ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY & SCIENCES
Autonomous status accorded by UGC and Andhra University
Approved by AICTE, Permanently Affiliated to Andhra University
Accredited by NBA (IT,CSE,EEE,ECE, and Mech) & accredited by NAAC with "A" Grade
Sangivalasa - 531162, Bheemunipatnam (Mandal), Visakhapatnam (Dist.)
Phone: 08933 - 225083, 225084, 226131, Fax: 08933-226395
Email: principal@anits.edu.in
COLLEGE CODE - ANIL

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

A LABORATORY MANUAL FOR OPERATING SYSTEMS

Faculty Incharge:
Mrs K. Amaravathi, Assistant Professor
Dr Sangeetha, Professor
Ms Spandana Valli, Assistant Professor

VISION:

Our vision is to emerge as a world class Computer Science and Engineering department through excellent teaching and a strong research environment that responds swiftly to the challenges of changing computer science technology and addresses technological needs of the stakeholders.

MISSION:

To enable our students to master the fundamental principles of computing and to develop in them the skills needed to solve practical problems using contemporary computer-based technologies and practices to cultivate a community of professionals who will serve the public as resources on state-of- the-art computing science and information technology.

Course outcomes:

CO1: Execute the Unix Shell programming on the given system configuration.

CO2: Learn the various services provided by the system calls.

CO3: Simulate the process scheduling, process synchronization, deadlock avoidance and detection algorithms.

CO4: Simulate memory management techniques and file handling.

PROGRAM OUTCOMES (POs):

Graduate Attribute1:	Engineering Knowledge
PO-A	An ability to apply the knowledge of basic engineering sciences, humanities, core engineering and computing concept in modeling and designing computer based systems.
Graduate Attribute2:	Problem Analysis
PO-B	An ability to identify, analyze the problems in different domains and define the requirements appropriate to the solution.
Graduate Attribute3:	Design/Development of Solution
PO-C	An ability to design, implement & test a computer based system, component or process that meet functional constraints such as public health and safety, cultural, societal and environmental considerations.
Graduate Attribute4:	Conduct Investigations of Complex Problems
PO-D	An ability to apply computing knowledge to conduct experiments and solve complex problems, to analyze and interpret the results obtained within specified timeframe and financial constraints consistently.
Graduate Attribute5:	Modern Tool Usage
PO-E	An ability to apply or create modern techniques and tools to solve engineering problems that demonstrate cognition of limitations involved in design choices.
Graduate Attribute6:	The Engineer and Society
PO-F	An ability to apply contextual reason and assess the local and global impact of professional engineering practices on individuals, organizations and society.
Graduate Attribute7:	Environment and Sustainability
PO-G	An ability to assess the impact of engineering practices on societal and environmental sustainability.
Graduate Attribute8:	Ethics
PO-H	Ability to apply professional ethical practices and transform into good responsible citizens with social concern.
Graduate Attribute9:	Individual and Team Work
PO-I	Acquire capacity to understand and solve problems pertaining to various fields of engineering and be able to function effectively as an individual and as a member or leader in a team.

Graduate Attribute10:	Communication
PO-J	An ability to communicate effectively with range of audiences in both oral and written forms through technical papers, seminars, presentations, assignments, project reports etc.
Graduate Attribute11:	Project Management and Finance
PO-K	An ability to apply the knowledge of engineering, management and financial principles to develop and critically assess projects and their outcomes in multidisciplinary areas.
Graduate Attribute12:	Life-long Learning
PO-L	An ability to recognize the need and prepare oneself for lifelong self learning to be abreast with rapidly changing technology.

PROGRAM SPECIFIC OUTCOMES (PSOs):

1	Programming and software Development skills: Ability to acquire programming efficiency to analyze, design and develop optimal solutions, apply standard practices in software project development to deliver quality software product.
2	Computer Science Specific Skills: Ability to formulate, simulate and use knowledge in various domains like data engineering, image processing and information and network security, artificial intelligence etc., and provide solutions to new ideas and innovations.

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES

A Laboratory Manual

For

OPERATING SYSTEMS (CSE 228)

Semester – II

Prepared by

1. K.Amaravathi, Assistant Professor
2. Dr Sangeetha, Professor
3. Ms Spandana Valli, Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

<table>
<tr><th>SI. No</th><th>List of Experiments</th><th>CO</th></tr>
<tr><td>1</td><td>Implement basic shell commands.</td><td>1</td></tr>
<tr><td>2</td><td>Shell programming: Simple logic programs.
i) Write a menu driven script using the select statement to print calories for food items such as pizza, burger, Salad, Pasta etc.
ii) Write a shell script that, given a filename as the argument, will count vowels, blank spaces, characters, number of lines and symbols.</td><td>1</td></tr>
<tr><td>3</td><td>i) Analyze the below situation and develop a program for creating processes as required. Print the PID’s of each process in a convenient way to understand.
ii) Write a program to create two processes P1 and P2. P1 takes a string and passes it to P2. P2 concatenates the received string with another string without using string function and sends it back to P1 for printing.</td><td>2</td></tr>
<tr><td>4</td><td>CPU Scheduling Algorithms
i) A washing machine which requires the process to be executed sequentially. Consider the processes P1, P2, P3, P4 whose arrival times are 1, 5, 9, 10 and burst times are 4, 3, 5, 2 respectively. Implement an appropriate algorithm. Find the CPU idle time, so that the water can be supplied during that period of time.
ii) Implement shortest job first for the following data: Consider the following set of processes, CPU burst time, Arrival time. Calculate the average waiting time, average response time and average turnaround time.
<table><tr><th>Process</th><th>Burst Time</th><th>Arrival Time</th></tr><tr><td>P1</td><td>3</td><td>0</td></tr><tr><td>P2</td><td>6</td><td>2</td></tr><tr><td>P3</td><td>4</td><td>4</td></tr><tr><td>P4</td><td>5</td><td>6</td></tr><tr><td>P5</td><td>2</td><td>8</td></tr></table>
iii) Implement Round Robin for the following data Consider the following set of processes and length of the CPU burst time given in milliseconds.
<table><tr><th>Process</th><th>Burst Time</th></tr><tr><td>P1</td><td>10</td></tr><tr><td>P2</td><td>1</td></tr><tr><td>P3</td><td>2</td></tr><tr><td>P4</td><td>1</td></tr><tr><td>P5</td><td>5</td></tr></table>
The processes are assumed to have arrived in the order P1, P2, P3, P4, P5 all at time 0 and time quantum in RR=1.Calculate the average waiting time, response time and turnaround time.</td><td>3</td></tr>
<tr><td>5</td><td>Develop a program to provide synchronization among the 5 philosophers in Dining Philosophers problem using semaphore.</td><td>3</td></tr>
<tr><td>6</td><td>Develop a program to provide synchronization among the producer and consumer processes in producer – consumer problem using a monitor.</td><td>3</td></tr>
<tr><td>7</td><td>Consider the following data:</td><td>3</td></tr>
</table>

	<table><tr><th>Process</th><th>Allocation A B C D</th><th>Max A B C D</th><th>Available A B C D</th></tr><tr><td>P1</td><td>0 0 1 2</td><td>0 0 1 2</td><td>2 1 0 0</td></tr><tr><td>P2</td><td>2 0 0 0</td><td>2 7 5 0</td><td></td></tr><tr><td>P3</td><td>0 0 3 4</td><td>6 6 5 6</td><td></td></tr><tr><td>P4</td><td>2 3 5 4</td><td>4 3 5 6</td><td></td></tr><tr><td>P5</td><td>0 3 3 2</td><td>0 6 5 2</td><td></td></tr></table> i) Calculate the need matrix ii) Is this system currently in a safe or unsafe state? iii) Is the system currently deadlock or not. iv) Which process, if any, or may become deadlocked?	
8	Consider the following scenario: A process has been allocated 3 page frames. Assume that none of the pages of the process are available in the memory initially. The process makes the following sequence of page references (reference string): 1, 2, 1, 3, 7, 4, 5, 6, 3, 1, 2, 4, 6, 3, 1. Find out a page replacement policy which gives the least number of page faults.	4
9	Simulate the Virtual Memory concept.	4
10	Implement the first fit and best fit algorithm in memory management.	4
11	Simulate the Contiguous file allocation method.	4
12	Implement a bit map for the following scenario. For a memory of size 32 blocks ,the allocated blocks are 2,3,4,5,8,9,10,11,12 and display the bitmap pattern.	4

LIST OF INDUSTRY RELEVANT SKILLS:

- Network Administrator
- Operating System Application Designer
- System Engineer

GUIDELINES TO TEACHERS

- Faculty must verify the observations and records before assigning the system.
- Faculty must verify Students Id cards before entering into Laboratory
- Faculty must take the attendance at the starting and ending of the lab time period.

SESSIONAL MARKS : 50 MARKS

1) Daily Evaluation (Includes Record, Observation & regular performance) – 25 marks
2) Attendance – 5 marks
3) Internal Exam – 20 marks

DAILY EVALUATION (25 MARKS)

1. Every Student must execute a minimum set of sample programs to secure 60% of marks in Daily Evaluation i.e. 18 Marks and to appear in external examination.
2. In addition to that if a student finishes the minimum set and 5 programs from an additional set of programs would secure 80% of marks in Daily Evaluation i.e. 24 Marks.
3. If a student finishes all the programs in both the set s will secure 100% of marks in Daily Evaluation

INTERNAL EXAM (20 MARKS)

- Every student is given 4 questions in the internal exam out of which the difficulty level of 2 questions is easy / medium and 2 questions of difficulty level is high
- Each easy / medium level question carries 20% of marks and difficulty level question carries 30% of marks

EXTERNAL EXAM (50 MARKS)

- Viva voce – 10 marks
- Write up + Execution – 40 marks

WRITE UP + EXECUTION (40 MARKS)

- Every student is given 4 questions in the external exam out of which the difficulty level of 2 questions is easy / medium and 2 questions of difficulty level is high
- Each easy / medium level question carries 30% of marks and difficulty level question carries 20% of marks.

INSTRUCTIONS TO STUDENTS:

- Students should use computer related components smoothly
- Students should not carry other items into the lab.
- Students must wear the dress code and ID cards.
- Every student is given 4 questions in the external exam out of which the difficulty level of 2 questions is easy / medium and 2 questions of difficulty level is high
- Each easy / medium level question carries 30% of marks and difficulty level question carries 20% of marks.

GUIDELINES TO LAB PROGRAMMERS:

- Lab Programmers must verify All the Systems whether they are working properly or not.
- Lab Programmers must verify All the other equipment (devices like ACs).

LAB RUBRICS

Name of the Programme: B.Tech **Session Duration: 3 Hours**

Semester:II **Course Code and Title :OPERATING SYSTEMS (CSE 228)**

Key Performance Criteria(KPC) (25 pts)	4-Very Good	3-Good	2-Fair	1-Need to improve
Problem Statement (2)	Detail understanding of the problem (2)	Understanding of the problem (2)	Basic understanding of the problem (1)	Partial understanding of the problem (1)
Experimental Procedure/ algorithm/ flow chart/ analysis (4)	The procedure is explained and well designed the problem with appropriate analysis (4)	The procedure is explained and designed the problem with analysis (3)	Missing some experimental procedure with partial analysis (2)	Missing major experimental details and analysis (1)
Implementation (4)	Implement Optimal solution with appropriate results for all the inputs(4)	Implement solution with correct results for most of the inputs(3)	Implement solution with the correct answers for some inputs and results wrong answers for some cases(2)	Implement Solution does not produce the appropriate results for the given inputs(1)
Test Case verification (3)	Produces correct output for all possible test cases(3)	Produces correct output for most of the test cases (2)	Produces correct output for some of the test cases (2)	Produces Wrong output for most of the test cases (1)
Viva voce / oral presentation(5)	In depth knowledge on the concept and answered all the questions(5)	Good knowledge on the concept and answered all the questions(4)	Basic knowledge on the concept and answered some of the questions(3)	With basic knowledge on the concept and answered few questions(2)
Presentation of record / documentation(4)	Presented the content effectively and Submitted on time (4)	Presented the content and Submitted on time (3)	Presented the in-complete content and Submitted . (2)	Presented the wrong content and submitted in delay.(1)

Code of conduct (courtesy, safety, behavioral aspects, ethics etc.)(3)	While conducting the procedure, the student is in proper dress code, always respectful of others and leaves the area clean.(3)	While conducting the procedure, the student is in proper dress code, many times respectful of others and leaves the area clean only after being reminded.(2)	While conducting the procedure, the student is in partial dress code, sometimes respectful of others and leaves the area clean only after being reminded.(2)	While conducting the procedure, the student is not in proper dress code , not respectful of others and leaves the area messy even after being reminded.(1)

PRACTICAL 1: Implement basic shell commands.

1. Practical significance :

1. **systems administration:** Like for example, how to use simple loops to process hundreds or thousands of files in one go - as opposed to clicking/opening on each file.
   An example : When we have a whole directory with pictures in jpeg, png, gif, etc. and we want to resize them all at once. Taking 5 to 10 seconds to type a loop with imagemagick's "mogrify" command (and then let it process in the background for a few seconds/minutes) is a LOT faster than opening each picture in an image editor and scaling it manually, which could take hours or even days. This is just one very simple example, and knowing just a few shell scripting tricks can help to solve MANY such problems.

2. Pretty much any kind of computer will have its system scripts written in its native shell code. So, if we want to understand and configure what it does, we will need to learn the shell script.

3. The other useful thing is that it will teach us how to use our command line properly. We can't do everything with tools(e.g., cp -r folder1 folder2). So, shell is the simplest and most appropriate tool for that kind of job.

4. Automating a simple task, like backups.

5. Shell programming lets us slice and dice tasks that would be painfully slow if we were to use the GUI.

2. Relevant Program Outcomes :PO 1, PO 2, PO 3, PO 4

3. Competency and practical skills :

The practical is expected to develop the following skills

1. Ability to work on Linux operating systems.

2. Familiarity with the Linux operating system and its commands/utilities at a user level.

3. Ability to edit files, use basic utilities and commands, navigate through the file system.

4. Prerequisites :

4. There are no prerequisites for studying Linux, except perhaps the most basic of computer skills, such as knowing how to turn a computer on and knowing how to use a mouse and keyboard.

5. It is also important to have a computer, a copy of Linux itself and a strong desire to learn.

5. Resources required :

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.
4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.
5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/circuit/Diagram/Description:

Linux is a Unix-Like operating system. All the Linux/Unix commands are run in the terminal provided by the Linux system. This terminal is just like the command prompt of Windows OS. Linux/Unix commands are case-sensitive. The terminal can be used to accomplish all Administrative tasks. This includes package installation, file manipulation, and user management. Linux terminal is user-interactive. The terminal outputs the results of commands which are specified by the user itself. Execution of typed commands is done only after you press the Enter key.

8. Test cases:

6. The Student should check the different variations of the commands by applying different options.
7. Outputs with different inputs must be given and appropriate results must be recorded after execution.
8. Students must also record the errors while executing the commands, so that they get deeper insights on how a command should be executed.

9. Sample output:

ls command:

```
root@JOSHUA:~# ls
a.txt  b.txt  b.txt~  c.txt  fifth  first  fourth  historycommands  joshua.txt  second  third
root@JOSHUA:~#
```

cat command:

```
root@JOSHUA:~# cat a.txt
these are the contents of a.txt
root@JOSHUA:~#
```

10.Practical Related Questions:

1. How will you List files from a directory?
Ans: The Linux file listing command ‘ls‘ comes to rescue here.

2. How will you list all the directories only using echo command?
Ans: echo */
Output: fifth/ first/ fourth/ second/ third/

3. How do you list all the files within a directory including hidden files, but do not list implied ‘.’ and ‘..’?
Ans:

```
root@JOSHUA:~# ls -A
.bash_history  .cache   .local    .viminfo  b.txt   c.txt  first   historycommands  second
.bashrc        .config  .profile  a.txt     b.txt~  fifth  fourth  joshua.txt       third
```

11 .Exercise Questions :

1) What is Linux?
Linux is an operating system based on UNIX and was first introduced by Linus Torvalds. It is based on the Linux Kernel and can run on different hardware platforms manufactured by Intel, MIPS, HP, IBM, SPARC, and Motorola. Another popular element in Linux is its mascot, a penguin figure named Tux.

2) What is the difference between UNIX and LINUX?
Unix originally began as a propriety operating system from Bell Laboratories, which later on spawned into different commercial versions. On the other hand, Linux is free, open source and intended as a non-proprietary operating system for the masses.

3) What is BASH?
BASH is short for Bourne Again SHell. It was written by Steve Bourne as a replacement to the original Bourne Shell (represented by /bin/sh). It combines all the features from the original version of Bourne Shell, plus additional functions to make it easier and more convenient to use. It has since been adapted as the default shell for most systems running Linux.

4) What is Linux Kernel?

The Linux Kernel is a low-level systems software whose main role is to manage hardware resources for the user. It is also used to provide an interface for user-level interaction.

5) What is LILO?
LILO is a boot loader for Linux. It is used mainly to load the Linux operating system into main memory so that it can begin its operations.

6) What is a swap space?
Swap space is a certain amount of space used by Linux to temporarily hold some programs that are running concurrently. This happens when RAM does not have enough memory to hold all programs that are executing.

7) What is the advantage of open source?
Open source allows you to distribute your software, including source codes freely to anyone who is interested. People would then be able to add features and even debug and correct errors that are in the source code. They can even make it run better and then redistribute these enhanced source code freely again. This eventually benefits everyone in the community.

8) What are the basic components of Linux?
Just like any other typical operating system, Linux has all of these components: kernel, shells and GUIs, system utilities, and an application program. What makes Linux advantageous over other operating systems is that every aspect comes with additional features and all codes for these are downloadable for free.

9) What is the basic difference between BASH and DOS?
The key differences between the BASH and DOS console lie in 3 areas:

- BASH commands are case sensitive while DOS commands are not;

- Under BASH, / character is a directory separator and \ acts as an escape character. Under DOS, / serves as a command argument delimiter and \ is the directory separator

- DOS follows a convention in naming files, which is an 8 character file name followed by a dot and 3 characters for the extension. BASH follows no such convention.

10) What is the pwd command?
The pwd command is short for print working directory command.
Example:
pwd

11. What are inode and process id?
inode is the unique name given by the operating system to each file. Similarly, process id is the unique id given to each process.

12. Which are the Linux Directory Commands?

There are 5 main Directory Commands in Linux:
pwd: Displays the path of the present working directory.
Syntax: $ pwd
ls: Lists all the files and directories in the present working directory.
Syntax: $ ls
cd: Used to change the present working directory.
Syntax: $ cd <path to new directory>

mkdir: Creates a new directory
Syntax: $ mkdir <name (and path if required) of new directory>
rmdir: Deletes a directory
Syntax: $ rmdir <name (and path if required) of directory>

PRACTICAL 2: Shell programming: Simple logic programs.

1. Practical significance :

1. Shell scripts can take input from a user or file and output them to the screen.
    - Whenever you find yourself doing the same task over and over again you should use shell scripting, i.e., repetitive task automation.
        - Creating your own power tools/utilities.
        - Automating command input or entry.
        - Customizing administrative tasks.
        - Creating simple applications.
        - Since scripts are well tested, the chances of errors are reduced while configuring services or system administration tasks such as adding new users.
2. It is a powerful programming method that can help you learn the command-line better, save time, and do away with tedious file management tasks.
3. Scripting allows you to use programming functions – such as 'for' loops, if/then/else statements, and so forth – directly within your operating system's interface. And, you don't have to learn another language because you're using what you already know: the command-line.

2. Relevant Program Outcomes :PO 1, PO 2, PO 3, PO 4, PO 12

3. Competency and practical skills :

The practical is expected to develop the following skills

1. Able to write basic shell scripting code for simple requirements such as file manipulation, program execution, and printing text.

4. Prerequisites :

2. Familiarity with the Unix/Linux command line and running simple commands

5. Resources required :

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.
4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.
5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/circuit/Diagram/Description:

A shell script is a computer program designed to be run by the Unix/Linux shell which could be one of the following:

- The Bourne Shell
- The C Shell
- The Korn Shell
- The GNU Bourne-Again Shell

A shell is a command-line interpreter and typical operations performed by shell scripts include file manipulation, program execution, and printing text.

- Usually shells are interactive, that means, they accept commands as input from users and execute them. However sometimes we want to execute a bunch of commands routinely, so we have to type in all commands each time in the terminal.

As shell can also take commands as input from file we can write these commands in a file and can execute them in shell to avoid this repetitive work. These files are called **Shell Scripts** or **Shell Programs**. Shell scripts are similar to the batch file in MS-DOS. Each shell script is saved with **.sh** file extension

eg.**myscript.sh**
A shell script has syntax just like any other programming language. If you have any prior experience with any programming language like Python, C/C++ etc. it would be very easy to get started with it.

- A shell script comprises following elements –
    - Shell Keywords – if, else, break etc.
    - Shell commands – cd, ls, echo, pwd, touch etc.
    - Functions
    - Control flow – if..then..else, case and shell loops etc

8. Test cases:

1. The Student should Execute Scripting programs which include shell keywords,commands, Functions and Control flow statements.
2. Outputs with different inputs must be given and appropriate results must be recorded after execution.
3. Students must also record the errors while executing the commands, so that they get deeper insights on how a Scripting program should be written and executed.

9. Sample output:

Example :Implementing if statement

```
#Initializing two variables
a=10
b=20
 #Check whether they are equal
if [ $a == $b ]
then
   echo "a is equal to b"
fi
 #Check whether they are not equal
if [ $a != $b ]
then
   echo "a is not equal to b"
fi
```

Output:

```
$bash main.sh
a is not equal to b
```

10.Practical Related Questions:

1. Write a shell script to get the current date, time, username and current working directory.

```
echo "Hello, $LOGNAME"
echo "Current date is $(date)"
echo "User is '$(whoami)'"
echo "Current directory '$PWD'"
```

2. How do you ask for input in a shell script from the terminal?

```
echo ‘Please enter your name’
read name
echo “My Name is $name”
```

3. Write a menu driven script using the select statement to print calories for food items such as pizza,burger, Salad, Pasta etc.

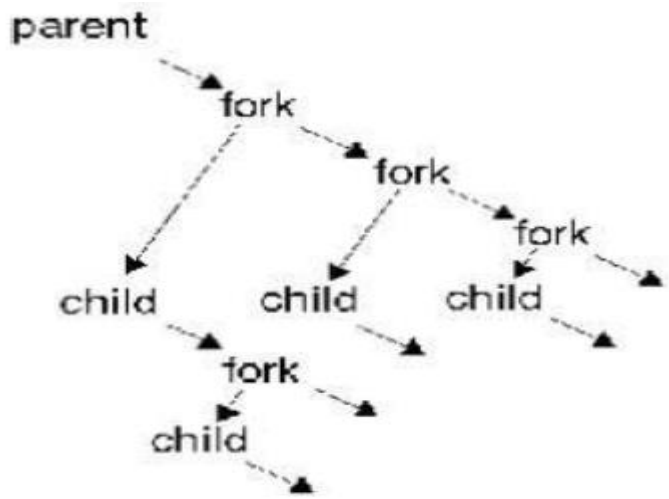
4. Write a shell script that, given a filename as the argument will count vowels, blank spaces, characters, number of lines and symbols.

11 .Exercise Questions :

1. Create a bash file with the name, ‘while_example.sh’, to know the use of while loop. In the example, the while loop will iterate for 5 times. The value of the count variable will increment by 1 in each step. When the value of the count variable is 5 then the while loop will terminate.
2. **Case** statement is used as the alternative of **if-elseif-else** statement. The starting and ending block of this statement is defined by ‘**case**’ and ‘**esac**’. Create a new file named, ‘**case_example.sh**’ and write a shell script using a case statement..

PRACTICAL 3 :

I) Analyze the below situation and develop a program for creating processes as required. Print the PID of each process in a convenient way to understand.



II)Write a program to create two processes P1 and P2. P1 takes a string and passes it to P2. P2 concatenates the received string with another string without using string function and sends it back to P1 for printing.

1. Practical significance :

I)

1. Fork() can be used at the place where there is division of work like a server has to handle multiple clients, So parent have to accept the connection on a regular basis, So server does fork for each client to perform read-write.
2. Multiprocessing is central to computing. For example, your IE or Firefox can create a process to download a file for you while you are still browsing the internet. Or, while you are printing out a document in a word processor, you can still look at different pages and still do some editing with it.

II) Pipe is a communication medium between two or more related or interrelated processes. It can be either within one process or a communication between the child and the parent processes. Communication can also be multi-level such as communication between the parent, the child and the grand-child, etc. Communication is achieved by one process writing into the pipe and other reading from the pipe. To achieve the pipe system call, create two files, one to write into the file and another to read from the file.

2. Relevant Program Outcomes :

I) PO 1, PO 2, PO 3, PO 4, PO 12
II)PO 1, PO 2, PO 3, PO 4, PO 12

3. Competency and practical skills :

The practical is expected to develop the following skills

1. Able to understand the fork() system call.
2. Able to create multiple processes , print their process identifiers.
3. Able to understand pipe concepts and interprocess communication.

4. Prerequisites :

1. A Good Understanding of fork() system call.
2. Knowledge about what is a process in the Operating System.
3. Understanding header files like **<sys/types.h>, <unistd.h>**
4. Understanding **pid_t fork(void);**
5. Understanding I/O System calls

5. Resources required :

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.
4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.
5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

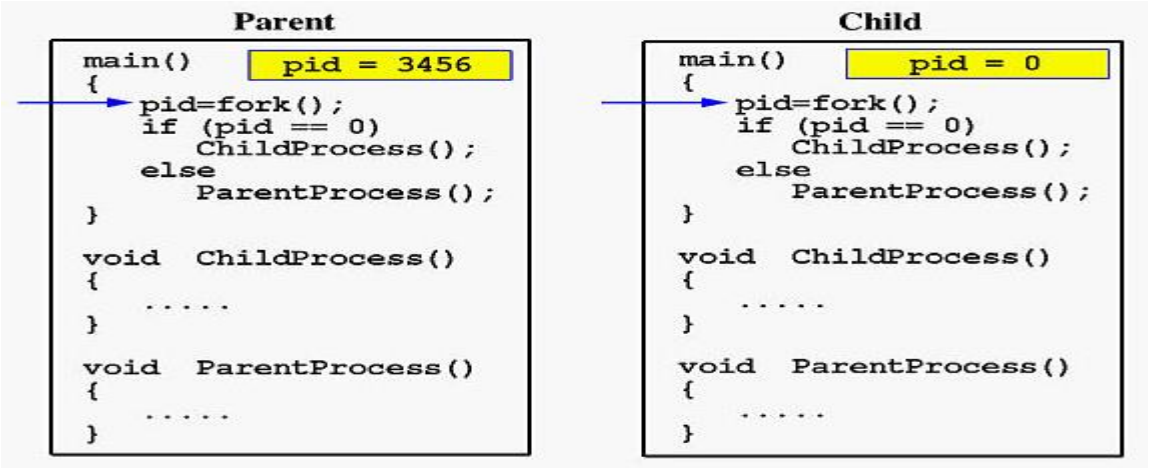
7. Algorithm/Diagram/Description:

I) System call fork() is used to create processes. It takes no arguments and returns a process ID. The purpose of fork() is to create a *new* process, which becomes the *child* process of the caller. After a new child process is created, *both* processes will execute the next instruction following the *fork()* system call. Therefore, we have to distinguish the parent from the child. This can be done by testing the returned value of fork():

- If fork() returns a negative value, the creation of a child process was unsuccessful.
- fork() returns a zero to the newly created child process.
- fork() returns a positive value, the *process ID* of the child process, to the parent. The returned process ID is of type pid_t defined in sys/types.h. Normally, the process ID is an integer.

Moreover, a process can use the function getpid() to retrieve the process ID assigned to this process.

Therefore, after the system calls fork(), a simple test can tell which process is the child. Please note that Unix will make an exact copy of the parent's address space and give it to the child. Therefore, the parent and child processes have separate address spaces.



II) Pipe() is used for passing information from one process to another. pipe() is unidirectional therefore, for two-way communication between processes, two pipes can be set up, one for each direction.

Example:

```
int fd[2];
pipe(fd);
fd[0]; //-> for using read end
fd[1]; //-> for using write end
```

Inside Parent Process : We firstly close the reading end of the first pipe (fd1[0]) then write the string though the writing end of the pipe (fd1[1]). Now the parent will wait until the child process is finished. After the child process, the parent will close the writing end of the second pipe(fd2[1]) and read the string through the reading end of pipe (fd2[0]).

Inside Child Process : Child reads the first string sent by the parent process by closing the writing end of the pipe (fd1[1]) and after reading concatenates both strings and passes the string to the parent process via fd2 pipe and will exit.

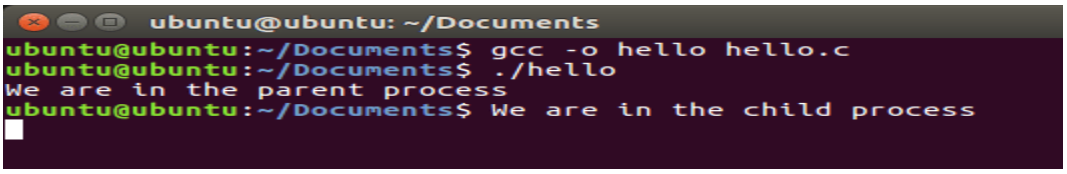
8. Test cases:

1. The students must execute different variations of the fork() system call and check how different child processes are being created.
2. Should record the pid values returned by child processes while nesting the fork system calls.
3. Execute pipe() system call for interprocess communication.

9. Sample output:

I)
```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
int main()
{
   pid_t p;
   p = fork();
   if(p==-1)
   {
      printf("There is an error while calling fork()");
   }
```

```
    if(p==0)
    {
       printf("We are in the child process");
    }
    else
    {
       printf("We are in the parent process");
    }
    return 0;
}
```



II)**int pipe(int fds[2]);**

Parameters :

fd[0] will be the fd(file descriptor) for the read end of the pipe.

fd[1] will be the fd for the write end of the pipe.

Returns : 0 on Success.

-1 on error.

10.Practical Related Questions:

1. Write a program to print Hello eight times using fork system call
2. Write a program using pipes to communicate between two processes.

11 .Exercise Questions :

1. What is the output of this program?

```
#include<stdio.h>
 int main()
{
   fork();
   printf("os lab\n");
   return 0;
}
```

2. What is the output of this program?

```
#include<stdio.h>
#include<unistd.h>
 int main()
{
   pid_t child;
   child = fork();
   printf("%d\n",child);
   return 0;
}
```

3. Which of the following system call is used for inter-process communication?
   a) fork
   b) pipe
   c) fcntl
   d) exec

PRACTICAL 4 :

i) A washing machine which requires the process to be executed sequentially. Consider the processes P1, P2, P3, P4 whose arrival times are 1, 5, 9, 10 and burst times are 4, 3, 5, 2 respectively. Implement an appropriate algorithm. Find the CPU idle time, so that the water can be supplied during that period of time.

ii) Implement shortest job first for the following data:

Consider the following set of processes, CPU burst time, Arrival time. Calculate the average waiting time, average response time and average turnaround time.

Process	Burst Time	Arrival Time
P1	3	0
P2	6	2
P3	4	4
P4	5	6
P5	2	8

iii) Implement Round Robin for the following data

Consider the following set of processes and length of the CPU burst time given in milliseconds. The processes are assumed to have arrived in the order P1, P2, P3, P4, P5 all at time 0 and time quantum in RR=1.Calculate the average waiting time, response time and turnaround time.

Process	Burst Time
P1	10
P2	1
P3	2
P4	1
P5	5

1. Practical significance :

In Multiprogramming Systems, the Operating System schedules processes on the CPU to have maximum utilization of it, and this procedure is called CPU scheduling. The Operating System uses various scheduling Algorithms to schedule the Processes.They are as Follows:

i) FCFS : A real-life example of the FCFS method is buying a movie ticket on the ticket counter. In this scheduling algorithm, a person is served according to the queue manner. The person who arrives first in the queue first buys the ticket and then the next one. This will continue until the last person in the queue purchases the ticket.

ii) SJF : Shortest job first (SJF) or shortest job next, is a scheduling policy that selects the waiting process with the smallest execution time to execute next. SJN is a non-preemptive algorithm.

Shortest Job first has the advantage of having minimum average waiting time among all Scheduling Algorithms.

iii) Round Robin :Round-robin (RR) is one of the algorithms employed by process and network schedulers in computing. As the term is generally used, time slices (also known as time quanta) are assigned to each process in equal portions and in circular order, handling all processes without priority (also known as cyclic executive). Round-robin scheduling is simple, easy to implement,

and starvation-free. Round-robin scheduling can be applied to other scheduling problems, such as data packet scheduling in computer networks.

2. Relevant Program Outcomes :PO 1, PO 2, PO 3, PO 4, PO 12

3. Competency and practical skills :

The practical is expected to develop the following skills

1. Able to Understand FIFO,SJF and Round Robin Algorithms and their Applications.
2. Able to find out waiting time, turnaround time, and comparison of the algorithms.

4. Prerequisites :

1. Understand the concept of Multiprocessing, and scheduling.
2. Understand the theory behind the FIFO , SJF, ROUND ROBIN Scheduling Algorithms.

5. Resources required :

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.
4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.
5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/Description:

FCFS:

1- Input the processes along with their burst time (bt).
2- Find waiting time (wt) for all processes.
3- As first process that comes need not to wait so
waiting time for process 1 will be 0 i.e. wt[0] = 0.
4- Find waiting time for all other processes i.e. for
process i ->
wt[i] = bt[i-1] + wt[i-1] .
5- Find turnaround time = waiting_time + burst_time
for all processes.
6- Find average waiting time =
total_waiting_time / no_of_processes.
7- Similarly, find average turnaround time =
total_turn_around_time / no_of_processes.

SJF :

Algorithm:

1. Sort all the processes according to the arrival time.
2. Then select that process which has minimum arrival time and minimum Burst time.
3. After completion of process make a pool of processes which after till the completion of previous process and select that process among the pool which is having minimum Burst time.
    a. Completion Time: Time at which process completes its execution.
    b. Turn Around Time: Time Difference between completion time and arrival time. Turn Around Time = Completion Time – Arrival Time
    c. Waiting Time(W.T): Time Difference between turn around time and burst time.
    Waiting Time = Turnaround Time – Burst Time

ROUND ROBIN:

Algorithm:

* The CPU scheduler picks the process from the circular/ready queue , set a timer to interrupt it after 1 time slice / quantum and dispatches it .

* If process has burst time less than 1 time slice/quantum

> Process will leave the CPU after the completion

> CPU will proceed with the next process in the ready queue / circular queue .

else If process has burst time longer than 1 time slice/quantum

> Timer will be stopped . It causes interruption to the OS .

> Executed process is then placed at the tail of the circular / ready queue by applying the context switch

> CPU scheduler then proceeds by selecting the next process in the ready queue .

1. Completion Time: Time at which process completes its execution.
2. Turn Around Time: Time Difference between completion time and arrival time. Turn Around Time = Completion Time – Arrival Time
3. Waiting Time(W.T): Time Difference between turn around time and burst time.
   Waiting Time = Turnaround Time – Burst Time

8. Test cases:

1. The students must execute FCFS,SJF,Round Robin Algorithms with different arrival times.
2. Should Compare all the algorithms for performance evaluation.

9. Sample output:

Processes	Burst time	Waiting time
1	10	0
2	5	10
3	8	15

Average waiting time = 8.33333

Average turnaround time = 16

10.Practical Related Questions:

1. Compare FCFS,SJF,ROUND ROBIN Algorithms wrt average waiting time and turnaround time.
2. What are the disadvantages of FCFS Algorithms?
3. Write down the formulas for turnaround time and average waiting time.

11 .Exercise Questions :

1. What is a process and process table? What are different states of process?
2. What is a Thread? What are the differences between process and thread?
3. What are the different scheduling algorithms?
4. What is Belady's Anomaly?
5. What are the advantages of the Round Robin Algorithm?

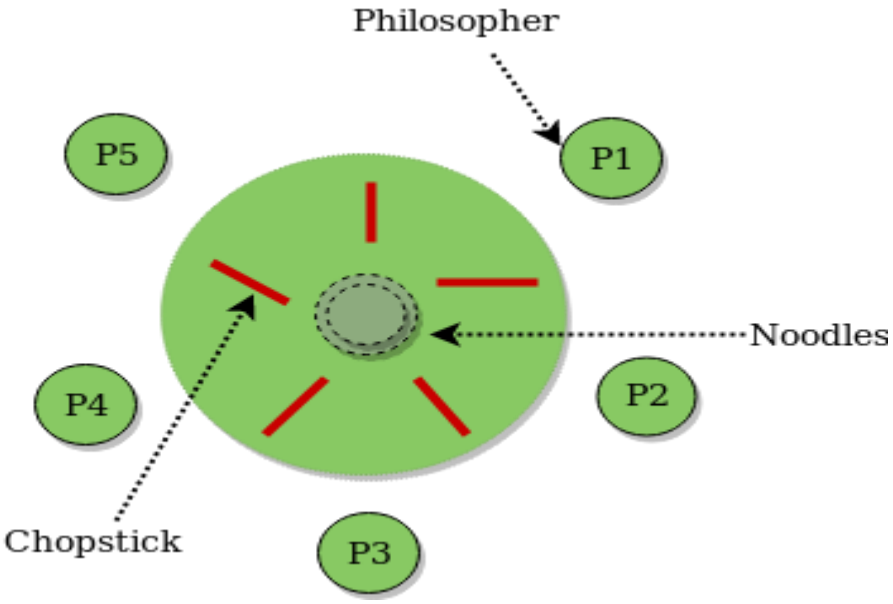
PRACTICAL 5 :

Develop a program to provide synchronization among the 5 philosophers in Dining Philosophers problem using semaphore.

1. Practical significance :

The dining philosophers problem used to demonstrate the concept of deadlock that is - another classic synchronization problem which is used to evaluate situations where there is a need of allocating multiple resources to multiple processes.

The Dining Philosopher Problem – The Dining Philosopher Problem states that K philosophers seated around a circular table with one chopstick between each pair of philosophers. There is one chopstick between each philosopher. A philosopher may eat if he can pick up the two chopsticks adjacent to him. One chopstick may be picked up by any one of its adjacent followers but not both.



2. Relevant Program Outcomes :PO 1, PO 2, PO 3, PO 4, PO 12

3. Competency and practical skills :

The practical is expected to develop the following skills

1. Able to Understand dining philosophers problem and its Applications.
2. Able to evaluate situations where there is a need of allocating multiple resources to multiple processes.

4. Prerequisites :

1. Understand the concept of Process Synchronization,Semaphores.
2. Understand the theory behind the Dining Philosopher Solutions using Monitors .

5. Resources required :

S.No	Name of the Resource	Broad Specification(Approx.)

1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.
4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.
5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/Description:

Five philosophers are sitting at a rounded dining table.

1. There is one chopstick between each pair of adjacent philosophers.
2. Philosophers are either thinking or eating.
3. Whenever a philosopher wishes to eat, she first needs to find two chopsticks.
4. If the hungry philosopher does not have two chopsticks (i.e. one or two of her neighbors already picked up the chopstick) she will have to wait until both chopsticks are available.
5. When a philosopher finishes eating, she puts down both chopsticks to their original places, and resumes thinking.
6. There is an infinite amount of food on their plate, so they only need to worry about the chopsticks.

There are a few conditions:

1. Philosophers are either thinking or eating. They do not talk to each other.
2. Philosophers can only fetch chopsticks placed between them and their neighbors.
3. Philosophers cannot take their neighbors' chopsticks away while they are eating.
4. Hopefully no philosophers should starve to death (i.e. wait over a certain amount of time before she acquires both chopsticks).

8. Test cases:

The students should try the various possibilities to solve concurrency control problems.

9. Sample output:

Fork 1 taken by Philosopher 1

Fork 2 taken by Philosopher 2

Fork 3 taken by Philosopher 3

Philosopher 4 is waiting for fork 3

Till now num of philosophers completed dinner are 0

Fork 4 taken by Philosopher 1

Philosopher 2 is waiting for Fork 1

Philosopher 3 is waiting for Fork 2

Philosopher 4 is waiting for fork 3

10.Practical Related Questions:

1. What is the problem if all philosophers simultaneously pick up their left fork?

   It leads to the condition of deadlock and none of the **philosophers** can eat.

2. Why might a group of dining philosophers starve?

   Resource **starvation might** also occur independently of deadlock if a particular **philosopher** is unable to acquire both forks because of a timing problem

11.Exercise Questions :

1. How deadlock is possible with the Dining Philosophers Problem?

   Deadlock would occur if every **philosopher** holds a left chopstick and waits perpetually for a right chopstick (or vice versa). Originally used as a means of illustrating the **problem** of **deadlock**, this system reaches **deadlock** when there is a 'cycle of unwarranted requests'.

2. Which solution does not suffer from deadlock in the Dining Philosophers Problem?

PRACTICAL 6 :

Develop a program to provide synchronization among the producer and consumer processes in producer – consumer problem using a monitor.

1. Practical significance :

The producer consumer problem is a synchronization problem. There is a fixed size buffer and the producer produces items and enters them into the buffer. The consumer removes the items from the buffer and consumes them.

2. Relevant Program Outcomes :PO 1, PO 2, PO 3, PO 4, PO 12

3. Competency and practical skills :

The practical is expected to develop the following skills

1. Able to Understand synchronization mechanism among the producer consumer processes
2. Able to evaluate with fixed size buffers , and provide synchronization.

4. Prerequisites :

Semaphores in operating system, Inter Process Communication

5. Resources required :

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.
4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.
5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/Description:

The producer consumer problem is a synchronization problem. There is a fixed size buffer and the producer produces items and enters them into the buffer. The consumer removes the items from the buffer and consumes them.

A producer should not produce items into the buffer when the consumer is consuming an item from the buffer and vice versa. So the buffer should only be accessed by the producer or consumer at a time.

Producer Process

The code that defines the producer process is given below −

```
do {
  .
  . PRODUCE ITEM
  .
  wait(empty);
  wait(mutex);
  .
  . PUT ITEM IN BUFFER
  .
  signal(mutex);
  signal(full);

} while(1);
```

In the above code, mutex, empty and full are semaphores. Here mutex is initialized to 1, empty is initialized to n (maximum size of the buffer) and full is initialized to 0.

The mutex semaphore ensures mutual exclusion. The empty and full semaphores count the number of empty and full spaces in the buffer.

After the item is produced, wait operation is carried out on empty. This indicates that the empty space in the buffer has decreased by 1. Then the wait operation is carried out on mutex so that the consumer process cannot interfere.

After the item is put in the buffer, signal operation is carried out on mutex and full. The former indicates that the consumer process can now act and the latter shows that the buffer is full by 1.

Consumer Process

The code that defines the consumer process is given below:

```
do {

  wait(full);
  wait(mutex);
  . .
  . REMOVE ITEM FROM BUFFER
  .
  signal(mutex);
  signal(empty);
  .
  . CONSUME ITEM
  .
} while(1);
```

The wait operation is carried out on full. This indicates that items in the buffer have decreased by 1. Then the wait operation is carried out on mutex so that the producer process cannot interfere.

Then the item is removed from the buffer. After that, signal operation is carried out on mutex and empty. The former indicates that consumer process can now act and the latter shows that the empty space in the buffer has increased by 1

8. Test cases:

The student must make sure that the producer won't try to add data into the buffer if it's full and that the consumer won't try to remove data from an empty buffer.

9. Sample output:

Producer produced-0
Producer produced-1
Consumer consumed-0
Consumer consumed-1
Producer produced-2

10.Practical Related Questions:

1. What are the different ways to solve producer consumer problems?

   The solution for the producer is to either go to **sleep** or discard data if the buffer is full. The next time the consumer removes an item from the buffer, it notifies the producer, who starts to fill the buffer again. In the same way, the consumer can go to **sleep** if it finds the buffer empty.

2. What is the producer consumer problem explain with an example?

   The Producer-Consumer problem is a classic problem that is used for multi-process synchronization i.e. synchronization between more than one processes. In the producer-consumer problem, there is one Producer that is producing something and there is one Consumer that is consuming the products produced by the Producer.

11.Exercise Questions :

1. How can we achieve the synchronization using semaphore for producer consumer problems?

   The producer consumer problem is a synchronization problem. There is a fixed size buffer and the producer produces items and enters them into the buffer. The consumer removes the items from the buffer and consumes them.

PRACTICAL 7 :

Consider the following data:

Process	Allocation A B C D	Max A B C D	Available A B C D
P1	0 0 1 2	0 0 1 2	2 1 0 0
P2	2 0 0 0	2 7 5 0	
P3	0 0 3 4	6 6 5 6	
P4	2 3 5 4	4 3 5 6	
P5	0 3 3 2	0 6 5 2	

i) Calculate the need matrix
ii) Is this system currently in a safe or unsafe state?
iii) Is the system currently deadlock or not?
iv) Which process, if any, or may become deadlocked?

1. Practical significance :

The banker's algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for predetermined maximum possible amounts of all resources, then makes an "s-state" check to test for possible activities, before deciding whether allocation should be allowed to continue.

2. Relevant Program Outcomes :PO 1, PO 2, PO 3, PO 4, PO 12

3. Competency and practical skills :

The practical is expected to develop the following skills

1. Able to Understand how to perform resource allocation and
2. Able to evaluate whether the current state of the system is safe or not.
3. Able to evaluate if there is any deadlock in the system.
4. Can calculate the need matrix.

4. Prerequisites : Basic understanding of Deadlocks and other related concepts, Data structures.

5. Resources required :

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.
4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.
5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/Description:

Following **Data structures** are used to implement the Banker's Algorithm:

Let **'n'** be the number of processes in the system and **'m'** be the number of resources types.

Available :

- It is a 1-d array of size **'m'** indicating the number of available resources of each type.
- Available[j] = k means there are **'k'** instances of resource type $\mathbf{R_j}$

Max :

- It is a 2-d array of size '**n*m'** that defines the maximum demand of each process in a system.
- Max[i, j] = k means process $\mathbf{P_i}$ may request at most **'k'** instances of resource type $\mathbf{R_j}$.

Allocation :

- It is a 2-d array of size **'n*m'** that defines the number of resources of each type currently allocated to each process.
- Allocation[i, j] = k means process $\mathbf{P_i}$ is currently allocated **'k'** instances of resource type $\mathbf{R_j}$

Need :

- It is a 2-d array of size **'n*m'** that indicates the remaining resource need of each process.
- Need [i, j] = k means process $\mathbf{P_i}$ currently need **'k'** instances of resource type $\mathbf{R_j}$ for its execution.
- Need [i, j] = Max [i, j] – Allocation [i, j]

$Allocation_i$ specifies the resources currently allocated to process P_i and $Need_i$ specifies the additional resources that process P_i may still request to complete its task.

Banker's algorithm consists of Safety algorithm and Resource request algorithm

Safety Algorithm

The algorithm for finding out whether or not a system is in a safe state can be described as follows:

1) Let Work and Finish be vectors of length 'm' and 'n' respectively.

Initialize: Work = Available

Finish[i] = false; for i=1, 2, 3, 4….n

2) Find an i such that both

a) Finish[i] = false

b) $Need_i$ <= Work

if no such i exists goto step (4)

3) Work = Work + Allocation[i]

Finish[i] = true

goto step (2)

4) if Finish [i] = true for all i

then the system is in a safe state

Resource-Request Algorithm

Let $Request_i$ be the request array for process P_i. $Request_i[j] = k$ means process P_i wants k instances of resource type R_j. When a request for resources is made by process P_i, the following actions are taken:

1) If $Request_i$ <= $Need_i$

Goto step (2) ; otherwise, raise an error condition, since the process has exceeded its maximum claim.

2) If $Request_i$ <= Available

Goto step (3); otherwise, P_i must wait, since the resources are not available.

3) Have the system pretend to have allocated the requested resources to process Pi by modifying the state as

follows:

Available = Available – Requesti

$Allocation_i = Allocation_i + Request_i$

$Need_i = Need_i – Request_i$

8. Test cases:

The students should test for safety by simulating the allocation for determining the maximum amount available for all resources**.**

9. Sample output:

Enter the number of resources: 4

Enter the number of processes: 5

Enter Claim Vector: 8 5 9 7

Enter Allocated Resource Table: 2 0 1 1 0 1 2 1 4 0 0 3 0 2 1 0 1 0 3 0

Enter Maximum Claim table: 3 2 1 4 0 2 5 2 5 1 0 5 1 5 3 0 3 0 3 3

The Claim Vector is: 8 5 9 7
The Allocated Resource Table:

2	0	1	1
0	1	2	1
4	0	0	3
0	2	1	0
1	0	3	0

The Maximum Claim Table:

3	2	1	4
0	2	5	2
5	1	0	5
1	5	3	0
3	0	3	3

Allocated resources: 7 3 7 5
Available resources: 1 2 2 2

Process3 is executing.

The process is in safe state.
Available vector: 5 2 2 5
Process1 is executing.

The process is in safe state.
Available vector: 7 2 3 6
Process2 is executing.

The process is in safe state.
Available vector: 7 3 5 7
Process4 is executing.

The process is in safe state.
Available vector: 7 5 6 7

Process5 is executing.

The process is in safe state.
Available vector: 8 5 9 7

10.Practical Related Questions:

1. Why Banker's algorithm is used?
   Banker's Algorithm is used majorly in the banking system to avoid deadlock. It helps you to identify whether a loan will be given or not. This **algorithm is used** to test for safely simulating the allocation for determining the maximum amount available for all resource.

11.Exercise Questions :

1. What is the drawback of Banker's algorithm?
   It requires the number of processes to be fixed; no additional processes can start while it is executing. It requires that the number of resources remain fixed; no resource may go down for any reason without the possibility of deadlock occurring.

2. What are the possible side effects of deadlock prevention?
   Deadlock has four necessary conditions to hold simultaneously true. By ensuring that at least one of these conditions cannot hold, we can **prevent** the occurrence of a **deadlock**. **Possible side effects** of **preventing deadlocks** by this method, however, are low device utilization and reduced system throughput.

3. What is the difference between deadlock prevention and avoidance?
   The main **difference between deadlock prevention** and **deadlock avoidance** is that the **deadlock prevention** ensures that at least one of the necessary conditions to cause a **deadlock** will never occur, while **deadlock avoidance** ensures that the system will not enter an unsafe state.

PRACTICAL 8 :

Consider the following scenario: A process has been allocated 3 page frames. Assume that none of the pages of the process are available in the memory initially. The process makes the following sequence of page references (reference string): 1, 2, 1, 3, 7, 4, 5, 6, 3, 1, 2, 4, 6, 3, 1. Find out a page replacement policy which gives the least number of page faults.

1. Practical significance :

In operating systems, whenever a new page is referred to and not present in memory, page fault occurs and the Operating System replaces one of the existing pages with a newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce the number of page faults.

2. Relevant Program Outcomes :PO 1, PO 2, PO 3, PO 4, PO 12

3. Competency and practical skills :

The practical is expected to develop the following skills

1. Able to Understand different page replacement algorithms.
2. Able to suggest different ways to decide which page to replace
3. The student can evaluate all algorithms to reduce the number of page faults.

4. Prerequisites :

The students should have sufficient knowledge on

1.How Virtual Memory is implemented and how Paging is used, the various types of Page replacement algorithms

2.What is page fault.

5. Resources required :

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.

4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.
5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/Description:

A) FIRST IN FIRST OUT (FIFO) PAGE REPLACEMENT ALGORITHM:

ALGORITHM:
Step1: Start
Step2: Read no of frames and reference
Step3: Read the frame list
Step4: Copy the reference list into stack
Step5: Insert the frame number into frame by FIFO
Step6: Display the frame stack S
Step7: Stop

B) LEAST RECENTLY USED(LRU) PAGE REPLACEMENT ALGORITHM:
ALGORITHM:
Step1: Start
Step2: Read no of frames and reference and reference list values
Step3: Insert the element into the frame by least recently used
Step4: While inserting an element i, the frame contents also having the same element i occur then print no page fault occurs
Step5: Otherwise print page fault occurs and continue from step3 until reference list number becomes zero
Step6: Stop

C) OPTIMAL PAGE REPLACEMENT ALGORITHM:
ALGORITHM:
Step1: Start
Step2: Read the number of frames, reference and reference list
Step3: Replace the page fault that will not be used for longer period of time
Step4: Count and print the no. of page faults occur
Step5: Display the reference list stack
Step6: Stop

8. Test cases:

1. The students must execute FIFO,LRU,OPTIMAL PAGE REPLACEMENT Algorithms with different number of frames.
2. Should Compare all the algorithms for performance evaluation.

9. Sample output:

Enter no of pages:18

Enter reference string: 7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7

Enter no of frames:3

7

7 0

7 0 1

2 0 1

2 0 1

2 0 3

2 0 3

2 4 3

2 4 3

2 4 3

2 0 3

2 0 3

2 0 3

2 0 1

2 0 1

2 0 1

2 0 1

2 0 7

total no of page faults=9

10.Practical Related Questions:

1.Consider the following reference string: 0, 2, 1, 6, 4, 0, 1, 0, 3, 1, 2, 1.

Using FIFO page replacement algorithm find out the number of page faults.

2.Consider a main memory with five page frames and the following sequence of page references: 3, 8, 2, 3, 9, 1, 6, 3, 8, 9, 3, 6, 2, 1, 3. Which one of the following is true with respect to page replacement policies First-In-First Out (FIFO) and Least Recently Used (LRU)?

(A) Both incur the same number of page faults

(B) FIFO incurs 2 more page faults than LRU

(C) LRU incurs 2 more page faults than FIFO

(D) FIFO incurs 1 more page faults than LRU

11.Exercise Questions :

1.What is page replacement algorithm in operating system?

The page replacement algorithm decides which memory page is to be replaced. The process of replacement is sometimes called swap out or write to disk. Page replacement is done when the requested page is not found in the main memory (page fault)

2.What is the best page replacement algorithm? Why.?

Optimal Page Replacement algorithm is the best page replacement algorithm as it gives the least number of page faults. It is also known as OPT, clairvoyant replacement algorithm, or Belady's optimal page replacement policy

3.What are the various methods of page replacement?

Page Replacement Algorithms :

- First In First Out (FIFO) – This is the simplest page replacement algorithm. ...
- Optimal Page replacement – In this algorithm, pages are replaced which would not be used for the longest duration of time in the future. ...
- Least Recently Used – In this algorithm page will be replaced which is least recently used

4.Which page replacement algorithm is used in Windows?

FIFO

PRACTICAL 9:

Simulate the Virtual Memory concepts.

1. Practical significance :

1. Read a file containing a list of logical addresses.

2. Translate logical addresses into physical addresses.

3. Output the value of the byte stored at the translated physical address.

Implementation Details

1.1. Extracting Page Numbers and Offset

Write a function is to extract page number and an offset from the following addresses: 1, 256, 32768, 23769, 128, 65534, 33153 A straightforward way of extracting page number and offset is to use bit-masking and bit-shifting operators. After you fully implement and test this function, you can integrate this function into your simulated virtual memory system.

1.2. Implementing the Page Table

Your system must handle page faults using the TLB and the page table. Before implement the TLB, you should focus on the implementation of a page table. Please make sure your system has a functional page table prior to the development of the TLB. This strategy is feasible, because your system can perform without a TLB, the goal of which is to improve the performance of a virtual memory system.

1.3. Implementing the TLB along with a Replacement Strategy

It is strongly suggested that you implement the TLB and paging module separately and then integrate them together. Assuming that you have successfully implemented the page table in your system, now we can start the TLB implementation. Recall that the TLB has a total of 16 entries, which means you must implement a replacement strategy when your system is about to update a full TLB. The easiest strategy to handle TLB faults is FIFO (i.e., First In and First Out). You must also implement the LRU (i.e., Least Recently Used) strategy to optimize the TLB hit rate. Your system must guarantee that the TLB state is initialized properly.

2. Relevant Program Outcomes :

3. Competency and practical skills :

The practical is expected to develop the following skills

1. Ability to work on Linux operating systems.
2. To design a simple virtual memory system.
3. Ability to edit files, use basic utilities and commands, navigate through the file system.

4. Prerequisites :

1. The basics of the C programming language
2. The very basics of the Linux filesystem and the shell

3. Use GDB to debug your C program.

5. Resources required:

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.

2. Ensure the keyboard, mouse and monitor are properly working.

3. Ensure that there are no power fluctuations while executing the commands.

4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.

5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/circuit/Diagram/Description:

Description:

Implement a standalone virtual memory manager, where there is a software-managed TLB. It is your responsibility to implement the code to manage TLB of your virtual memory system. Your program is responsible to
(1) load a file containing a list of logical addresses,
(2) translate logical addresses into physical addresses for a virtual address space of size 216 = 65,536 bytes,
(3) output the value of the byte stored at the translated physical address.

2. Information about the Simulated Virtual Memory

· 16-bit Logical Addresses Your program must read a file containing a list of 32-bit integer numbers, which represent 32-bit logical addresses. Please note that your program only deals with 16-bit addresses. Thus, you need to mask the rightmost 16 bits of each logical address loaded from the file.

Each 16-bit logical address is divided into two parts, namely, (1) an 8-bit page number and (2) an 8-bit page offset. The logical address structure is illustrated in Figure 1. Page number Offset 15 8 7 0 16 bits Figure 1. 16-bit Logical Address Structure.

· System Parameters of the Virtual Memory The page table size is 28 ; the TLB contains 16 entries. The page size is 28 bytes, which is the same as the frame size. There are a total of 256 frames in the physical memory, meaning that the total physical memory capability is 65,536 bytes (i.e., 256 frames * 256 bytes/frame). The system parameters of the simulated virtual

memory is summarized below. • Page table size: 28 • Number of TLB entries: 16 • Page size: 28 bytes • Frame size: 28 bytes • Number of frames: 256 • Physical memory size: 65,536 bytes.

Address Translation

Paging:

The operating system provides a virtual memory system in a way to use physical memory as a cache of virtual pages. As we mentioned in our lecture, with paging in place, all the pages in a process's virtual address space are NOT required to be in physical memory. Rather, a process can have its pages either on a disk or in memory. A page fault occurs in case that the process issues an access to a page on the disk. The operating system has to retrieve the page from the disk and bring it into the memory. Note that pages with valid TLB entries reside in physical memory. Thus, a reference to a page on disk results in a TLB fault. When a TLB fault occurs, the hardware generates a TLB exception, trapping to the operating system. The operating system then checks its own page table to locate the virtual page requested. If the page is in memory without being mapped by the TLB, then we merely need to update the TLB. However, if the page is on disk, the operating system must read the page from the disk, update the page table, and update the TLB.

Performing Address Translations:

Your simulated virtual memory system is expected to translate logical addresses into physical addresses using TLB and a page table. Given a logical address, your program takes the following three steps to perform address translation.

Step 1: The page number is extracted from the logical address.

Step 2: Access the TLB using the extracted page number.

- If there is a TLB-hit, the frame number of the page is obtained from the TLB.
- Otherwise, follow step 3 to access the page table.

Step 3: Access the page table

- The frame number of the page is obtained from the page table if the page has been loaded into the main memory;
- Otherwise, a page fault occurs.

Diagram:

Figure 2 below depicts the address-translation process, where the TLB and page table are two key data structures.

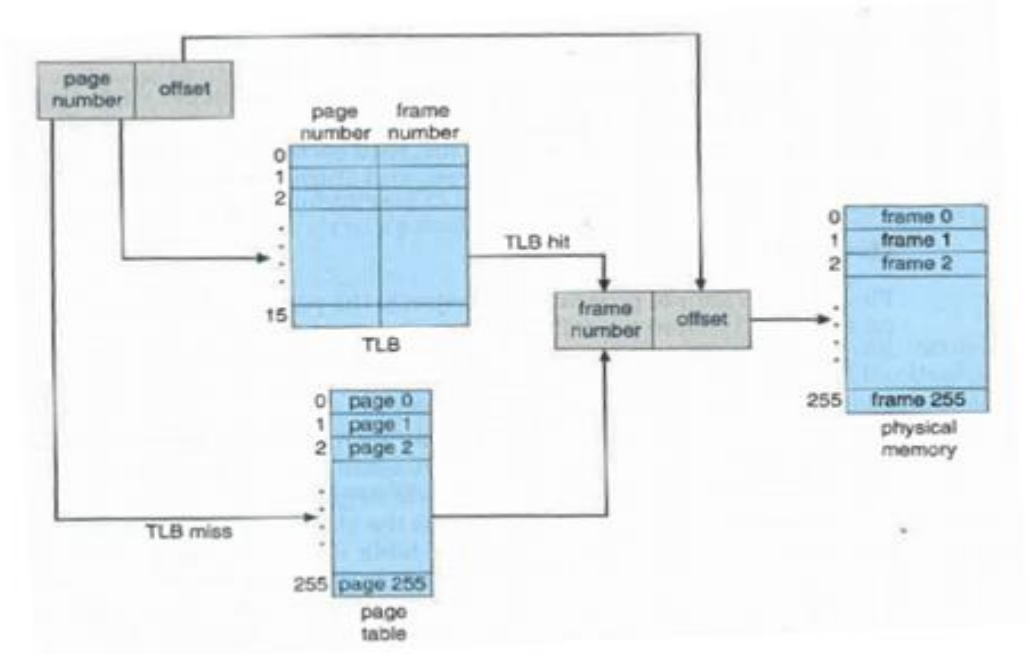


Figure 2. The address-translation process.

Algorithm:

step1: Input memory size and page size

Step2: Calculate the no. of pages

Step3: Input no. of processes and the pages required for each process

Step4: Assign the required no of pages to each process keeping in view of the remaining pages obtained after subtracting the no. of pages assigned for each consecutive process
Step5: Enter page table
Step6: Enter logical address to find physical address
Step7: Compute physical address as follows:
Physical address=frame no[x][y]*page size+offset
Step8: Print physical address
step9: Stop

8. Test cases:

The Student should check the different variations in the program by applying different Input values.

Outputs with different inputs must be given and appropriate results must be recorded after execution.

Students must also record the errors while executing the program, so that they get deeper insights on how a program should be executed.

9. Sample output:

```
Enter memory size:100
Enter page size:4
The no of pages available in memory are:25
Enter no of processes:3
Enter no of pages required for p[1]=8
Enter page table for p[1]=0 1
1 2
2 3
3 4
Enter no of pages required for p[2]=8
Enter page table for p[2]=0 1
1 2
2 3
3 4
Enter no of pages required for p[3]=8
Enter page table for p[3]=0 1
1 2
2 3
3 4
Enter logical address to find physical address
37
Enter process no, page number and offset:1 1 1 1
The physical address is : 5
```

10.Practical Related Questions:

1. Define the concept of virtual memory

Ans: Virtual memory is a feature of an operating system that enables a computer to be able to compensate shortages of physical memory by transferring pages of data from random access memory to disk storage. This process is done temporarily and is designed to work as a combination of RAM and space on the hard disk

2. What is the purpose of Page replacement

Ans: Page replacement algorithms are an important part of virtual memory management and it helps the OS to decide which memory page can be moved out, making space for the currently needed page. However, the ultimate objective of all page replacement algorithms is to reduce the number of page faults.

3. List out the various page replacement techniques

Ans: FIFO Page Replacement Algorithm, LIFO Page Replacement Algorithm, LRU Page Replacement Algorithm, Optimal Page Replacement Algorithm, Random Page Replacement Algorithm.

4. Which page replacement algorithm suffers with the problem of Belady's anamoly.

Ans: LRU page replacement algorithm suffers from Belady's anomaly . Belady's anomaly proves that it is possible to have more page faults when increasing the number of page frames while using the First in First Out (FIFO) page replacement algorithm.

5. Define the concept of Thrashing? What is the scenario that leads to the situation of Thrashing.

Ans: Thrashing occurs when a computer's virtual memory resources are overused, leading to a constant state of paging and page faults, inhibiting most application-level processing. This causes the performance of the computer to degrade or collapse.

PRACTICAL 10:

Implement the First-fit and Best-fit Algorithm in Memory Management.

1. Practical significance:

1. First Fit. In the **first fit** approach is to allocate the first free partition or hole large enough which can accommodate the process.

2. Best Fit. The **best fit** deals with allocating the smallest free partition which meets the requirement of the requesting process.

2. Relevant Program Outcomes:

3. Competency and practical skills:

The practical is expected to develop the following skills

1. Ability to work on Linux operating systems.
2. How a memory list keeps track of available memory.

4. Prerequisites:

Need to have knowledge including

§ the location of process control information,

§ the **execution** stack, and the code entry.

§ After loading of the **program** into main **memory**, the processor and the operating system must be able to translate logical addresses into physical addresses.

5. Resources required:

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.

4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.

5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/circuit/Diagram/Description:

Description:

One of the simplest methods for memory allocation is to divide memory into several fixed-sized partitions. Each partition may contain exactly one process. 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. The operating system keeps a table indicating which parts of memory are available and which are occupied. Finally, when a process arrives and needs memory, a memory section large enough for this process is provided. When it is time to load or swap a process into main memory, and if there is more than one free block of memory of sufficient size, then the operating system must decide which free block to allocate. Best-fit strategy chooses the block that is closest in size to the request. First-fit chooses the first available block that is large enough. Worst-fit chooses the largest available block

ALGORITHM: **BEST-FIT**

Step 1: Include the necessary header files required.
Step 2: Declare the variables needed.
Step 3: Read the number of blocks and the size of each blocks.
Step 4: Read the number of process and the size of each process.
Step 5: Arrange both the process and block size in an order.
Step 6: Check if the process size is less than or equal to block size.
Step 7: If yes, assign the corresponding block to the current process.
Step 8: Else print the current process is not allocated.

ALGORITHM: **FIRST-FIT**

Step 1: Include the necessary header files required.
Step 2: Declare the variables needed.
Step 3: Read the number of blocks and the size of each blocks.
Step 4: Read the number of process and the size of each process.
Step 5: Check if the process size is less than or equal to block size.
Step 6: If yes, assign the corresponding block to the current process.
Step 7: Else print the current process is not allocated.

8. Test cases:

1. The Student should Execute programs which include Functions and Control flow statements.

2. Outputs with different inputs must be given and appropriate results must be recorded after execution.

3. Students must also record the errors while executing the program, so that they get deeper insights on how a program should be written and executed.

9. Sample output:

MEMORY MANAGEMENT SCHEME - BEST FIT

Enter No. of Blocks: 5

Enter the 0st block size: 500

Enter the 1st block size: 100

Enter the 2st block size: 250

Enter the 3st block size: 650

Enter the 4st block size: 850

Enter No. of Process: 5

Enter the size of 0st process: 450

Enter the size of 1st process: 605

Enter the size of 2st process: 820

Enter the size of 3st process: 110

Enter the size of 4st process: 230

Process	Block Size
820	850
605	650
450	500
230	250
110	100

OUTPUT:

MEMORY MANAGEMENT SCHEME - FIRST FIT

Enter No. of Blocks: 5
Enter the 0st block size: 120
Enter the 1st block size: 230
Enter the 2st block size: 340
Enter the 3st block size: 450
Enter the 4st block size: 560
Enter No. of Process: 5
Enter the size of 0st process: 530
Enter the size of 1st process: 430
Enter the size of 2st process: 630
Enter the size of 3st process: 203
Enter the size of 4st process: 130

Process	Block Size
530	120
430	230
630	340
203	450
130	560

The process 3 [size 203] allocated to block 230
The process 4 [size 130] allocated to block 340
The process 1 [size 430] allocated to block 450
The process 0 [size 530] allocated to block 560
The process 2 is not allocated.

10. Practical Related Questions:

1. What is a process?

Ans: A process is an instance of a program running in a computer. It is close in meaning to task , a term used in some operating systems. ... Like a task, a process is a running program with which a particular set of data is associated so that the process can be kept track of.

2. Differentiate between first fit, best fit and worst fit.

Ans: Best fit Allocation: It is the Contiguous allocation technique in which a file is saved on the blocks whose size is same to file size or little more than file size. ... Worst Fit Allocation: This is the contiguous allocation Technique in which any available space is allocated to file which is too much large for that file.

3. How does swapping result in better memory management?

Ans: During regular intervals that are set by the operating system, processes can be copied from main memory to a backing store, and then copied back later. Swapping allows more operations to be run that can fit into memory at one time.

4. What is first fit?

Ans: First Fit. In the first fit approach is to allocate the first free partition or hole large enough which can accommodate the process. It finishes after finding the first suitable free partition.

5. What is the amount of memory required for storing the page tables of the process if a process has only 4 pages in its virtual address space?

   Ans: we have in total 4 pages and page size = 212=4KB.

   Therefore, Memory required to store page table = $4 * 4KB = 16KB$.

PRACTICAL 11:

Simulate the Contiguous file allocation method.

1. Practical significance:

A file is a collection of data, usually stored on disk. As a logical entity, a file enables to divide data into meaningful groups. As a physical entity, a file should be considered in terms of its organization. The term "file organization" refers to the way in which data is stored in a file and, consequently, the method(s) by which it can be accessed.

SEQUENTIAL FILE ALLOCATION:

In this file organization, the records of the file are stored one after another both physically and logically. That is, record with sequence number 16 is located just after the 15th record. A record of a sequential file can only be accessed by reading all the previous records.

LINKED FILE ALLOCATION:

With linked allocation, each file is a linked list of disk blocks; the disk blocks may be scattered anywhere on the disk. The directory contains a pointer to the first and last blocks of the file. Each block contains a pointer to the next block.

INDEXED FILE ALLOCATION:

Indexed file allocation strategy brings all the pointers together into one location: an index block. Each file has its own index block, which is an array of disk-block addresses. The ith entry in the index block points to the ith block of the file. The directory contains the address of the index block. To find and read the ith block, the pointer in the i th index-block entry is used.

2. Relevant Program Outcomes:

3. Competency and practical skills:

The practical is expected to develop the following skills

1. Able to understand the file allocation system.
2. Able to understand the memory management system.

4. Prerequisites:

1. A Good Understanding of file allocation system.
2. Knowledge about what is a process in the Operating System.

5. Resources required :

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)

2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.

2. Ensure the keyboard, mouse and monitor are properly working.

3. Ensure that there are no power fluctuations while executing the program.

4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.

5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/Diagram/Description:

DESCRIPTION: Contiguous Allocation

In this scheme, each file occupies a contiguous set of blocks on the disk. For example, if a file requires n blocks and is given a block b as the starting location, then the blocks assigned to the file will be: *b, b+1, b+2,……b+n-1*. This means that given the starting block address and the length of the file (in terms of blocks required), we can determine the blocks occupied by the file.

The directory entry for a file with contiguous allocation contains

- Address of starting block
- Length of the allocated portion.

The *file 'mail'* in the following figure starts from the block 19 with length = 6 blocks. Therefore, it occupies *19, 20, 21, 22, 23, 24* blocks.

Advantages:

- Both the Sequential and Direct Accesses are supported by this. For direct access, the address of the kth block of the file which starts at block b can easily be obtained as (b+k).
- This is extremely fast since the number of seeks are minimal because of contiguous allocation of file blocks.

Disadvantages:

- This method suffers from both internal and external fragmentation. This makes it inefficient in terms of memory utilization.
- Increasing file size is difficult because it depends on the availability of contiguous memory at a particular instance.

ALGORITHM:

STEP 1: Start the program.

STEP 2: Gather information about the number of files.

STEP 3: Gather the memory requirement of each file.

STEP 4: Allocate the memory to the file in a sequential manner.

STEP 5: Select any random location from the available location.

STEP 6: Check if the location that is selected is free or not.

STEP 7: If the location is allocated set the flag = 1.

STEP 8: Print the file number, length, and the block allocated.

STEP 9: Gather information if more files have to be stored.

STEP 10: If yes, then go to STEP 2.

STEP 11: If no, Stop the program.

8. Test cases:

1. The students must create a number of file systems in different memory locations and check how sequentially memory is created.

9. Sample output:

SEQUENTIAL FILE ALLOCATION

```
 #include<stdio.h>
 #include<conio.h>
struct fileTable { char name[20];
int sb, nob; }ft[30];
void main()
{
 int i, j, n;
 char s[20];
 clrscr();
 printf("Enter no of files :");
 scanf("%d",&n);
 for(i=0;i<n;i++)
{
printf("\nEnter file name %d :",i+1);
 scanf("%s",ft[i].name);
 printf("Enter starting block of file %d :",i+1);
 scanf("%d",&ft[i].sb);
 printf("Enter no of blocks in file %d :",i+1);
 scanf("%d",&ft[i].nob);
}
printf("\nEnter the file name to be searched-- ");
scanf("%s",s);
for(i=0;i<n;i++)
if(strcmp(s, ft[i].name)==0)
break;
if(i==n)
printf("\nFile Not Found");
else
{
        printf("\nFILE NAME START BLOCK NO OF BLOCKS BLOCKS OCCUPIED\n");
        printf("\n%s\t\t%d\t\t%d\t",ft[i].name,ft[i].sb,ft[i].nob);
        for(j=0;j<ft[i].nob;j++)
```

```
printf("%d, ",ft[i].sb+j);
}
getch();
}
```

INPUT:

```
Enter no of files :3
Enter file name 1 :A
Enter starting block of file 1 :85
Enter no of blocks in file 1 :6
Enter file name 2 :B
Enter starting block of file 2 :102
Enter no of blocks in file 2 :4
Enter file name 3 : C
Enter starting block of file 3 : 60
Enter no of blocks in file 3 : 4
Enter the file name to be searched :  B
```

OUTPUT:

FILE NAME	START BLOCK	NO OF BLOCKS	BLOCKS OCCUPIED
B	102	4	102, 103, 104, 105

10.Exercise Questions :

1. What is the advantage of a file allocation table (FAT) file structure over a simple linked file structure.

   Ans: The file allocation table method allows faster seeks in files.

2. What is Clustering?

   Ans: Creating large logical disk blocks from collections of contiguous physical disk blocks.

3. To read or write a block on a disk, what does a process must have?

   Ans: To make a system call.

PRACTICAL 12:

Implement bit map for the following scenario. For a memory of size 32 blocks ,the allocated blocks are 2,3,4,5,8,9,10,11,12 and display the bitmap pattern.

1. Practical significance:

· Student will understand the file system and its responsibility to allocate the free blocks to the file therefore it has to keep track of all the free blocks present in the disk. There are mainly two approaches by using which, the free blocks in the disk are managed.

· A bitmap is a mapping from one system such as integers to bits. It is also known as bitmap index or a bit array. The bitmap given in the image writes 1 for the occupied memory unit and 0 for the unoccupied memory unit. The first 5 units are occupied with A and the corresponding entry in the bitmap is 11111.

2. Relevant Program Outcomes:

3. Competency and practical skills:

The practical is expected to develop the following skills

1. Able to Understand Bit vector and free space memory management.
2. Able to understand bitmap for Dynamic partitioning.

4. Prerequisites:

· Understand the concept of Memory Management and File access methods

5. Resources required:

S.No	Name of the Resource	Broad Specification(Approx.)
1	Computer System	1. Processor – 2GHz 2. RAM – 4GB 3. Hard-Drive Space – 20GB 4. VGA with 1024×768 screen resolution (exact hardware requirement will depend upon the distribution that we choose to work with)
2	Operating System	UNIX/LINUX/UBUNTU

6. Precautions:

1. Check Whether the computer is getting proper power or not.
2. Ensure the keyboard, mouse and monitor are properly working.
3. Ensure that there are no power fluctuations while executing the commands.
4. Safe working conditions help prevent injury to people and damage to computer equipment. A safe work space is clean, organized, and properly lighted. Everyone must understand and follow safety procedures.
5. Follow electrical safety guidelines to prevent electrical fires, injuries, and fatalities in the home and the workplace. Power supplies and CRT monitors contain high voltage.

7. Algorithm/Description:

A bitmap is a mapping from one system such as integers to bits. It is also known as bitmap index or a bit array.

The memory is divided into units for bitmap. These units may range from a few bytes to several kilobytes. Each memory unit is associated with a bit in the bitmap. If the unit is occupied, the bit is 1 and if it is empty, the bit is zero.

The bitmap provides a relatively easy way to keep track of memory as the size of the bitmap is only dependent on the size of the memory and the size of the units.

The bitmap given in the image writes 1 for the occupied memory unit and 0 for the unoccupied memory unit. The first 5 units are occupied with A and the corresponding entry in the bitmap is 11111. The next three units are empty so their entry in the bitmap is 000. After that, 6 units occupy B. So there entry in the bitmap is 11111. This continues and the result obtained in the bitmap for A, B, C, D and E is shown in the image.

Key Features of Bitmap

Some important features of bitmap are −

- The unit size in bitmaps is very important and should be chosen with care.
- If the unit size is smaller, the bitmap will be larger as it will hold the value 0 or 1 for each of the units. Similarly, if the unit size is larger, bitmap will be smaller.
- The unit size need not be too large, as even with a unit size as small as 3 bytes, only one bit of the bitmap can represent 24 bits.,

Advantage of Bitmap

The bitmap is quite useful as it provides a way to keep track of the memory using only a little memory for the bitmap table. The size of the bitmap is purely dependent on the size of the memory as well as the size of the memory unit.

Disadvantage of Bitmap

A major problem in the bitmap occurs if a ‘n’ size memory block needs to be occupied by a process. Then a ‘n’ size vacancy is needed in the bitmap where the values are all zeroes.

ALGORITHM:

Step 1: Start

Step 2: Input the bit series and assign it to an array bitmap

Step 3: Input the word length

Step 4: Initialize offset to 0

Step 5: Scan through the bitmap array until first ‘1’ is encountered

Step 6: Increment the offset value for every character scan

Step 7: Initialize check and count to 0

Step 8: Increment count for every 1 in the bitmap series till offset length

Step 9: If count==word length, then increment check and assign count=0,else break

Step 10: Calculate offset as : offset=(offset % word length)+1

Step 11: Calculate block number=(word length*check)+offset

Step 12: Print block number

Step 13: Stop

8. Test cases:

1. The students must execute by providing with different word lengths

9. Sample output:

Enter bit series: 000010101010

Enter word length: 8

The first free block is available at 5

10 .Exercise Questions:

1. What is free space management in OS?

Ans: A file system is responsible to allocate the free blocks to the file therefore it has to keep track of all the free blocks present in the disk. There are mainly two approaches by using which, the free blocks in the disk are managed.

2. Which are the free space management techniques in operating system?

Ans: Bitmap or Bit vector – A Bitmap or Bit Vector is series or collection of bits where each bit corresponds to a disk block.

Linked List – In this approach, the free disk blocks are linked together i.e. a free block contains a pointer to the next free block,Grouping, Counting

3. What is Disk Space Management?

Ans: You need to manage your disk space to ensure adequate storage space for your system and all temporary partitions. Managing your disk space involves maintaining separate file systems, providing sufficient disk space, using shared devices for storage, managing temporary files.