Department of Computer Science

Gujarat University

Certificate

Roll No: **32** Seat No: _____

This is to certify that Mr./Ms. **SAMPAT PUSHTI KETANKUMAR** student of MCA Semester – III has duly completed his/her term work for the semester ending in December 2020, in the subject of **OPERATING SYSTEMS** towards partial fulfillment of his/her Degree of Masters in Computer Applications.

Date of Submission **10th-Dec-2020** Internal Faculty

Head of Department

Department Of Computer Science
Rollwala Computer Centre
Gujarat University

MCA – III

Subject: - Operating Systems

Name: - Pushti Sampat

Roll No.: - 32 **Exam Seat No.: -_____**

Sr. No.	Contents	Pg. No	Date	Signature
1.	Assignment -1		10/12/2020	
	Definitions			
	(Introduction of operating system			
	,process management,process			
	Synchronization, memory			
	Management , Storage			
	Management , File System)			
2.	Assignment-2		10/12/2020	
	Banker's algorithm			
	FIFO algorithm			
	LRU algorithm			
	FIFO algorithm			
	OPTIMAL algorithm			

Operating System

Assignment.

Defination of all Chapters

* **Base Address**

An address that is used as the origin in the calculation of address in the execution of a computer program.

*

* **Batch Processing**

Pertaining to the technique of executing a set of computer programs such that each is completed before the next program of the set is started.

* **Binary Semaphore**

A semaphore that takes on only the values 0 & 1. A binary semaphore always only one process or thread to have access to a shared critical resource at a time.

Page No.
Date: - - 20

* Block

A collection of contiguous records that one recorded as a unit, the units are seperated by interblock gaps.

A group of bits that are transmitted as a unit.

* Btree

A techinque for organizing indices in order to keep access time to a minimum.

It stores the data keys in a balanced hierarchy that continually realigns itself as items are inserted & deleted, this all nody always have a similar number of keys.

* Busy waiting.

The repeated execution of a loop of code while waiting for

can event to occur.

* Cache Memory

⇒ A memory that is smaller and faster than main memory & that is interpeted between the processor & main memory.

The cache acts as a buffer for recently used memory locations.

* Central Processing Unit (CPU)

That portion of a computer that fetches & executes instructions.

It consists of an Arithmetic & logic unit & control unit & registers often simply refferred to as a processor.

* Cluster

A group of interconnected whole

Page No.
Date: - -20

Computers working together as a unified computing resource that can create the illusion of being one machine

→ The term whole computer means a system that can run on its own, apart from cluster.

* Concurrent

Pertaining to process or threads that take place within a common interval of time during which they may have to alternatively share a common resources.

* Consumable resources

A resource that can be created & destroyed, when a resource is acquired by a process, the resource ceases to exist.

Examples of consumable resources are interrupts, signals, messages &

information in I/O buffers.

* Database

A collection of interrelated data often with controlled redundancy, organized according to a schema to serve one or more applications: the data are stored so that can be used by different programs without concern for the data structure or organization.

A common approach is used to add new data and to modify & retrieve existing data.

* Deadlock

An impass that occurs when multiple processes are waiting for the availability of a resource that will not become available because it is being held by another process that is in similar wait state.

An impass that occurs when multiple processes are waiting for an action by or a response from another process that is in a similar wait state.

* Demand Paging:

The transfer of page from secondary memory to main memory storage at the moment of need.
Compare prepaging.

* Device driver

An operating System module that deals directly with a device or I/O module.

* Direct Access.

The capability to obtain data from a storage device or to enter data into a storage device in a sequence independent of

their relative position, by means of address that indicate the physical location of the data.

* Direct Message Access (DMA)

A form of I/O which a special module, called a DMA, Module controls the exchange of data between main memory and I/O device.

The processor sends a request for the transfer of a block of data to the DMA module and is interrupted only after the entire block has transfered.

* Disable Interrupt

A condition usually created by the operating system, during which the processor will ignore interrupt request signal of a specified class.

* Disk allocation table

A table that indicates which blocks of secondary storage are free & available for allocation to files.

* Distributed Operating System.

A common operating system shared by a network of computers.

The distributed operating system provides support for interprocess communication, process migration, mutual exclusion & this prevention or detection of deadlock.

* Dynamic Relocation.

A process that assigns here absolute address to a computer program during execution so that the program may be executed from a different area of main storage.

* Enable Interrupt

A condition usually created by the operating system during which the processor will respond to interrupt request signals of a specified class.

* External fragmentation.

Occurs when memory is devided into variable size partition corresponding to the blocks of data assigned to the memory.

As segments are moved into and out of the memory, gaps will be occur between the occupied partitions of the memory.

*

* File

A set of related records treated as a unit

* field

(1) Defined logical data that are part of a record.

(2) The elementary unit of a record that may contain a data item, a data aggregate, a pointer or a link.

* [illegible]

* file allocation table (FAT)

A table that indicates the physical location on secondary storage of the space allocated to a file.

There is one file allocation table for each file.

* file Management System

A set of system software that provides services to users & applicates in the use of files including file access, directory maintanence & access control.

* file organization

The Physical order of records in a file, as determined by the access method used to store & retrieve them.

* First in first out (FIFO)

A queuing technique in which the next item to be retrieved is the item that has been in the queue for the longest time.

* First come first serve (FCFS)

Same as FIFO

*

* Hash file

A file in which records are accessed according to the values of a key field.

Hashing is used to locate a record

on the basis of its key value.

* Hashing

The selection of a storage a location for an item of data by calculating the address as a function of the contents of the Data.

This technique complicates the storage allocation function but results in a rapid random interval

* Hit ratio

In a two-level memory the fraction of all memory access that are found in the faster memory

* Indexed access

Pertaining to the organization & accessing of the records of a

Storage structure through a seperate index to the locations of a stored records.

* Indexed file

A file in which records are accessed during to the value of key fields.

An index is required that indicates the location of each record on the basis of each key value.

* Indexed sequential access

Pertaining to the organization & accessing of the records of a storage structure through an index of the keys that are stored in arbitrarity partitioned sequential files

* Indexed sequential file

A file in which records are ordered

according to the values of a key field.

The main file is supplemented with an index file that contains a partial list of key values; the index provides a lookup capability to quickly reach the vicinity of a desired record.

* Instruction Cycle

The time period during which one instruction is fetched from memory & executed when a computer is given instruction in Machine Language.

* Internal fragmentation

Occurs when memory is divided into fixed size partitions. If a block of data is assigned to one or more partitions, then there may be wasted space in the last partition.

This will occur if the last partition

of data is smaller than the last partition

* Interrupt

A suspension of a process such as the execution of computer program, caused by an event external to the process & performed in such a way that the process can be resumed.

* Interrupt handler

A notice generally part of the operating system, when an interrupt occurs, control is transferred into the corresponding interrupt handler which takes some action in response to the condition that caused the interrupt

* Job

A set of computational steps packaged to run as a unit

Page No.
Date: - -20

* Kernel:

A partition of the operating system that includes most heavily used portions of software. Generally the kernel is maintained permanently in main memory.

The Kernel runs in a priviledge mode & responds to calls from processes & interrupts from divice.

* Kernel mode:

A priviledge mode of execution reserved the Kernel of the operating system.

Typically Kernel mode allows acess to regions of main memory that are unavailable to processes executing a less priviledged mode & also enables execution of certain machine instructions that are restricted to the Kernel mode.

Also seffered as system mode or priviledge mode

* LIFO

A queuing technique in which the next item to be retrived is the item most recently placed in queue.

* Livelock

A condition in which two or more processes continuously change their state in response to change in the other process without doing any useful work.

This is similar to deadlock in that no progress is made but it differs in that neither process is blocked or waiting for anything.

* Logical Address

A reference to a memory location independent of the current assignment data to memory.

A Translation must be made to

a physical address before the memory access can be achieve.

* Logical record.

A record independent of its physical environment portions of one logical record may be located in different physical record or several logical records or several logical records or parts of logical records may be located in one physical record.

* Main Memory.

Memory that is internal to the computer system, is program addressable and can be loaded into registers for subsequent execution or processing.

* Malicious Software

Any software designed to

cause damage to or use up the resources of a target computer.

Malicious software is frequently concealed within or masquerades as legitimate software.

In some cases, it spreads itself to other computers via email or infected disks.

Types of malicious software include viruses, Trojan horses, worms & hidden software for launching denial of service attacks.

* Memory cycle time

The time it takes to read one word from or write one word to memory.

This is the inverse of the rate it which words can be read or written to memory.

* Memory Partitioning

Date: - - 20

The subdividing of storage into independent sections.

* Microkernel

A small privilege operating system core that provides process scheduling, memory management & communication services & relies on other processes to perform some of the function traditionally associated with the operating system kernel.

* Multiprocessing

A mode of operation that provides for parallel processing by two or more processors of multiprocessor.

* Multiprogramming

A mode of operations that provides for the interleaved

execution of two or more computer programs by a single processor.

The same as Multitasking using different terminology.

* Multiprogramming level

The number of processes that are partially or fully resident in main memory.

* Multitasking.

A mode of operation that provides for the concurrent performance or interleaved execution of 2 or more computer tasks.

The same as multiprogramming using different terminology.

* Mutual Exclusion

A condition in which there is a set of processes, only one of which is able to access a given resource or perform a given function at any time.

* Operating System

Software that controls the execution of program & that provides services such as resource allocation, scheduling, input/output control & data management.

* Page

In virtual storage, a fixed length block that has a virtual address & that is transferred as a unit between main memory & secondary memory.

* Page built.

Occurs when the page containing

a referenced word is not in main memory.

This causes an interrupt & requires that the proper page be brought into main memory.

* Page frame

A fixed size contiguous block of main memory used to hold a page

* Paging

The transfer of pages between main memory & secondary memory

* Physical address

The absolute location of a unit of data in memory Eg word or byte in memory block on secondary memory.

Date: - - 20

* Pipes

A circular buffer allowing 2 processes to communicate on the producer - consumer model.

Thus, it is a first in first out queue, written by one process & read by another.

In some systems the pipe is generalized to allow any item in the queue to be selected for consumption

* Preemption.

Reclaiming a resource from a process before the process has finished using it.

* Paging

The retrival of pages other than the one demanded by a page fault.

The hope is that the additional pages will be needed in the near future, conserving disk I/O. compared demand paging.

* Process:

A program in execution, a process is controlled & scheduled by a operating system.

* Process control Block (PCB)

The manifestation of proces in an operating system.

It is a data structure containing information about the characteristics & data of the process.

* Process Migration:

The Transfer of a sufficient amount of the state of a process from one machine to another for

the process to execute on the target machine.

* Process State

All the information that operating system needs to manage a process & that the processor needs to properly execute the process.

The process state includes the contents of various processor registers such as the program counter & data registers; it also includes informations of uses to the operating system, such as the priority of the process & whether the process is waiting for the completion of a particular I/O event, same as execution context.

* Program counter (PC)

Instruction address register

* Processor

In a computer, a functional unit that interprets & executed instruction.

A processor consists of at least an instruction control unit & an arithmatic unit.

* Programmed I/O.

A form of I/O in which the CPU issues an I/O command to an I/O module & must then wait for the operation to be completed before proceeding.

* Real time System.

An operating system that must schedule & manage real time things

* Real Time Tasks.

A task that is executed in connection with some process a function or set of events external to the Computer Syste & that must meet one or mor deadlines to interact effectively & correctly with the external environment.

* Registers.

High Speed memory internal to the CPU.

Some register are user visible that is available to the programmer via the machine instruction set other registers are used by the CPU for control purposes

* Relative Address.

An address calculated as a displacement from a base address.

* Remote Procedure call (RPC):

A technique by which 2 program on different machines interact using procedure call & return syntax & semantics

Both the called & calling program behave as if the partner program were running on the same machine.

* Response Time:

In a data system, the elapsed time between the end of trans-mission of an Enquiry message & the beginning of the receipt of a response messages, measured at the enquiry terminal

* Round Robin

A Scheduling algorithm in which process are activated in a fixed cyclic order; that is all

processes are in a circular queue.

A process that cannot proceed because it is waiting for an event returns control to the scheduler.

* Scheduling

To select jobs as tasks that are to be dispatched. In some operating systems, other units of work, such as input/output operations may also be scheduled.

* Secondary Memory

Memory located outside the computer system itself; that is it cannot be processed directly by the processor.

It must first be copied into main memory.

Examples include disks & tapes

Bindi
Page No.
Date: - -20

* Segment

In virtual memory a block that has a virtual address.

The blocks of a program may be unequal length & may even be of dynamically varying lengths.

* Segmentation:

The division of a program or application into segments as part of virtual memory scheme.

* Semaphore.

An integer value used for signaling among processes only three operations may be performed on a semaphore, all of which are atomic: initialize, decrement & increment.

Depending on the exact definition of the semaphore, the decrement operation may result in the

Blocking of a process & the increment operation may result in the unblocking of a process.

Also known as counting semaphore & general semaphore

* Sequential file.

A file in which records are ordered according to the values of one or more key fields & processed in the same sequence from the beginning of the file.

* Session.

A collection of one or more processes that represents single interactive user application or operating system function.

All keyboard & mouse input is directed to the foreground session is directed to the display screen.

* Shell

The portion of the operating system that interprets interactive user commands & job control language commands.

It functions as an interface between the user & the operating system

* Stack.

An ordered list in which items are appended to and deleted from the same end of the list known as top that is the next item appended to the list is put on the top & the next item to be removed from the list is the item that has been in list shortest time.

This method is characterized as last in first out.

* Starvation.

A condition in which a process is indefinitely delayed because other processes are always given performance.

* Strong Semaphore

A semaphore in which all processes waiting on the same semaphore are queued and will eventually processed in the same order as they executed the wait operations.

* Swapping

A procss that interchange the contents of an area of main storage with the contents of an area in secondary memory.

* Symmetric Multiprocessing (SMP)

A form of Multiprocessing that allows the operating system to

execute on any available processor or on several available processor simultaneously.

* Synchronous Operations.

An operation that occurs regularly or predictably with respect to the occurance of specified event in another process, for example the calling of an input/output routine that receives control at precoded location in a computer program.

* Synchronization

Situation in which two or more processes coordinate their activities based on a condition.

* System bus

A bus used to interconnect major computer components.

((PU, memory, I/O).

* Task

Same as process.

* Thrashing.

A phenomenon in virtual memory schemes, in which the processor spends most of its time swapping pieces rather than executing instructions.

* Thread.

A dispatable unit of work It includes a processor context and its own data area for stack.

A thread executes sequentially & is interruptible to that the processor can turn to another thread.

A process may contain one or more thread.

* Thread Switch.

The act of switching processor control from one thread to another within the same process

* Time sharing.

The concurrent use of a device by a number of users.

* Time slice.

The maximum amount of time that a process can execute before being interrupted.

* Time Slicing.

A model of an operation or

Date: - - 20

which two or more processes are assigned quanta of time on the same processor

* Trace

A sequence of instructions that are executed when a process is running.

* Trap

An unprogrammed conditional jump to a specified address that is automatically activated by the hardware;

the location from which the jump was made is recorded

* Trap door

Secret undocumented entry point into a program, used to grant access without normal

methods of access authentication

* Trojan horse.

Secret undocumented routine embeded within a useful program. execution of program results in execution of secret routine.

* User mode.

The least priviledged mode of execution, contain regions of main memory & certain machine instructions cannot be used in this mode

* Virtual Address

The address of a storage location in virtual memory.

* Virtual memory.

The storage space that may be regarded as addressable main storage by the user of computer system in which virtual addresses are mapped into real addresses.

The size of virtual storage is limited by the addressing scheme of the computer system and by the amount of secondary memory available & not by actual number of main storage locations.

* Virus

Secret undocumented routine embeded within useful program execution of the program results in execution of secret routine.

* weak Semaphore

A semaphore in which all processes waiting on the same semaphore processed in an unspecified order.

Assignment 2

PAGE NO :-
DATE

Bankers Algorithm.

Process	Allocation A	B	C	Max A	B	C	Available A	B	C	Need
P_0	0	1	0	7	5	3	3	3	2	
P_1	2	0	0	3	2	2				
P_2	3	0	2	9	0	2				
P_3	2	1	1	2	2	2				
P_4	0	0	2	4	3	3				

→

Process	A	B	C	A	5	3
P_0	0	1	0	3	2	
P_1	2	0	0	9	0	2
P_2	3	0	2	2		
P_3	2	1	1			
P_4	0	0	2			

Process	Allocation A	B	C	Max A	B	C	Available A	B	C	Need A	B	C
P_0	0	1	0	7	5	3	3	3	2	7	4	3
P_1	2	0	0	3	2	2				1	2	2
P_2	3	0	2	9	0	2				6	0	0
P_3	2	1	1	2	2	2				0	1	1
P_4	0	0	2	4	3	3				4	3	1

⇒ Need ≤ work ⇒ work = work + Allocation

P_0 743 ≤ 332 ← × condition fails.

P_1 122 < 332 condition true

PAGE NO:-
DATE

W = w + allocation
= 532 + 211
= 743

→ P4 Need ≤ work ⇒ w = w + alloc
431 ≤ 743 = 743 + 002
= 745

→ P0 Need ≤ work ⇒ w = work + alloc
743 ≤ 745 = 745 + 010
= 755

→ P2 Need ≤ work ⇒ w = work + alloc
600 ≤ 755 = 755 + 302
= 1057

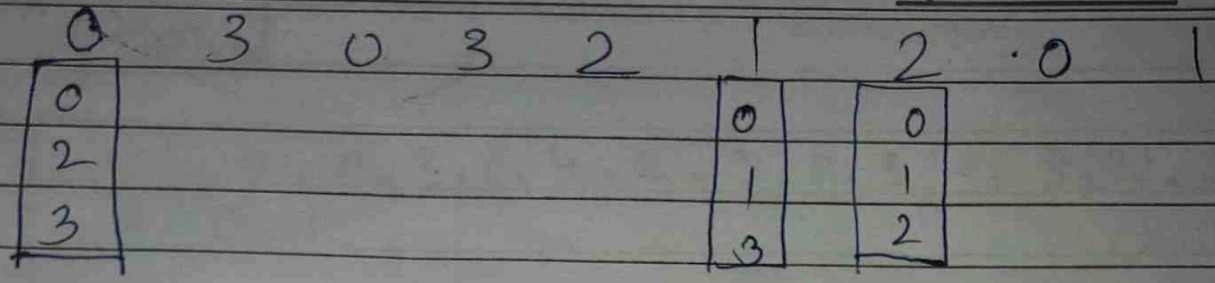
Safe sequence is → <P1, P3, P4, P0, P2>

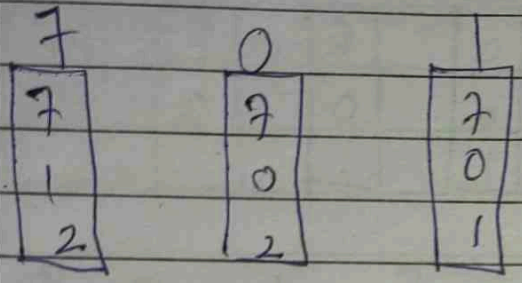
* FIFO.

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

7	0	1	2	0	3	0	4	2
7	7	7	2		2	2	4	4
	0	0	0		3	3	3	2
		1	1		1	0	0	0

PAGE NO:-
DATE



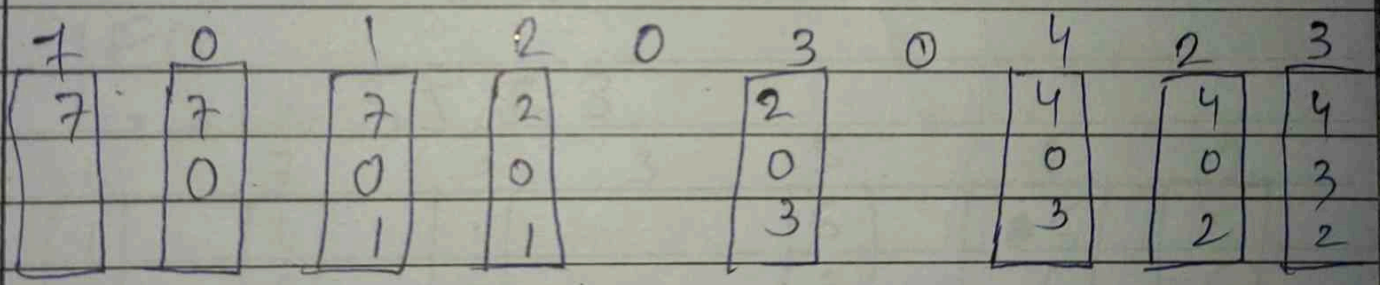


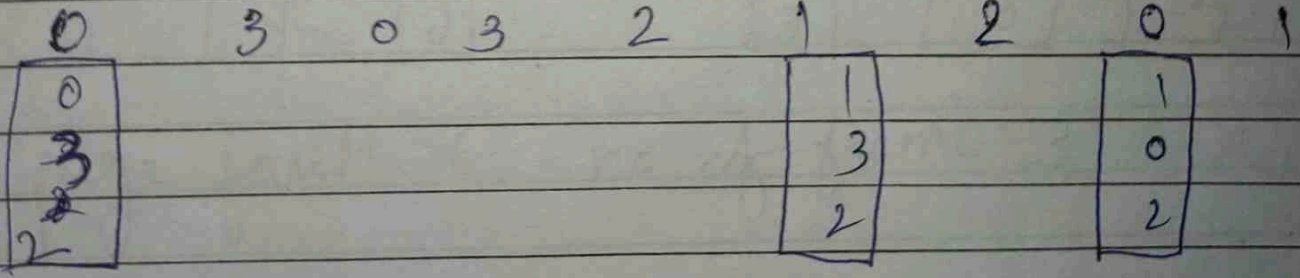
page fault = 15

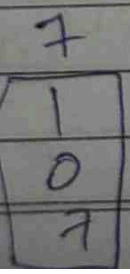
no of frames = 3

* LRU.

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1.







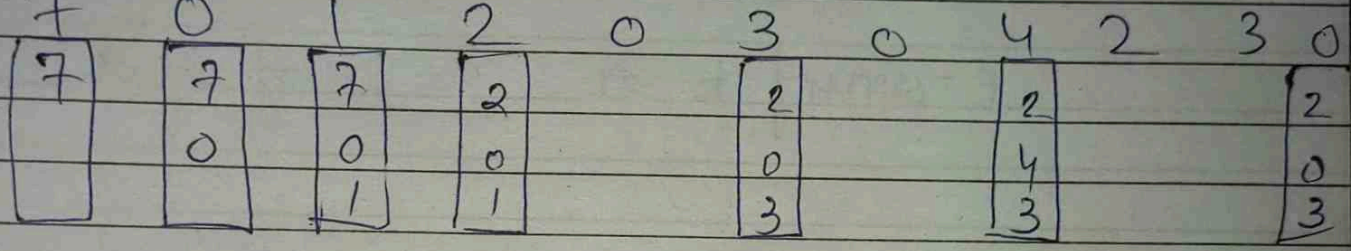
no of frames = 3

page fault = 12

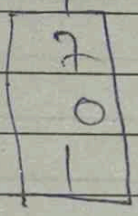
PAGE NO :-
DATE

Optimal

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1



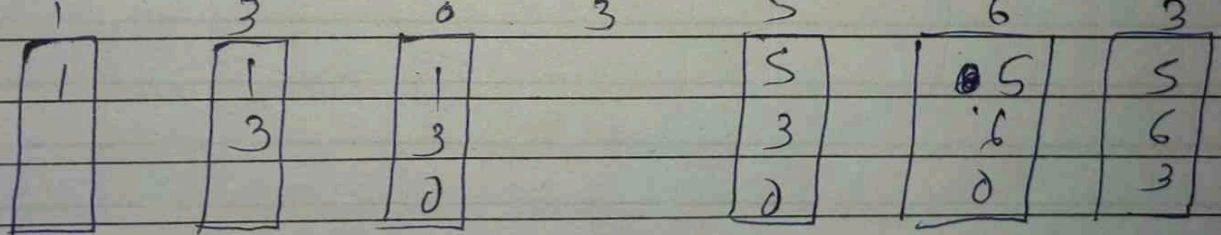
2 0 1 7 0 1



no of frames = 3
page fault = 9

⇒ FIFO

1, 3, 0, 3, 5, 6, 3.



page fault = 6 no of frame = 3

⇒ optimal

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 3

PAGE NO :-
DATE

7	0	1	2	0	3	0	4	2	3
7	7	7	7		3		3		
	0	0	0		0		0		
		1	1		1		4		
			2		2		2		

Page fault = 6 no of frames = 4.

D E P A R T M E N T O F C O M P U T E R S C I E N C E
R O L L W A L A C O M P U T E R C E N T R E
G U J A R A T U N I V E R S I T Y

M.C.A. – III

R O L L NO : 32
N A M E : PUSHTI SAMPAT
S U B J E C T : OPERATING SYSTEMS

NO.	TITLE	PAGE NO.	DATE	SIGN
1	Basic salary of a person is input through the keyboard. His dearness allowance is 40% of basic salary and house rent is 20% of basic salary. Write a program to calculate the gross pay.		10th Dec 20	
2	The distance between two cities is input through the keyboard. (in km). Write a program to convert this distance into metres, feet, inches and centimeters and display the results.		10th Dec 20	
3	The length and breadth of a rectangle and radius of a circle are entered through the keyboard, calculate the perimeter and area of rectangle and area and circumference of the circle.		10th Dec 20	
4	If a five digit number is entered through the keyboard, calculate the sum of its digits.		10th Dec 20	
5	The file /etc/passwd contains info about all users. Write a program which would receive the logname during execution, obtain information about it from the file and display the information on screen in some appropriate format.		10th Dec 20	
6	The script will receive the filename or filename with its full path, the script should obtain information about this file as given by "ls -l" and display it in proper format.		10th Dec 20	
7	If cost price and selling price of an item are entered through the keyboard, write a program to determine whether the seller has made profit or loss. Also determine how much profit/loss is made.		10th Dec 20	
8	Check whether the entered no. is odd or even.		10th Dec 20	
9	Check whether the entered no. is prime or not.		10th Dec 20	
10	Check whether the entered year is a leap year or not.		10th Dec 20	
11	The script receives two file names as arguments, the script must check whether the files are same or not, if they are similar then delete the second file		10th Dec 20	
12	Write a script which will display whether your friend has logged in or not, if he has logged in then send him some message.		10th Dec 20	
13	While executing a shell script, either the logname or uid is supplied at the command prompt, write a shell script to find out at how many terminals has this user logged in.		10th Dec 20	

D E P A R T M E N T O F C O M P U T E R S C I E N C E
R O L L W A L A C O M P U T E R C E N T R E
G U J A R A T U N I V E R S I T Y
M.C.A. – III

R O L L NO : 32
N A M E : PUSHTI SAMPAT
S U B J E C T : OPERATING SYSTEMS

NO.	TITLE	PAGE NO.	DATE	SIGN
14	Write a shell script to display the date with the format :- 25th October 2005 is a Tuesday.		10th Dec 20	
15	Write a shell script to display the appropriate message like : Good Morning / Good Afternoon / Good Evening		10th Dec 20	
16	Write a shell script to display the menu driven interface :- 1) list all files of the current directory 2) print the current directory 3) print the date 4) print the users otherwise display "Invalid Option".		10th Dec 20	
17	Create a menu driven calculator which asks for two integers and perform basic arithmetic operations.		10th Dec 20	
18	Find the factorial of any number.		10th Dec 20	
19	Display the fibonacci series upto some number		10th Dec 20	
20	Two numbers are entered through the keyboard, find the power, one number raised to another.		10th Dec 20	
21	Write a script which has the functionality similar to head and tail commands.		10th Dec 20	
22	Write a script which reports name and size of all files in a directory. whose sizes exceed 1000. The filenames should be printed in the descending order of their sizes. The total no. of files must be reported		10th Dec 20	
23	A friend of yours has promised to log in a particular time. You want to contact him as soon as he logs in, write a script which checks after every minute whether the friend has logged in or not. The logname should be supplied at command prompt.		10th Dec 20	
24	Print the prime nos. from 1 to 300.		10th Dec 20	
25	Program must display all the combinations of 1, 2, and 3.		10th Dec 20	
26	Write a script for renaming each file in the directory such that it will have the current shell PID as an extension. The shell script should ensure that the directories do not get renamed.		10th Dec 20	
27	A file called wordfile consists of several words. Write a shell script which will receive a list of filenames, the first of which would be wordfile. The shell script should report all occurences of each word in wordfile in the rest of the files supplied as arguments.		10th Dec 20	

D E P A R T M E N T O F C O M P U T E R S C I E N C E
R O L L W A L A C O M P U T E R C E N T R E
G U J A R A T U N I V E R S I T Y

M.C.A. – III

R O L L NO : 32
N A M E : PUSHTI SAMPAT
S U B J E C T : OPERATING SYSTEMS

NO.	TITLE	PAGE NO.	DATE	SIGN
28	Write a shell script which deletes all the lines containing the word "unix" in the files supplied as arguments to it.		10th Dec 20	
29	The word "unix" is present in only some of the files syupplied as arguments to the shell script. You script should search each of these files in turn and stop at the first file that it encounters containing the word unix. The filename should be displayed on the screen.		10th Dec 20	
30	A shell script receives even number of filenames. Suppose four filenames are supplied then the first file should get copied into second file, the third file should get copied into fourth and so on.. If odd number of filenames are supplied display error message.		10th Dec 20	
31	The script displays a list of all files in the current directory to which you have read, write and execute permissions.		10th Dec 20	
32	The script receives any number of filenames as arguments. It should check whether every argument supplied is a file or directory. If it is a directory it should be reported. If it is a filename then name of the file as well as the number of lines present in it should be reported.		10th Dec 20	
33	A script will receive any number of filenames as arguments. It should check whether such files already exist. If they do, then it should be reported, if not then check if a subdirectory "mydir" exists or not in the current directory, if it doesnt exist then it should be created and in it the files supplied as arguments should be created.		10th Dec 20	
34	Accept the marks of 5 subjects and calculate the percentage and grade. 35. Print armstrog nos. from 1 to 500.		10th Dec 20	
35	Print armstrog nos. from 1 to 500.		10th Dec 20	
36	Accept the measure (angles) of a triangle and display the type of triangle. (eg. acute, right, obtuse)		10th Dec 20	
37	Display all the numbers from 1 to 100 which are divisible by 7		10th Dec 20	
38	Find the largest and smallest of 3 different numbers.		10th Dec 20	
39	Find HCF and LCM of a given no.		10th Dec 20	
40	Display the dates falling on Sundays of the current month.		10th Dec 20	

****
****

****
****

Name: Pushti Sampat

Class: MCA III

Rollno: 32

Subject: Operating System - OS

****
****

****
****

Q)

Basic salary of a person is input through the keyboard. His dearness allowance is 40% of basic salary and

house rent is 20% of basic salary. Write a program to calculate the gross pay.

****
****

****
****

```
echo "Enter Salary : "

read salary

da=$(($salary * 40  / 100))

h_r=$(($salary * 20 / 100))
```

```
gross_pay=`expr $salary + $da + $h_r`

echo "Gross Pay Salary  = " $gross_pay
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a1.sh
Enter Salary :
10000
Gross Pay Salary  =  16000
```

****

****

Q)

The distance between two cities is input through the keyboard(in km). Write a program to convert this

distance into metres, feet,inches and centimeters and display the results.

****

****

```
echo "Enter Distance between 2 cities :"
read dist
meter=`expr $dist \* 1000`
```

```
echo "Distance in meter = "$meter

feet=`echo "scale = 2;$dist * 3280.84"| bc`
echo "Distance in Feet = "$feet

inch=`echo "scale = 2;$dist *39370.08"| bc`
echo "Distance in Inches = "$inch

cm=`expr $dist \* 100`
echo "Distance in Centimeter = "$cm
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a2.sh
Enter Distance between 2 cities :
125
Distance in meter = 125000
Distance in Feet = 410105.00
Distance in Inches = 4921260.00
Distance in Centimeter = 12500
```

****

****

Q)

The length and breadth of a rectangle and radius of a circle are entered through the keyboard, calculate

the perimeter and area of rectangle and area and circumference of the circle.

****
****

****
****

```
echo "Enter length for rectangle :"
read length
echo "Enter breadth for rectangle :"
read breadth

area=`echo "scale = 2;$length * $breadth" | bc`
echo "Area of Rectangle = "$area
peri=`echo "scale = 2;$area * 2" | bc`
echo "Perimeter of rectangle = "$peri
echo "Enter radius for circle :"
read radius

area_c=`echo "scale = 2;3.14 * $radius * $radius" | bc`
echo "Area of circle = "$area_c

cirm=`echo "scale = 2; 2 * 3.14 * $radius" | bc`
echo "Circumference  of the circle = "$cirm
```

****
****

****
****

Output:

pushti@DESKTOP-A0UDC7F:~$ sh a3.sh

Enter length for rectangle :

2

Enter breadth for rectangle :

3

Area of Rectangle = 6

Perimeter of rectangle = 12

Enter radius for circle :

3

Area of circle = 28.26

Circumference of the circle = 18.84

****

****

Q)

If a five digit number is entered through the keyboard, calculate the sum of its digits.

****

****

```
echo "Enter Five digit Number : "
read no
num=$no
sum=0
while [ "$no" -gt 0 ]
do
    rem=$(($no % 10))
    sum=$(($sum + $rem))
```

```
    no=$(($no / 10))
done
echo "Sum of "$num" of digits = "$sum
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a4.sh
Enter Five digit Number :
12345
Sum of 12345 of digits = 15
```

****

****

Q)

The file /etc/passwd contains info about all users. Write a program which would receive the logname during

execution, obtain information about it from the file and display the information on screen in some

appropriate format. (Hint : use cut)

eg. Logname : UID : GID : Default working directory : Default working shell

****

****

```
cut -f 1,3,4,6,7 -d":" /etc/passwd | tail -n 1
```

****
****

****
****

Output:

pushti@DESKTOP-A0UDC7F:~$ sh a5.sh

pushti:1000:1000:/home/pushti:/bin/bash

****
****

****
****

Q)

The script will receive the filename or filename with its full path, the script should obtain information about

this file as given by "ls -l" and display it in proper format.

eg. Filename : File access permissions : Number of links : Owner of the file : Group to which belongs :

Size of file : File modification date : File modification time

****
****

****
****

```
echo "Enter File Name : "
read fname

ls -l $fname | cut -d ' ' -f 1,2,3,4,5,6,7,8,9
```

****
****

****
****

Output:

pushti@DESKTOP-A0UDC7F:~$ sh a6.sh

Enter File Name :

a1.sh

-rw-r--r-- 1 pushti pushti 169 Nov 20 22:46 a1.sh

****
****

****
****

Q)

If cost price and selling price of an item are entered through the keyboard, write a program to determine

whether the seller has made profit or loss. Also determine how much profit/loss is made.

****
****

****
****

```
echo "Enter Cost price : "
read cp
echo "Enter Selling price : "
read sp

if [ $sp -gt $cp ]
then
    echo "The Seller Made profit of "`expr $sp - $cp`
else
```

```
    echo "The seller made loss of "`expr $cp - $sp`
fi
```

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a7.sh
Enter Cost price :
100
Enter Selling price :
120
The Seller Made profit of 20
```

****

Q)

Check whether the entered no. is odd or even.

****

```
echo "Enter Number : "
read no

rem=`expr $no % 2`
if [ $rem -eq 0 ]
```

```
then
    echo "$no is even number.."
else
    echo "$no is odd number.."
fi
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a8.sh
Enter Number :
12
12 is even number..
```

****

****

Q)

Check whether the entered no. is prime or not.

****

****

```
echo "Enter Number : "
read no
```

```
flag=1
i=2

while [ $i -lt $no ]
do
	rem=`expr $no % $i`
	#echo "$rem is..."
	if [ $rem -eq  0 ]
	then
		flag=0
		break
	fi
	i=`expr $i + 1`
done

if [ $flag -eq 1 ]
then
	echo "$no is prime Number."
else
	echo "$no is not prime Number."
fi
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a9.sh
Enter Number :
6
6 is not prime Number.
```

****

****

Q)

Check whether the entered year is a leap year or not.

****

****

```
echo "Enter Year : "
read yr

if [ `expr $yr % 4` -eq 0 -a `expr $yr % 100` -ne 0 -o `expr $yr % 400` -eq 0 ]
then
    echo "$yr is leap year."
else
    echo "$yr is not leap year."
fi
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a10.sh
Enter Year :
```

2020

2020 is leap year.

****

****

Q)

The script receives two file names as arguments, the script must check whether the files are same or not, if

they are similar then delete the second file.

****

****

`cmp -s $1 $2`

```
if [ $? -eq 0 ]
then
    echo "$1 and $2 are same files"
    rm $2
    echo "$2 file deleted"

else
    echo "$1 and $2 are not same files"
fi
```

****

****

Output:

pushti@DESKTOP-A0UDC7F:~$ sh a11.sh demo2.txt demotest.txt

demo2.txt and demotest.txt are not same files

****

****

Q)

Write a script which will display whether your friend has logged in or not, if he has logged in then send him

some message.

****

****

```
echo "Enter User Name :"
read uname
who | grep $uname > /dev/null

if [ $? -eq 0 ]
then
    echo "User is Logged in.."
    echo "Please enter message :"
    read msg
    echo $msg
else
    echo "User is not logged in .."
fi
```

****
****

****
****

Output:

[ec2-user@ip-172-31-93-145 ~]$ sh a12.sh

Enter User Name :

pushti

User is Logged in..

Please enter message :

hii

hii

****
****

****
****

Q)

While executing a shell script, either the logname or uid is supplied at the command prompt, write a shell

script to find out at how many terminals has this user logged in.

****
****

****
****

```
if [ $# -eq 1 ]
then
        total=`who | grep -c $1`
        echo "$1 logged in on total $total terminals"
```

```
else
        echo "please enter user name..."
fi
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a13.sh
please enter user name...
pushti@DESKTOP-A0UDC7F:~$ sh a13.sh pushti
pushti logged in on total 1 terminals
```

****

****

Q)

Write a shell script to display the date with the format :- 25th October 2005 is a Tuesday.

****

****

```
date +"%dth %B %Y is a %A"
```

****

****

Output:

pushti@DESKTOP-A0UDC7F:~$ sh a14.sh

08th December 2020 is a Tuesday

****

****

Q)

Write a shell script to display the appropriate message like : Good Morning / Good Afternoon / Good Evening

****

****

```
time=`date '+%H'`
echo "$time"

if [ $time -ge 6 -a $time -lt 12 ]
then
    echo "Good Morning.."
elif [ $time -ge 12 -a $time -lt 16 ]
then
    echo "Good Afternoon.."
elif [ $time -ge 16 -a $time -lt 20 ]
then
    echo "Good Evening.."
else
    echo "Good Night.."
fi
```

****

****

****
****

Output:

pushti@DESKTOP-A0UDC7F:~$ sh a15.sh

16

Good Evening..

****
****

****
****

Q)

Write a shell script to display the menu driven interface :- 1) list all files of the current directory, 2) print the

current directory, 3) print the date, 4) print the users otherwise display "Invalid Option".

****
****

****
****

```
while [ 1 ]
do
    echo -e "\n\n1.List of all the current directory\n2.Print the current directory\n3.print the date
        \n4.print the users\n0.exit"

    echo "Enter your  choice :"
    read ch

    case $ch in
        "1") ls ;;
        "2") pwd ;;
```

```
        "3") date ;;
        "4") who ;; # awk -F: '{print $1}' /etc/passwd
        "0") echo "Exit"
            break ;;
         *) echo "Invalid choice.."
    esac
done
```

****

****

Output:

```
[ec2-user@ip-172-31-93-145 ~]$ sh a16.sh


1.List of all the current directory
2.Print the current directory
3.print the date
4.print the users
0.exit
Enter your  choice :1
a16.sh  emp.dat  first.sh  mca06  sample.dat  sample.sh  second.sh  stud.dat  third.sh


1.List of all the current directory
2.Print the current directory
3.print the date
4.print the users
0.exit
```

Enter your choice :2

/home/ec2-user

1.List of all the current directory

2.Print the current directory

3.print the date

4.print the users

0.exit

Enter your choice :3

Sun Nov 22 08:12:42 UTC 2020

1.List of all the current directory

2.Print the current directory

3.print the date

4.print the users

0.exit

Enter your choice :4

ec2-user pts/0   2020-11-22 07:55 (43.241.144.141)

1.List of all the current directory

2.Print the current directory

3.print the date

4.print the users

0.exit

Enter your choice :5

Invalid choice..

1.List of all the current directory

2.Print the current directory

3.print the date

4.print the users

0.exit

Enter your choice :0

Exit

****
****

****
****

Q)

Create a menu driven calculator which asks for two integers and perform basic arithmetic operations.

****
****

****
****

```
echo "Enter !st Integer :"
read no1
echo "Enter 2nd Integer :"
read no2

while [ 1 ]
do
    echo  "\n1.Addition\n2.Subtraction\n3.Multiplication\n4.Division\n0.Exit\n"
    echo "Enter Your Choice :"
    read ch
```

```
        case $ch in
                "1") ans=`expr $no1 + $no2`
                        echo "Addition = " $ans ;;
                "2") ans=`expr $no1 - $no2`
                        echo "Subtraction = " $ans ;;
                "3") ans=`expr $no1 \* $no2`
                        echo "Multiplication = " $ans ;;
                "4") ans=`expr $no1 \/ $no2`
                        echo "Division = " $ans ;;
                "0") echo "Exit.."
                        break ;;
                *) echo "Invalid Choice.."
        esac
done
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a17.sh
Enter !st Integer :
40
Enter 2nd Integer :
20

1.Addition
2.Subtraction
3.Multiplication
4.Division
0.Exit
```

Enter Your Choice :

1

Addition = 60

1.Addition

2.Subtraction

3.Multiplication

4.Division

0.Exit

Enter Your Choice :

3

Multiplication = 800

1.Addition

2.Subtraction

3.Multiplication

4.Division

0.Exit

Enter Your Choice :

4

Division = 2

1.Addition

2.Subtraction

3.Multiplication

4.Division

0.Exit

Enter Your Choice :

2

Subtraction = 20

1.Addition

2.Subtraction

3.Multiplication

4.Division

0.Exit

Enter Your Choice :

u

Invalid Choice..

1.Addition

2.Subtraction

3.Multiplication

4.Division

0.Exit

Enter Your Choice :

5

Invalid Choice..

1.Addition

2.Subtraction

3.Multiplication

4.Division

0.Exit

Enter Your Choice :

0

Exit..

****

****

Q)

Find the factorial of any number.

****

****

```
echo "Enter Number : "
read no

fact=1

while [ $no -gt 1 ]
do
    fact=$(($fact * $no))
    no=$(($no - 1))
done
echo "Factorial = $fact"
```

or

```
echo "Enter Number : "
read no
```

```
num=$no
fact=1

while [ $no -gt 1 ]
do
    fact=`expr $fact \* $no`
    no=$(($no - 1))
    #echo "$no+1! is $fact.."
done
echo "Factorial = $fact"
```

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a18.sh
Enter Number :
5
Factorial = 120
```

****

Q)

Display the fibonacci series upto some number.

****

```
********************************************************************************
*****************************************
echo "Enter number :"
read num

n1=0
n2=1
i=2

echo "Fibonacci Series : "
echo "$n1\n$n2"

while [ $i -lt $num ]
do
    i=`expr $i + 1`
    n3=`expr $n1 + $n2`
    echo $n3
    n1=$n2
    n2=$n3
done

********************************************************************************
*****************************************
********************************************************************************
*****************************************
```

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a19.sh
Enter number :
5
Fibonacci Series :
```

0

1

1

2

3

****

****

Q)

Two numbers are entered through the keyboard, find the power, one number raised to another.

****

****

```
echo "Enter Power : "
read pow
echo "Enter Exponent : "
read exp

i=0
ans=1

while [ $i -lt $pow ]
do
    ans=`expr $ans \* $exp`
    i=`expr $i + 1`
done
```

```
echo "$exp ^ $pow = $ans"
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a20.sh
Enter Power :
2
Enter Exponent :
5
5 ^ 2 = 25
```

****

****

Q)

Write a script which reports name and size of all files in a directory. whose sizes exceed 1000. The filenames should be printed in the descending order of their sizes. The total no. of files must be reported.

****

****

```
ls --sort=size -l | awk '$5 >= 1000 {print $5,$9}'
```

****
****

Output:

pushti@DESKTOP-A0UDC7F:~$ sh a22.sh

4096 mydir

1496 a16.sh

1392 a17.sh

1059 MCA3_32_Pushti.sh

****
****

****
****

Q)

Print the prime nos. from 1 to 300.

****
****

****
****

```
checkPrime () {
    n=$1
    if [ $n -le 1 ]
    then
        return 0
    fi

    if [ $n -le 3 ]
    then
        return 1
```

```
    fi

    if [ $(($n % 2)) -eq 0 ]
    then
        return 0
    fi

    if [ $(($n % 3)) -eq 0 ]
    then
        return 0
    fi
    i=5
    while [ $(($i*$i)) -le $n ]
    do
        if [ $(($n % $i)) -eq 0 ]
        then
            return 0
        fi
        if [ $(($n % ($i+2))) -eq 0 ]
        then
            return 0
        fi
        i=$(($i+6))
    done
    return 1
}

num=2
while [ $num -le 300 ]
do
    checkPrime $num
```

```
    isPrime=$?

    if [ $isPrime -eq 1 ]
    then
        echo "$num "
    fi

    num=$((num+1))
done
```

****

****

```
pushti@LAPTOP-8T1JFM8S:~$ sh 24.sh
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
```

59

61

67

71

73

79

83

89

97

101

103

107

109

113

127

131

137

139

149

151

157

163

167

173

179

181

191

193

197

199

211

223

227

229

233

239

241

251

257

263

269

271

277

281

283

293

****
****

****
****

Q)

Program must display all the combinations of 1, 2, and 3.

****
****

****
****

for i in 1 2 3

```
do
for j in 1 2 3
do
for k in 1 2 3
do
if [ $k -le $j -a $j -le $i ]
 then
echo $i $j $k
fi
done
done
done
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a25.sh
1 1 1
2 1 1
2 2 1
2 2 2
3 1 1
3 2 1
3 2 2
3 3 1
```

3 3 2

3 3 3

****
****

****
****

Q)

Write a script for renaming each file in the directory such that it will have

the current shell PID as an extension. The shell script should ensure that

the directories do not get renamed.

****
****

****
****

```
for f in *
do
   [ -e $f ] || continue
      mv $f $f.$$
done
```

****
****

****
****

```
pushti@DESKTOP-A0UDC7F:~$ ls -l
total 0
-rwxrwxrwx 1 pushti pushti 13 Dec  9 16:38 Hello.sh.82
-rwxrwxrwx 1 pushti pushti 58 Dec  9 16:38 file1.txt.82
```

-rwxrwxrwx 1 pushti pushti 0 Dec 9 14:42 pushti.sh.82

-rwxrwxrwx 1 pushti pushti 0 Dec 9 16:43 s_s_26.sh

-rwxrwxrwx 1 pushti pushti 55 Dec 9 16:39 s_s_26.sh.82

****
****

*

****
****

Q)

A file called wordfile consists of several words. Write a shell script which
will receive a list of filenames, the first of which would be wordfile.
The shell script should report all occurences of each word in
wordfile in the rest of the files supplied as arguments.

****
****

****
****

```
if [ $# -eq 0 ]; then
    printf "Usage:\n"
    echo "./27-findWordFromFile.sh <wordFile> <findFile ...>"
    exit
fi

filesToRead=$(($#-1))
echo $filesToRead

# Reading Line by Line
while read line; do
```

```
# Reading Word by Word
    for word in $line; do
        echo "Searching word: '$word' ..."
        # 2 is slice starting index
        # filesToRead is slice length
        grep --color=always -n $word ${@:2:filesToRead}
        printf "Done.\n\n"
    done
done <"$1" # $1 is the file name we want to search
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a27.sh wordFile.txt findFile.txt
1
Searching word: 'samsung' ...
```

****

****

Write a shell script which deletes all the lines containing the word "unix"

in the files supplied as arguments to it.

****

****

```
word="unix"

# Read all args
for i; do
```

```
    # I is for Insensitive
    # d is for delete
    # I must be written first
    sed -i "/\b$word\b/Id" $i
done
echo "Lines Contain word unix are Deleted successfully"
```

****

****

Output:

```
pushti@LAPTOP-8T1JFM8S:~$ cat wordfile.txt
hi i am unix
unixunixuni
unixccc
oh hi

pushti@LAPTOP-8T1JFM8S:~$ sh 28.sh wordfile.txt
Lines Contain word unix are Deleted successfully

pushti@LAPTOP-8T1JFM8S:~$ cat wordfile.txt
unixunixuni
unixccc
oh hi
```

****

****

****
****

Q)

The word "unix" is present in only some of the files supplied as arguments

to the shell script. You script should search each of these files in turn

and stop at the first file that it encounters containing the word unix.

The filename should be displayed on the screen.

****
****

****
****

```
for i
do
    echo "Searching file : $i..."

    if grep -q "unix" "$i";then
        echo "Found in $i"
        exit
    fi
    echo "done"
done
```

****
****

****
****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a29.sh temp.txt
Searching file : temp.txt...
```

Found in temp.txt

pushti@DESKTOP-A0UDC7F:~$ cat temp.txt

hello

unix

how r u

unix

****

****

Q)

A shell script receives even number of filenames. Suppose four filenames are supplied then the first file should get copied into second file, the third file should get copied into fourth and so on.. If odd number of filenames are supplied display error message

****

****

```
if [ $# -eq 0 ]
then
    echo "No arguments"
    exit
fi

prevFile=$1

#if even no of args
```

```
if [ $(echo $# % 2 | bc) -eq 0 ]
then
	#looping through each argument
	count=1
	for i
	do
		if !(($count%2))
		then
			cp $prevFile $i
			echo "'$prevFile' copied to -> $i"
		else
			prevFile=$i
		fi
		count=$(echo $count+1 | bc)
	done
#if odd no of args
else
	echo "Odd no of arguments"
	exit
fi
```

****

```
pushti@DESKTOP-A0UDC7F:~$ cat temp.txt
hello
unix
how r u
unix
```

pushti@DESKTOP-A0UDC7F:~$ cat newdemo.txt

pushtisampat

fgdffgef

fdgfdvd

pushti@DESKTOP-A0UDC7F:~$ cat demo.txt

pushti/sampat

fgdf/fgef

fdgf/dvd

pushti@DESKTOP-A0UDC7F:~$ cat demotest.txt

pushti/sampat

fgdf/fgef

fdgf/dvd

****
****

****
****

Q)

The script displays a list of all files in the current directory to which

you have read, write and execute permissions.

****
****

****
****

```
echo "The list of File Names in the current Directory"
echo "Which have read,write and execute permission"
```

```
for file in *
do
    if [ -f $file ]
    then
        if [ -r $file -a -w $file -a -x $file ]
        then
            ls $file
        fi
    fi
done
```

****
****
**

****
****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a31.sh
The list of File Names in the current Directory
Which have read,write and execute permission
a1.sh
```

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

Q)

The script receives any number of filenames as arguments. It should check whether every argument supplied is a file or directory. If it is a

directory it should be reported. If it is a filename then name of the

file as well as the number of lines present in it should be reported.

****

****

```
for i; do
    if [ -d $i ]; then
        echo "$i -> directory"
    elif [ -f $i ]; then
        printf "$i -> file with lines: "
        wc -l $i | awk {'print $1'}
    else
        echo "$i -> Invalid"
    fi
done
```

****

****

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a32.sh demo.txt
demo.txt -> file with lines: 3
```

****

****

****
****

Q)

A script will receive any number of filenames as arguments. It should check whether such files already exist. If they do, then it should be reported, if not then check if a subdirectory "mydir" exists or not in the current directory, if it doesnt exist then it should be created and in it the files supplied as arguments should be created.

****
****

****
****

```
if [ $# -eq 0 ]; then
        echo "No Arguments passed"
        exit
fi

for i; do
# If file exists
      if [ -f $i ]; then
            echo "$i exists"

      else
      # if "mkdir" exists
            if [ -d "mydir" ]; then
            # Directory exists
            printf "directory exists.."
            else
                  mkdir mydir
```

```
        fi
        touch mydir/$i
        echo "$i file created in \"mydir\""
    fi
done
```

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a33.sh pushti
directory exists..pushti file created in "mydir"
```

****

Q)

Accept the marks of 5 subjects and calculate the percentage and grade.

****

```
echo "Enter Marks for subject 1 : "
read s1
echo "Enter Marks for Subject 2 : "
read s2
echo "Enter Marks for Subject 3 : "
read s3
echo "Enter Marks for Subject 4 : "
```

```
read s4
echo "Enter Marks for Subject 5 : "
read s5

total=`expr $s1 + $s2 + $s3 + $s4 + $s5`
echo "Total = $total"

per=$(($total * 100 / 500 ))
echo "Percentage = $per"

if [ $per -gt 80 ]
then
	echo "Grade is A.."
elif [ $per -lt 80 -a $per -gt 60 ]
then
	echo "Grade is B.."
elif [ $per -lt 60 -a $per -gt 40 ]
then
	echo "Grade is c.."
else
	echo "Fail.."
fi
```

****

**

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a34.sh
Enter Marks for subject 1 :
80
Enter Marks for Subject 2 :
89
Enter Marks for Subject 3 :
87
Enter Marks for Subject 4 :
78
Enter Marks for Subject 5 :
80
Total = 414
Percentage = 82
Grade is A.
```

****

****

Q)

Print armstrog nos. from 1 to 500.

****

****

```
i=1
while [ $i -lt 500 ]
do
    j=$i
    total=0
    while [ $j -gt 0 ]
    do
```

```
        temp=$(echo $j%10 | bc)
        sum=$(echo $temp^3 | bc)
        total=$(echo $total+$sum | bc)
        j=$(echo $j/10 | bc)
    done
    if [ $total -eq $i ]
    then
        echo "Armstrong number : " $i
    fi
    i=`expr $i + 1`
done
```

****

***

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a35.sh
Armstrong number :  1
Armstrong number :  153
Armstrong number :  370
Armstrong number :  371
Armstrong number :  40
```

****

****

Q)

Accept the measure (angles) of a triangle and displa the type of triangle.

(eg. acute, right,obtuse)

****

****

```
echo "Enter angle "
read angle

if [ $angle -ge 0 -a $angle -lt 90 ]
then
    echo "Acute angle"
elif [ $angle -eq 90 ]
then
    echo "Right angle "
elif [ $angle -gt 90 -a $angle -le 180 ]
then
    echo "Obtuse angle "
else
    echo "Incorrect input"
fi
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a36.sh
Enter angle
120
```

Obtuse angle

****
****

****
****

Q)

Display all the numbers from 1 to 100 which are divisible by 7.

****
****

```
checkDivisible(){
        n=$1
        if [ $((n % 7)) -eq 0 ]; then
            return 1
        fi
            return 0

}

num=1

while [ $num -le 100 ]
do
      checkDivisible $num
      isDivisible=$?

      if [ $isDivisible -eq 1 ]
      then
            printf "$num "
      fi
```

```
        num=$((num+1))
done
```

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a37.sh
7 14 21 28 35 42 49 56 63 70 77 84 91 98
```

****

****

Q)

Find the largest and smallest of 3 different numbers.

****

****

```
echo "Enter 1st Number :"
read no1
echo "Enter 2nd Number :"
read no2
echo "Enter 3rd Number :"
read no3

if [ $no1 -gt $no2 -a $no1 -gt $no3 ]
```

```
then
    echo "$no1 is Largest.."
elif [ $no2 -gt $no1 -a $no2 -gt $no3 ]
then
    echo "$no2 is Largest.."
else
    echo "$no3 is Largest.."
fi


if [ $no1 -lt $no2 -a $no1 -lt $no3 ]
then
    echo "$no1 is Smallest.."
elif [ $no2 -lt $no1 -a $no2 -lt $no3 ]
then
    echo "$no2 is Smallest.."
else
    echo "$no3 is Smallest..."
fi
```

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a38.sh
Enter 1st Number :
3
Enter 2nd Number :
```

1

Enter 3rd Number :

2

3 is Largest..

1 is Smallest..

****
****

**

****
****

Q)

Find HCF and LCM of a given no.

****
****

****
****

```
echo -n "Enter first number : "
read num1
echo -n "Enter second number : "
read num2

max=$num1
den=$num2

if [ $num2 -gt $max ]
then
    max=$num2
    den=$num1
fi
```

```
    rem=$((max % den))

while [ $rem -ne 0 ]
do
    max=$den
    den=$rem
    rem=$((max % den))
    max=$((max - 1))
done

    gcd=$den
    lcm=`expr $num1 \* $num2 / $gcd`

echo "HCF of $num1 and $num2 = $gcd"
echo "LCM of $num1 and $num2 = $lcm"
```

****

****

****

****

Output:

```
pushti@DESKTOP-A0UDC7F:~$ sh a39.sh
Enter first number : 15
Enter second number : 55
HCF of 15 and 55 = 5
LCM of 15 and 55 = 165
```

****
****

*

****
****

Q)

Display the dates falling on Sundays of the current month.

****
****

****
****

```
echo "Sundays in current month are:"


echo " ----- Using AWK ----- "
cal | awk 'FNR > 2{print $1}'



#Sundays in current month are:
 #----- Using AWK -----
 #1
 #6
 #13
 #20
 #27
```

****
****

****
****

Output:

Sundays in current month are:

----- Using AWK -----

1

6

13

20

27

****
****

****
****