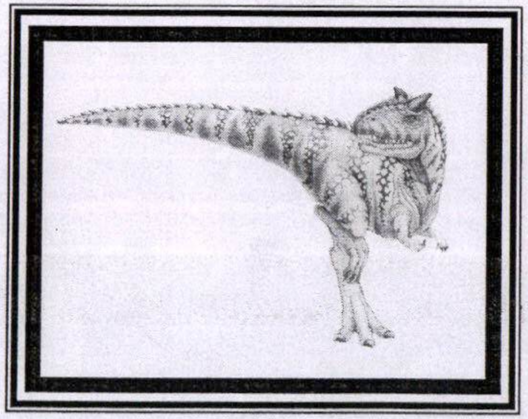


**Figure 12.8 - Indexed allocation of disk space.**

- Some disk space is wasted ( relative to linked lists or FAT tables ) because an entire index block must be allocated for each file, regardless of how many data blocks the file contains. This leads to questions of how big the index block should be, and how it should be implemented. There are several approaches:
    - **Linked Scheme** - An index block is one disk block, which can be read and written in a single disk operation. The first index block contains some header information, the first N block addresses, and if necessary a pointer to additional linked index blocks.
    - **Multi-Level Index** - The first index block contains a set of pointers to secondary index blocks, which in turn contain pointers to the actual data blocks.
    - **Combined Scheme** - This is the scheme used in UNIX inodes, in which the first 12 or so data block pointers are stored directly in the inode, and then singly, doubly, and triply indirect pointers provide access to more data blocks as needed. ( See below. ) The advantage of this scheme is that for small files ( which many are ), the data blocks are readily accessible ( up to 48K with 4K block sizes ); files up to about 4144K ( using 4K blocks ) are accessible with only a single indirect block ( which can be cached ), and huge files are still accessible using a relatively small number of

# Chapter 1: Introduction



# Chapter 1: Introduction

- What Operating Systems Do
- Computer-System Organization
- Computer-System Architecture
- Operating-System Structure
- Operating-System Operations
- Process Management
- Memory Management
- Storage Management
- Protection and Security
- Kernel Data Structures
- Computing Environments
- Open-Source Operating Systems

# Objectives

- To describe the basic organization of computer systems
- To provide a grand tour of the major components of operating systems
- To give an overview of the many types of computing environments
- To explore several open-source operating systems

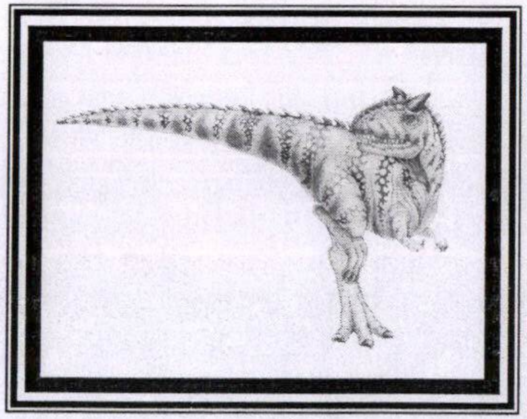
# What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware
- Operating system goals:
  - Execute user programs and make solving user problems easier
  - Make the computer system convenient to use
  - Use the computer hardware in an efficient manner

# Computer System Structure

- Computer system can be divided into four components:
  - Hardware – provides basic computing resources
    - CPU, memory, I/O devices
  - Operating system
    - Controls and coordinates use of hardware among various applications and users
  - Application programs – define the ways in which the system resources are used to solve the computing problems of the users
    - Word processors, compilers, web browsers, database systems, video games
  - Users
    - People, machines, other computers

# Chapter 2: Operating-System Structures



# Chapter 2: Operating-System Structures

- Operating System Services
- User Operating System Interface
- System Calls
- Types of System Calls
- System Programs
- Operating System Design and Implementation
- Operating System Structure
- Operating System Debugging
- Operating System Generation
- System Boot

# Objectives

- To describe the services an operating system provides to users, processes, and other systems
- To discuss the various ways of structuring an operating system
- To explain how operating systems are installed and customized and how they boot

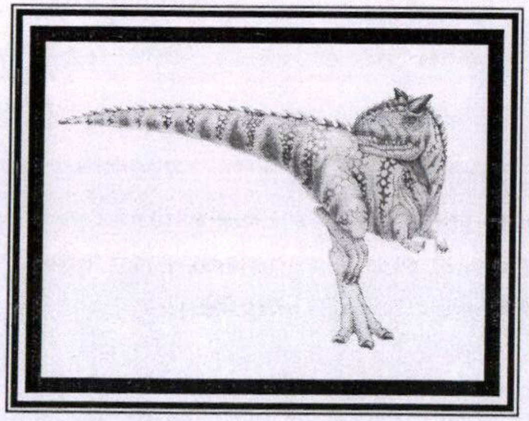
# Operating System Services

- Operating systems provide an environment for execution of programs and services to programs and users
- One set of operating-system services provides functions that are helpful to the user:
  - **User interface** - Almost all operating systems have a user interface (**UI**).
    - Varies between **Command-Line (CLI)**, **Graphics User Interface (GUI)**, **Batch**
  - **Program execution** - The system must be able to load a program into memory and to run that program, end execution, either normally or abnormally (indicating error)
  - **I/O operations** - A running program may require I/O, which may involve a file or an I/O device

# Operating System Services (Cont.)

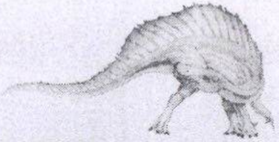
- One set of operating-system services provides functions that are helpful to the user (Cont.):
  - **File-system manipulation** - The file system is of particular interest. Programs need to read and write files and directories, create and delete them, search them, list file Information, permission management.
  - **Communications** – Processes may exchange information, on the same computer or between computers over a network
    - Communications may be via shared memory or through message passing (packets moved by the OS)
  - **Error detection** – OS needs to be constantly aware of possible errors
    - May occur in the CPU and memory hardware, in I/O devices, in user program
    - For each type of error, OS should take the appropriate action to ensure correct and consistent computing
    - Debugging facilities can greatly enhance the user' s and programmer' s abilities to efficiently use the system

# Chapter 3:  Processes



# Chapter 3:  Processes

- Process Concept
- Process Scheduling
- Operations on Processes
- Interprocess Communication
- Examples of IPC Systems
- Communication in Client-Server Systems

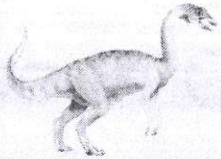


# Objectives

- To introduce the notion of a process -- a program in execution, which forms the basis of all computation
- To describe the various features of processes, including scheduling, creation and termination, and communication
- To explore interprocess communication using shared memory and message passing
- To describe communication in client-server systems

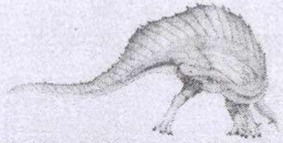
# Process Concept

- An operating system executes a variety of programs:
  - Batch system – **jobs**
  - Time-shared systems – **user programs** or **tasks**
- Textbook uses the terms ***job*** and ***process*** almost interchangeably
- **Process** – a program in execution; process execution must progress in sequential fashion
- Multiple parts
  - The program code, also called **text section**
  - Current activity including **program counter**, processor registers
  - **Stack** containing temporary data
    - Function parameters, return addresses, local variables
  - **Data section** containing global variables
  - **Heap** containing memory dynamically allocated during run time

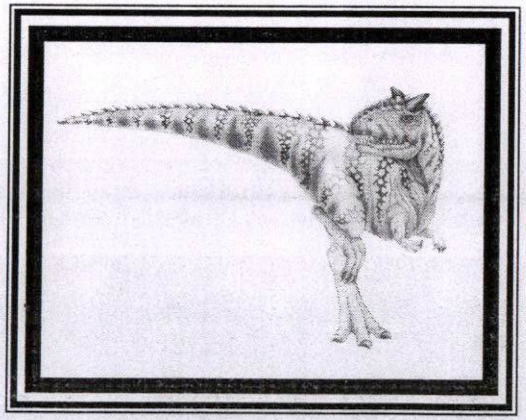


# Process Concept (Cont.)

- Program is ***passive*** entity stored on disk (**executable file**), process is ***active***
  - Program becomes process when executable file loaded into memory
- Execution of program started via GUI mouse clicks, command line entry of its name, etc
- One program can be several processes
  - Consider multiple users executing the same program



# Chapter 4:  Threads



# Chapter 4: Threads

- Overview
- Multicore Programming
- Multithreading Models
- Thread Libraries
- Implicit Threading
- Threading Issues
- Operating System Examples

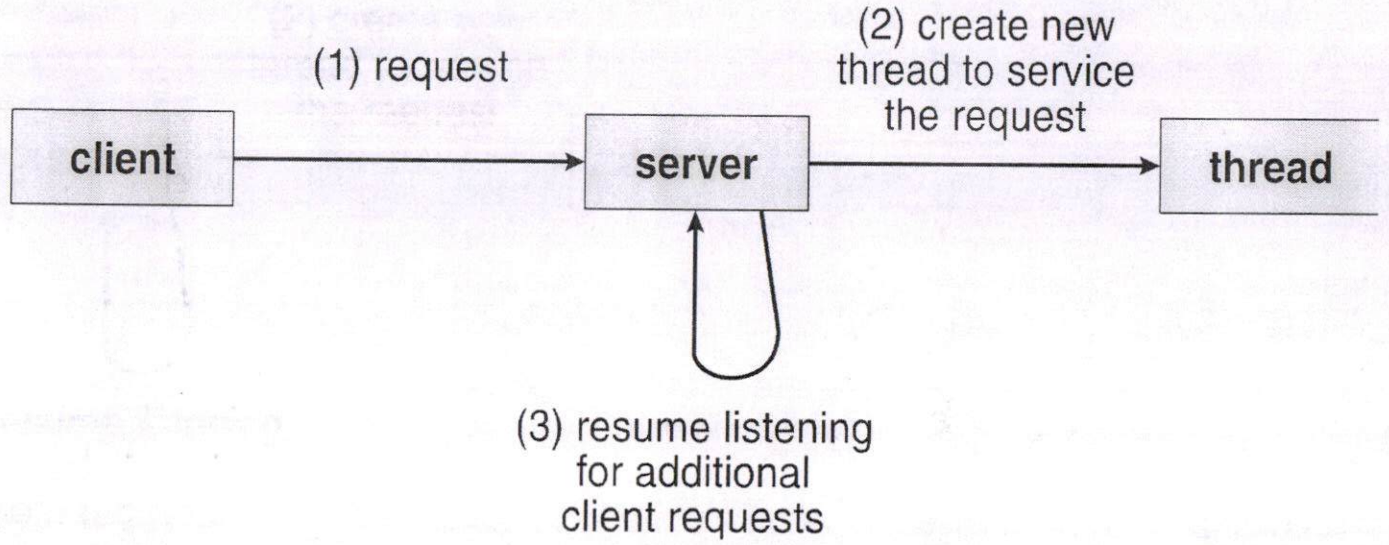
# Objectives

- To introduce the notion of a thread—a fundamental unit of CPU utilization that forms the basis of multithreaded computer systems
- To discuss the APIs for the Pthreads, Windows, and Java thread libraries
- To explore several strategies that provide implicit threading
- To examine issues related to multithreaded programming
- To cover operating system support for threads in Windows and Linux

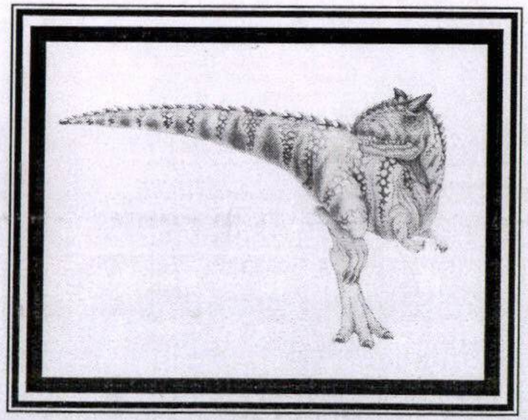
# Motivation

- Most modern applications are multithreaded
- Threads run within application
- Multiple tasks with the application can be implemented by separate threads
  - Update display
  - Fetch data
  - Spell checking
  - Answer a network request
- Process creation is heavy-weight while thread creation is light-weight
- Can simplify code, increase efficiency
- Kernels are generally multithreaded

# Multithreaded Server Architecture


(1) request
(2) create new thread to service the request
client
server
thread
(3) resume listening for additional client requests

# Chapter 5: Process Synchronization



# Chapter 5: Process Synchronization

- Background
- The Critical-Section Problem
- Peterson' s Solution
- Synchronization Hardware
- Mutex Locks
- Semaphores
- Classic Problems of Synchronization
- Monitors
- Synchronization Examples
- Alternative Approaches

# Objectives

- To present the concept of process synchronization.
- To introduce the critical-section problem, whose solutions can be used to ensure the consistency of shared data
- To present both software and hardware solutions of the critical-section problem
- To examine several classical process-synchronization problems
- To explore several tools that are used to solve process synchronization problems

# Background

- Processes can execute concurrently
  - May be interrupted at any time, partially completing execution
- Concurrent access to shared data may result in data inconsistency
- Maintaining data consistency requires mechanisms to ensure the orderly execution of cooperating processes
- Illustration of the problem:
  Suppose that we wanted to provide a solution to the consumer-producer problem that fills ***all*** the buffers. We can do so by having an integer **counter** that keeps track of the number of full buffers. Initially, **counter** is set to 0. It is incremented by the producer after it produces a new buffer and is decremented by the consumer after it consumes a buffer.

# Producer

```
while (true) {
     /* produce an item in next produced */

     while (counter == BUFFER_SIZE) ;
          /* do nothing */
     buffer[in] = next_produced;
     in = (in + 1) % BUFFER_SIZE;
     counter++;
}
```

# Consumer

```
while (true) {
     while (counter == 0)
          ; /* do nothing */
     next_consumed = buffer[out];
     out = (out + 1) % BUFFER_SIZE;
      counter--;
     /* consume the item in next consumed */
}
```

# Race Condition

- **`counter++`** could be implemented as

```
register1 = counter
register1 = register1 + 1
counter = register1
```

- **`counter--`** could be implemented as

```
register2 = counter
register2 = register2 - 1
counter = register2
```

- Consider this execution interleaving with "count = 5" initially:

S0: producer execute `register1 = counter` {register1 = 5}
S1: producer execute `register1 = register1 + 1` {register1 = 6}
S2: consumer execute `register2 = counter` {register2 = 5}
S3: consumer execute `register2 = register2 - 1` {register2 = 4}
S4: producer execute `counter = register1` {counter = 6 }
S5: consumer execute `counter = register2` {counter = 4}

# Critical Section Problem

- Consider system of ***n*** processes $\{p_0, p_1, \ldots p_{n-1}\}$
- Each process has **critical section** segment of code
  - Process may be changing common variables, updating table, writing file, etc
  - When one process in critical section, no other may be in its critical section
- ***Critical section problem*** is to design protocol to solve this
- Each process must ask permission to enter critical section in **entry section**, may follow critical section with **exit section**, then **remainder section**

# Critical Section

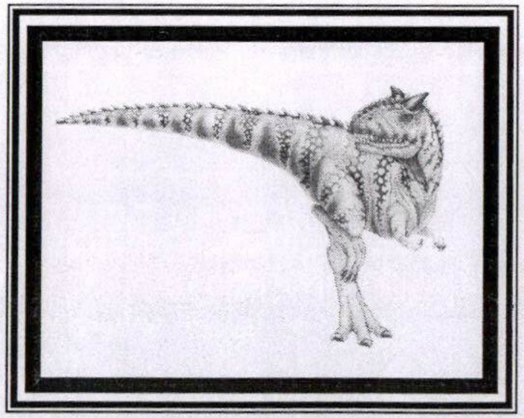
- General structure of process $P_i$

```
do {

     entry section

          critical section

     exit section

          remainder section

} while (true);
```

# Algorithm for Process $P_i$

```
do {

        while (turn == j);
                critical section
        turn = j;
                remainder section
 } while (true);
```

# Chapter 6: CPU Scheduling



# Chapter 6: CPU Scheduling

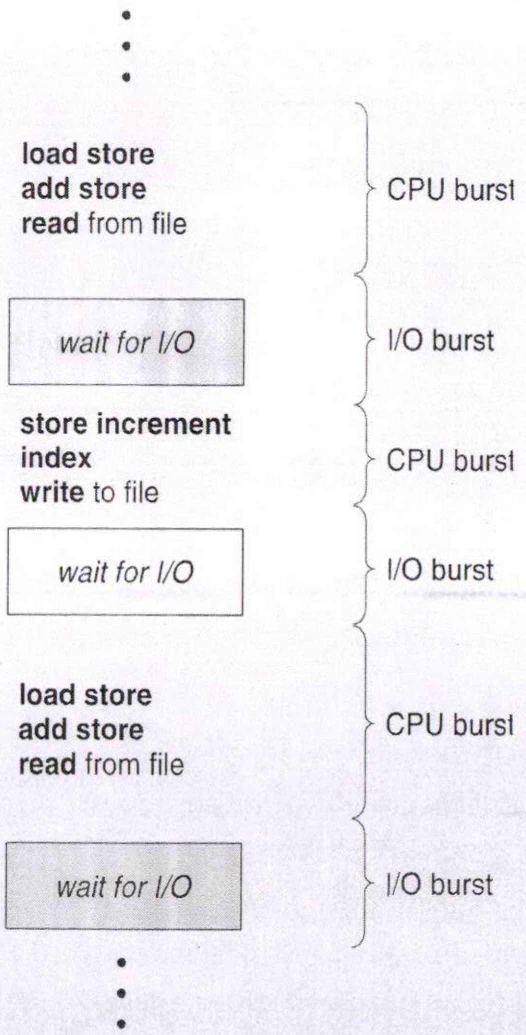
- Basic Concepts
- Scheduling Criteria
- Scheduling Algorithms
- Thread Scheduling
- Multiple-Processor Scheduling
- Real-Time CPU Scheduling
- Operating Systems Examples
- Algorithm Evaluation

# Objectives

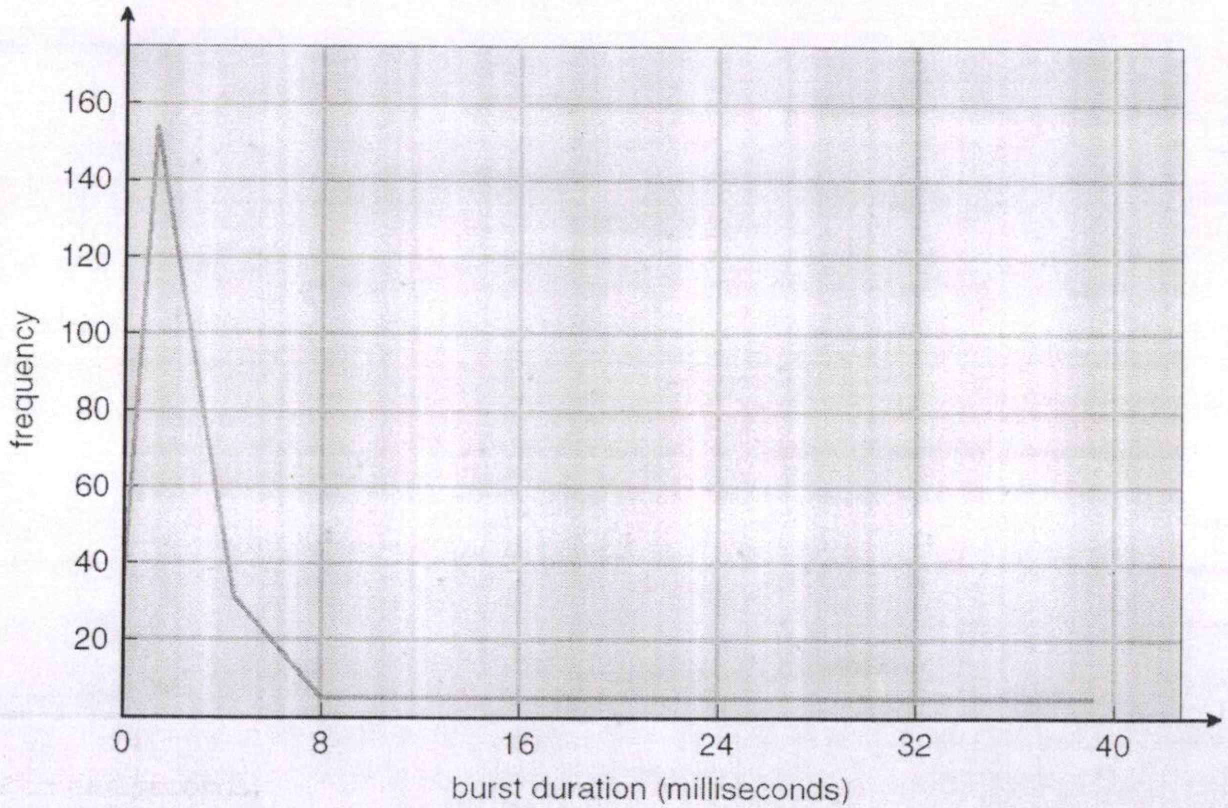
- To introduce CPU scheduling, which is the basis for multiprogrammed operating systems
- To describe various CPU-scheduling algorithms
- To discuss evaluation criteria for selecting a CPU-scheduling algorithm for a particular system
- To examine the scheduling algorithms of several operating systems

# Basic Concepts

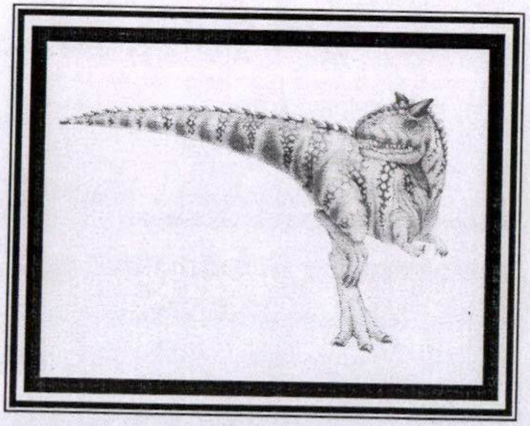
- Maximum CPU utilization obtained with multiprogramming
- CPU–I/O Burst Cycle – Process execution consists of a **cycle** of CPU execution and I/O wait
- **CPU burst** followed by **I/O burst**
- CPU burst distribution is of main concern



# Histogram of CPU-burst Times



# Chapter 7:  Deadlocks

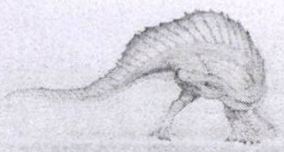


# Chapter 7:  Deadlocks

- System Model
- Deadlock Characterization
- Methods for Handling Deadlocks
- Deadlock Prevention
- Deadlock Avoidance
- Deadlock Detection
- Recovery from Deadlock

# Chapter Objectives

- To develop a description of deadlocks, which prevent sets of concurrent processes from completing their tasks
- To present a number of different methods for preventing or avoiding deadlocks in a computer system



# System Model

- System consists of resources
- Resource types $R_1, R_2, \ldots, R_m$

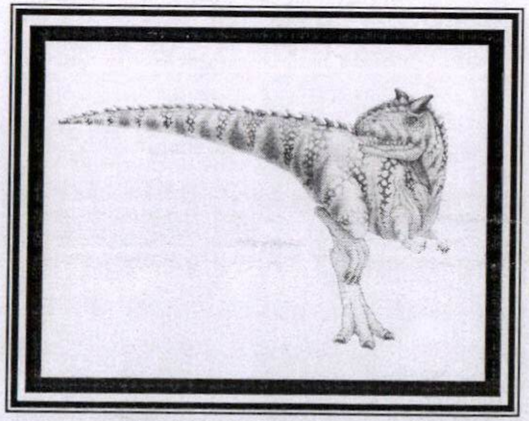
  *CPU cycles, memory space, I/O devices*
- Each resource type $R_i$ has $W_i$ instances.
- Each process utilizes a resource as follows:
  - **request**
  - **use**
  - **release**

# Deadlock Characterization

Deadlock can arise if four conditions hold simultaneously.

- **Mutual exclusion:** only one process at a time can use a resource
- **Hold and wait:** a process holding at least one resource is waiting to acquire additional resources held by other processes
- **No preemption:** a resource can be released only voluntarily by the process holding it, after that process has completed its task
- **Circular wait:** there exists a set $\{P_0, P_1, \ldots, P_n\}$ of waiting processes such that $P_0$ is waiting for a resource that is held by $P_1$, $P_1$ is waiting for a resource that is held by $P_2$, …, $P_{n-1}$ is waiting for a resource that is held by $P_n$, and $P_n$ is waiting for a resource that is held by $P_0$.

# Chapter 8: Main Memory



# Chapter 8:  Memory Management

- Background
- Swapping
- Contiguous Memory Allocation
- Segmentation
- Paging
- Structure of the Page Table
- Example: The Intel 32 and 64-bit Architectures
- Example: ARM Architecture

# Objectives

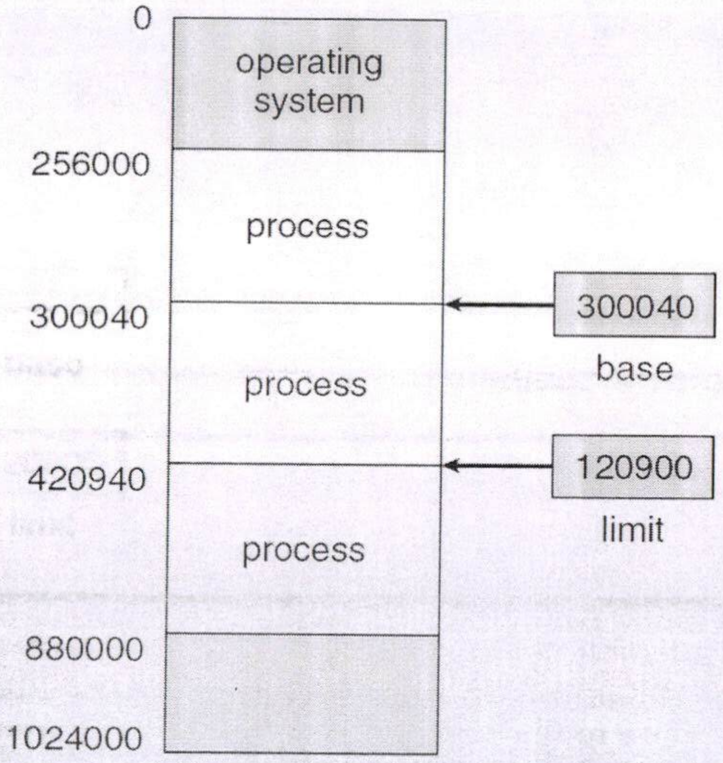
- To provide a detailed description of various ways of organizing memory hardware
- To discuss various memory-management techniques, including paging and segmentation
- To provide a detailed description of the Intel Pentium, which supports both pure segmentation and segmentation with paging

# Background

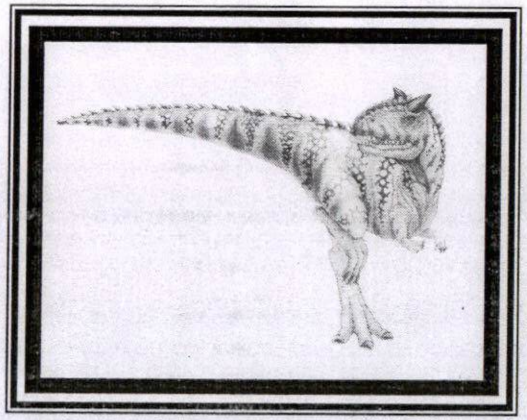
- Program must be brought (from disk) into memory and placed within a process for it to be run
- Main memory and registers are only storage CPU can access directly
- Memory unit only sees a stream of addresses + read requests, or address + data and write requests
- Register access in one CPU clock (or less)
- Main memory can take many cycles, causing a **stall**
- **Cache** sits between main memory and CPU registers
- Protection of memory required to ensure correct operation

# Base and Limit Registers

- A pair of **base** and **limit registers** define the logical address space
- CPU must check every memory access generated in user mode to be sure it is between base and limit for that user



# Chapter 9: Virtual Memory



# Chapter 9: Virtual Memory

- Background
- Demand Paging
- Copy-on-Write
- Page Replacement
- Allocation of Frames
- Thrashing
- Memory-Mapped Files
- Allocating Kernel Memory
- Other Considerations
- Operating-System Examples

# Chapter 9:  Virtual Memory

- Background
- Demand Paging
- Copy-on-Write
- Page Replacement
- Allocation of Frames
- Thrashing
- Memory-Mapped Files
- Allocating Kernel Memory
- Other Considerations
- Operating-System Examples

# Objectives

- To describe the benefits of a virtual memory system
- To explain the concepts of demand paging, page-replacement algorithms, and allocation of page frames
- To discuss the principle of the working-set model
- To examine the relationship between shared memory and memory-mapped files
- To explore how kernel memory is managed

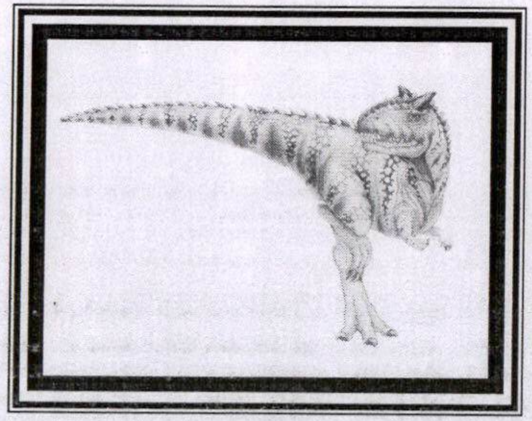
# Background

- Code needs to be in memory to execute, but entire program rarely used
  - Error code, unusual routines, large data structures
- Entire program code not needed at same time
- Consider ability to execute partially-loaded program
  - Program no longer constrained by limits of physical memory
  - Each program takes less memory while running -> more programs run at the same time
    - Increased CPU utilization and throughput with no increase in response time or turnaround time
  - Less I/O needed to load or swap programs into memory -> each user program runs faster

# Background (Cont.)

- **Virtual memory** – separation of user logical memory from physical memory
  - Only part of the program needs to be in memory for execution
  - Logical address space can therefore be much larger than physical address space
  - Allows address spaces to be shared by several processes
  - Allows for more efficient process creation
  - More programs running concurrently
  - Less I/O needed to load or swap processes

# Chapter 11: File-System Interface



# Chapter 11: File-System Interface

- File Concept
- Access Methods
- Disk and Directory Structure
- File-System Mounting
- File Sharing
- Protection

# Objectives

- To explain the function of file systems
- To describe the interfaces to file systems
- To discuss file-system design tradeoffs, including access methods, file sharing, file locking, and directory structures
- To explore file-system protection

# File Concept

- Contiguous logical address space
- Types:
  - Data
    - numeric
    - character
    - binary
  - Program
- Contents defined by file's creator
  - Many types
    - Consider **text file, source file, executable file**

# File Attributes

- **Name** – only information kept in human-readable form
- **Identifier** – unique tag (number) identifies file within file system
- **Type** – needed for systems that support different types
- **Location** – pointer to file location on device
- **Size** – current file size
- **Protection** – controls who can do reading, writing, executing
- **Time, date, and user identification** – data for protection, security, and usage monitoring
- Information about files are kept in the directory structure, which is maintained on the disk
- Many variations, including extended file attributes such as file checksum
- Information kept in the directory structure

H.T No: □□□□□□□□□□ **R18** **Course Code: A30516**

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

**(UGC AUTONOMOUS)**

**B.Tech V Semester Regular/Supplementary Examinations February -2022**

**Course Name: OPERATING SYSTEMS**

**(Common for CSE & IT)**

**Date: 16.02.2022 FN** **Time: 3 hours** **Max.Marks: 70**

**(Note: Assume suitable data if necessary)**

## PART-A

**Answer all TEN questions (Compulsory)**
**Each question carries TWO marks.** **10x2=20M**

1. What is a system call? Name some system calls. 2 M
2. What is a Time-shared operating system? 2 M
3. Define Inter-Process Communication. 2 M
4. Define the terms process and thread. 2 M
5. What is a Deadlock? Give an example. 2 M
6. Draw and explain process control block. 2 M
7. List out the differences between paging and segmentation 2 M
8. Explain Swapping. 2 M
9. List out file attributes. 2 M
10. List any four common file types along with their extensions and describe each file type. 2 M

## PART-B

**Answer the following.Each question carries TEN Marks.** **5x10=50M**

11.A). i) Explain the components of an operating system. 5 M
ii) Explain Batch and Real Time Operating Systems. 5 M

**OR**

11. B). Explain all operating system structures in detail. 10 M

12. A). Determine the average waiting time and average turnaround time for FCFS, SJF, non-preemptive priority and round robin scheduling algorithms for the given process, burst and priority given below. 10 M

| Process | Burst Time (milli sec) | Priority |
|---|---|---|
| P1 | 8 | 4 |
| P2 | 6 | 1 |
| P3 | 1 | 2 |
| P4 | 9 | 2 |
| P5 | 3 | 3 |

**OR**

12. B). i) Define Process and explain various states of a Process. 4 M
ii) Explain various scheduling criteria for CPU Scheduling. 6 M

*(P.T.O..)*

13. A). Consider the table given below for a system, determine the need matrix and the safety sequence, using Banker's algorithm. 10 M

Resource – 3 types<br>
A – (10 instances)<br>
B – (5 instances)<br>
C – (7 instances)

| Process | Allocation | | | Maximum | | | Available | | |
|---|---|---|---|---|---|---|---|---|---|
| | A | B | C | A | B | C | A | B | C |
| P0 | 0 | 1 | 0 | 7 | 5 | 3 | 3 | 3 | 2 |
| P1 | 2 | 0 | 0 | 3 | 2 | 2 | | | |
| P2 | 3 | 0 | 2 | 9 | 0 | 2 | | | |
| P3 | 2 | 1 | 1 | 2 | 2 | 2 | | | |
| P4 | 0 | 0 | 2 | 4 | 3 | 3 | | | |

**OR**

13. B). Explain all Deadlock Recovery methods in detail. 10 M

14. A). Explain in detail about demand paging, page replacement. 10 M

**OR**

14. B). Given page reference string: 1,2,3,2,1,5,2,1,6,2,5,6,3,1,3,6,1,2,4,3. Identify the number of page faults for LRU, FIFO and Optimal page replacement algorithm. Choose the best algorithm for the given reference string. 10 M

15. A). What are different directory structures? Explain about each with neat diagram. 10 M

**OR**

15. B). Explain free space management techniques in detail. 10 M

*****

H.T No: □□□□□□□□□□ **R18** **Course Code: A30516**

## CMR COLLEGE OF ENGINEERING & TECHNOLOGY
## (UGC AUTONOMOUS)

**B.Tech V Semester Regular & Supplementary Examinations March -2021**

**Course Name: OPERATING SYSTEMS**

## (Comupter Sceince & Engineering)

**Date: 09.03.2021 FN** **Time: 3 hours** **Max.Marks:70**

**(Note: Assume suitable data if necessary)**

### PART-A

**Answer all TEN questions (Compulsory)**

**Each question carries TWO marks.** **10x2=20M**

1. Define Real-Time Operating System. 2 M
2. Compare Multiprocessing and Distributed Systems. 2 M
3. List and define the types of schedulers. 2 M
4. Differentiate semaphore and counting semaphore. 2 M
5. Define race condition. 2 M
6. List the necessary conditions for deadlock. 2 M
7. What is the purpose of the paging and page table? 2 M
8. Consider the page reference string 1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1,2,3,6. Find the number of page faults using 3 page frames by applying LRU Page Replacement algorithm. 2 M
9. List the various file attributes. 2 M
10. Compare internal and external fragmentations. 2 M

### PART-B

**Answer the following. Each question carries TEN Marks.** **5x10=50M**

11. A). What is a System Call? Explain different types of System Calls. 10 M

**OR**

11. B). Explain the following Operating Systems structures in detail. 10 M
i) Simple Structure
ii) Layered Approach

12. A). i) Demonstrate process control block with a neat sketch. 10 M
ii) Explain multi-thread models in detail.

**OR**

12. B). Consider 3 processes P1, P2 and P3, which require 5, 7, and 4- burst time in milli seconds. 10 M

Draw the Gant chart, process completion sequence and average waiting time for
i) Round robin scheduling with CPU quantum of 2-time units.
ii) FCFS.

13. A). i) List and explain the various deadlock handling methods. 10 M

ii) Consider the table given below for a system, find the need matrix and the safety sequence.

Resource – 3 types
A – (10 instances)
B – (5 instances)
C – (7 instances)

*(P.T.O..)*

| Process | Allocation | Maximum | Available |
|---|---|---|---|
| | A B C | A B C | A B C |
| p0 | 0 1 0 | 7 5 3 | 3 3 2 |
| pl | 2 0 0 | 3 2 2 | |
| p2 | 3 0 2 | 9 0 2 | |
| p3 | 2 1 1 | 2 2 2 | |
| p4 | 0 0 2 | 4 3 3 | |

**OR**

13. B). i) How to recover the system from a deadlock? Discuss. 10 M

ii) Illustrate the synchronization hardware for Critical section Problem.

14. A). i) Illustrate with example how paging is implemented for non-contiguous memory allocation. 10 M

ii) Describe the Swap Space Management in operating systems.

**OR**

14. B). i) Consider the following segment table: 10 M

| Segment | Base | Length |
|---|---|---|
| 0 | 219 | 600 |
| 1 | 2300 | 14 |
| 2 | 90 | 100 |
| 3 | 1327 | 580 |
| 4 | 1952 | 96 |

Identify the physical addresses for the following logical addresses?

a) 0,430

b) 1,10

c) 2,50

d) 3,400

e) 4,11

ii) What is Page Replacement? Discuss the FIFO, Optimal page replacement algorithms with example.

15. A). i) Explain about linked allocation method of a file. 10 M

ii) Explain different Directory Structures.

**OR**

15. B). i) What is File? What are various file access methods. 10 M

ii) Explain the File System Structure.

*****

H.T No: ☐☐☐☐☐☐☐☐☐☐ **R18** Course Code: A30516

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(UGC AUTONOMOUS)

**B.Tech V Semester Supplementary Examinations June/July-2022.**

**Course Name: Opearting Systems**

**(Common for CSE & IT)**

**Date: 09.07.2022 AN** **Time: 3 hours** **Max.Marks: 70**

**(Note: Assume suitable data if necessary)**

## PART-A

**Answer all TEN questions (Compulsory)**
**Each question carries TWO marks.** **10x2=20M**

1. Define distributed system. 2 M
2. Write different categories of system calls. 2 M
3. Differentiate process and thread. 2 M
4. What is scheduling? What criteria affect the scheduler's performance? 2 M
5. Define 'Monitor'. What does it consist of? 2 M
6. What necessary conditions can lead to a deadlock situation in a system? 2 M
7. What is demand paging? 2 M
8. Define thrashing. 2 M
9. List out the major attributes and operations of a file. 2 M
10. Write any four common file types. 2 M

## PART-B

**Answer the following.Each question carries TEN Marks.** **5x10=50M**

11.A). Define the essential properties of the following types of operating systems: 10M
i) Batch ii) Time sharing iii) Real time iv) Parallel v) Multi-programmed

OR

11. B). List out the various functions of operating system. 10M

12. A). Compare preemptive and non-pre-emptive scheduling methods and explain in detail the SJF and Round Robin scheduling policies with an example. 10M

OR

12. B). What are the advantages of inter-process communication? How communication takes place in a shared-memory environment? Explain. 10M

13. A). Discuss in detail about Binary Semaphore and Counting semaphore? 10M

OR

13. B). Explain the Banker's algorithm for deadlock avoidance with an example. 10M

14. A). i) What is the purpose of paging the page tables? 5M
ii) Discuss the hardware support required to support demand paging. 5M

OR

14. B). Explain Optimal and FIFO Page Replacement algorithms with an example. 10M

15. A). Explain the three allocation methods in file system implementation. Illustrate with proper diagram. 10M

OR

15. B). Write and explain the following system calls. 10M
i) open ii) create iii) read iv) stat v) close

*****