Department of Computer Science

Gujarat University

Certificate

Roll No: **09** Seat No: _____

This is to certify that Mr./Ms. **NEEL S. GOLORANA** 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 System

Name: - Neel s. Golarana

Roll No.: - 09 **Exam Seat No.: -_____**

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

O. S.

Assignment:

K. Definition of the 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 and 1. A binary semaphore always only one process or thread to have access to a shared critical resource at a time

* Block:

A collection of contiguous records

Scanned by TapScanner

that are recorded as a unit; the units are seperated by interblock gaps.

(2) A group of bits that are transmitted as a unit.

* B-tree:

A technique for organizing indexes. 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 and deleted, thus all nodes always have a similar number of keys.

* busy-waiting:

The repeated execution of a loop of code while waiting for an event to occure.

* cache memory:

A memory that is smaller and faster than main memory and that is interpreted between the processor and main memory, The cache acts as a buffer for recently used memory locations.

Scanned by TapScanner

Central processing Unit (CPU):

That portion of a computer that fetches and executes instructions. It consists of an Arithmetic and logic unit (ALU), a control unit, and registers. Often simply referred to as a processor.

Cluster:

A group of interconnected whole 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 the cluster.

Concurrent:

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

Consumable resources:

A resource that can be created and destroyed, when a resource is acquired by a process, the resource

cases to exist. Examples of consumable resources are interrupts, signals, messages, and 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 they 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 and retrive existing data

* Deadlock:

(1) 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.

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

Scanned by TapScanner

Page
Date

* 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 driver 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 Memory 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

Scanned by TapScanner

only after the entire block has been transbered.

☆ 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 indecates which blocks of secondary storage are free and 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~~ exclusion and the prevention or detection of deadlock.

Scanned by TapScanner

Page
Date

Dynamic Relocation:

A process that assigns new absolute address to a computer program during execution so that the program may be executed from a dibberent area of memory.

Enable interrupt:

A condition usually created by the operating system during which the processor will respond to interupt 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 occure between the occupied partitions of the memory.

~~Dictd~~: File:

A set of related records ~~that~~ treated as a unit.

Scanned by TapScanner

✱ 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.

✱ 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 and applications for the use of their files including file access, directory maintenance and access control.

✱ File Organization:

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

Scanned by TapScanner

* First in first out (FIFO),

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

* First come first Served : (FCFS)

Same as FIFO.

* Flash 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 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.

Scanned by TapScanner

Page
Date

☆ 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 and 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 and accessing of the records of a storage structure through an index of the keys that are stored in arbitrarily partitioned sequential files.

Scanned by TapScanner

✱ 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 ~~capacity~~ capability to quickly reach the vicinity of a desired record.

✱ Instruction Cycle:

The time period during which one instruction is fetched from memory and executed when a computer is given instruction in machine language.

✱ Internal fragmentation:

Occurs when memory is devided 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 portion of data is smaller than the last partition

Scanned by TapScanner

Page
Date

* interupt

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

* Interupt handler:

A routine, generally part of the operating system. when an interupt 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.

* Kernel:

A portion 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 and responds to calls

Scanned by TapScanner

from processes and interrupts from device

* Kernel mode:

A priviledge mode of execution reserved for the kernel of the operating system. Typically kernel mode allows access to regions of main memory that are unavailable to processes executing a less-priviledged mode, and also enables execution of certain machine instructions that are restricted to the kernel mode. Also referred 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 responce to change in the other process(es) 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.

Scanned by TapScanner

* Logical Address:

A reference to a memory location independent of the current assignment of data to memory. A translation must be made to a physical address before the memory access can be achieved.

* 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 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.

Scanned by TapScanner

Malicious software is frequently concealed within or masquerades as legitime software. In some cases, it spreads itself to other computer via e-mail or infected disks. Types of malicious software include virus, Trojans horses, worms and hidden software for launching denical-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:

The subdividing of storage into independent sections.

* Microkernel:

A small priviledge operating system core that provides process scheduling, memory management and communication services and relies on other processes to perform some of the function traditionally

Scanned by TapScanner

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 processer. The same as multitasking using diff. terminology.

* Multiprogramming level:

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

* Multitasking:

A mode of operation that provides for the concurrent performance or interleaved execution of two or more computer tasks. The same as multiprogramming, using diff. terminology.

Scanned by TapScanner

Page
Date

* 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 and that provides services such as resource allocation, scheduling, input/output control and data management.

* page:

In virtual storage, a fixed length block that has a virtual address and that is transferred as a unit between main memory and ~~Exeond memory~~. Secondary memory.

* Page fault:

Occurs when the page containing a referenced word is not in main memory. This causes an interrupt and requires that the proper page be brought into main memory.

Scanned by TapScanner

Page
Date

* Page frame:

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

* Paging:

The transfer of pages between main memory and secondary memory.

* Physical Address:

The absolute location of a unit of data in memory (eg word or byte in memory blocks on secondary memory).

* Pipe:

A circular buffer allowing two processes to communicate on the producer-consumer model. Thus, it is a first in first out queue, written by one process and read by another. In some systems, the pipe is ~~generated~~ generalized to allow any item in the queue to be selected for consumption.

Scanned by TapScanner

Page
Date

* Preemption:

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

* Prepaging:

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. compare demand paging.

* Process:

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

* Process Control Block (PCB).

The manifestation of process in an operating system. It is a data structure containing information about the characteristics and state of the process.

Scanned by TapScanner

Page
Date

* 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 and that the processor needs to properly execute the process. The process state includes the contents of the various processor register such as the program counter and data registers; it also includes informations of use to the operating system, such as the priority of the process and whether the process is waiting for the completion of a particular I/O event. same as execution context.

* Program Counter (PC).

Instruction address register.

Page
Date

* Processor:

In a computer, a functional unit that interprets and executes instruction. A processor consists of at least an instruction control unit and an arithmetic unit.

* Programmed I/O.

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

* Real time system:

An operating system that must shedule and manage real time things.

* Real-time Tasks:

A task that is executed in connection with some process or function or set of events external to the computer system and that must meet one or more deadlines to interact effectively and correctly with the external environment.

Scanned by TapScanner

Page
Date

* Registers:

High speed memory internal to the CPU. Some registers 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 two programs on different machines interact using procedure call / return syntax and semantics. Both the called and 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 transmission of an enquiry message and the begining of the receipt of a response messages, measured

Scanned by TapScanner

at the enquiry terminals.

* Round Robin:

A sheduling algorithm in which processes 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 (e.g. termination of a child process or an input/output operation) returns control to the sheduler.

* Sheduling:

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

* 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 and Tapes

Scanned by TapScanner

Page
Date

* Segment

In virtual memory, a block that has a virtual address. The blocks of a program may be unequal length and 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 and increment. Depending on the exact definition of the semaphore, the decrement operation may result in the blocking of a process and the increment operation may result in the unblocking of a process. Also known as counting Semaphore and general Semaphore.

Scanned by TapScanner

Page
Date

* Sequential file:

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

* Session:

A collection of one or more processes that represents ~~a~~ ~~single~~ single interactive user application or operating system function. All keyboard and 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 and job control language commands. It functions as an interface between the user and the operating system.

Scanned by TapScanner

Page
Date

* 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 and the next item to be removed from the list is the item that has been in the list the 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 preference.

* 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 (P) operations (FIFO order).

Scanned by TapScanner

Page
Date

* Scapping:

A process 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 recieves control at a precoded location in a computer program.

* Synchronization:

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

Scanned by TapScanner

Page
Date

* System bus :

A bus used to interconnect major computer components. (CPU, 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 dispatchable unit of work. It includes a processor context (which include the program counter and stack pointer) and its own data area for stack. (to enable subroutine branching).
A thread executes sequentially and is interruptible so that the processor can turn to another thread. A process may contains one or more thread.

Scanned by TapScanner

Page
Date

* 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 on 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.

Scanned by TapScanner

Page
Date

★ 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 undocument routine embeded within a useful program. Execution of the program results in execution of the secret routine.

★ ~~Trojan horse~~: 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 the program results ~~into~~ in execution of secret routine.

Scanned by TapScanner

Page
Date

* User mode :

The least privileged mode of execution, Certain regions of memory and 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 address 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 and not by the actual number of main storage locations.

Scanned by TapScanner

☆ Virus :

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

☆ Weak Semaphore :

A semaphore in which all processes waiting on the same semaphore processed in an unspecified order. (i.e. the order is unknown or intermediate).

☆ Word :

An ordered set of bytes or bits, that is the normal unit of which information may be stored, transmitted or operated on within a given computer. Typically it a processor has a fixed length instruction set, then the instruction length is equals the word length.

Scanned by TapScanner

* Working set:

The working set with parameter
Δ for a process at virtual time
t, $w(t, \Delta)$ is the set of pages
of that process that have been
referenced in the last Δ time
units.

* Worm:

Program that can travel from
computer to computer across
network connections. May contain
a virus or bacteria.

* File allocation Table (FAT).

A table that indecates
the physical location on secondary
storage of the space allocated
to a file.
There is one file allocation
table for each file.

Scanned by TapScanner

Page
Date

* File management System.

A set of system software that provides services to users and applications in the use of files, including file access, directory maintenence, and access control.

* Encryption:

The conversion of plain text or data into unintelligible form by means of a reversible mathematical computation.

* Multilevel Security:

A capability that enforce access control across multiple levels of classification of data.

* Object Request broker:

An entity in an object oriented system acts as an intermediatory for requests sent from a client to server

Scanned by TapScanner

* Priority Inversion:

A circumstances in which the operating system forces a higher priority task to wait for lower priority task.

* Process discriptor:

Same as process control block (PCB)

* Process image:

All of the ingredients of a process, including program, data, stack and process control block.

* Server:

(1) A process that responds to request from clients via message.

(2) In a network, a data station that provides facilities to other stations.
Ex eg., a file server, a print server, a mail server etc.

Scanned by TapScanner

Page
Date

★. Assignment - 2.

Sub: O.S.

Q.1. Bankers algorithm.

Process	Allocation			MAX			Available (work)			Need (max-alloc)		
	A	B	C	A	B	C	A	B	C			
P0	0	1	0	7	5	3	3	3	2			
P1	2	0	0	3	2	2						
P2	3	0	2	9	0	2						
P3	2	1	1	2	2	2						
P4	0	0	2	4	3	3						

Ans:

Process	Allocation			MAX			Available			Need.		
	A	B	C	A	B	C	A	B	C	A	B	C.
P0	0	1	0	7	5	3	3	3	2	7	4	3
P1	2	0	0	3	2	2				1	2	2
P2	3	0	2	9	0	2				6	0	0
P3	2	1	1	2	2	2				0	1	1
P4	0	0	2	4	3	3				4	3	1

=> Need ≤ work => work = work + allocation

P0. 743 ≤ 332 ← × condition fails.

P1 122 ≤ 332 ← condition true.

W = work allocation

= 332 + 200

= 532

Scanned by TapScanner

P2 need ≤ work

600 ≤ 532 condition false

P3 need ≤ work

011 ≤ 532 condition true

ω = ω + allocation

= 532 + 332

= 743

x) P4 need ≤ work ω = ω + allocation

431 ≤ 523 = 743 + 002

= 745

P0 need ≤ work → ω = ω + alloc.

743 ≤ 745 = 745 + 010.

= 755

P2 need ≤ work → work = work + alloc.

600 ≤ 755 = 755 + 302.

= 1057.

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

Scanned by TapScanner

Page
Date

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

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

7	0	1
1	1	1
7	7	7
0	0	0

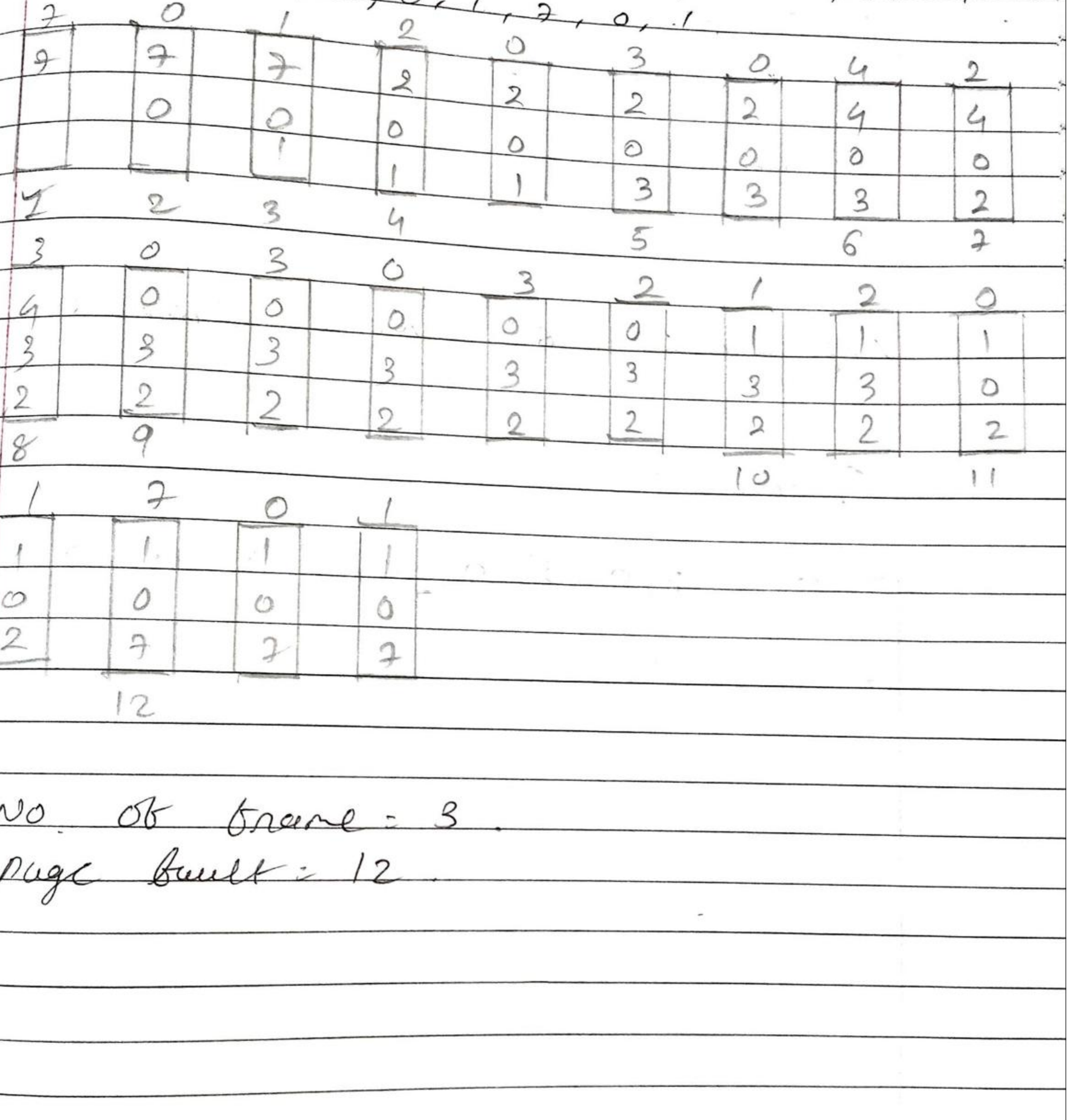
no of frames : 3 Page fault = 13.

Scanned by TapScanner

Page
Date

* LRU.

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



No. of frame = 3.

Page fault = 12.

Scanned by TapScanner

Page
Date

=) FIFO.

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

	1	3	0	3	5	6	3
	1	1	1	1	5	5	5
		3	3	3	3	6	6
			0	0	0	0	3
	1	2	3	4	5	6	

page fault : 6, no of frames : 3

=) Optimal :

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

	7	0	1	2	0	3	4	2	3
	7	7	7	7		3	3	3	3
		0	0	0		0	0	0	0
			1	1		1	4	4	4
				2		2	2	2	2
	1	2	3	4		5	6		

0 3 2 3

page fault = 6.

no of frame : 4.

Scanned by TapScanner

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 : 09
N A M E : NEEL S. GOLARANA
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 : 09
N A M E : NEEL S. GOLARANA
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 : 09
N A M E : NEEL S. GOLARANA
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: Neel Rana
Class: MCA III
Rollno: 09
Subject: Operating System - OS

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

Q1)

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

announces=$(($salary * 40  / 100))
echo "Announces is : " $announces
rent=$(($salary * 20 / 100))
echo "Rent is : " $rent
gross_pay=`expr $salary + $announces + $rent`
echo "Gross Pay Salary  = " $gross_pay
```

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

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh grosspay.sh
Enter Salary :
20000
Announces is :  8000
Rent is :  4000
Gross Pay Salary  =  32000
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

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

Q2)

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 the distance in kilometers  : "
read km
meter=`expr $km \* 1000`
echo " Distance in meter is : " $meter

cm=`expr $meter \* 100`
echo " Distance in centimeter is : " $cm

feet=`echo " scale = 2; $km * 3280.84" | bc`
echo " Distance in feet is : " $feet

inches=`echo " scale = 2; $km * 39370.08" | bc`
echo " Distance in inches is : " $inches
```

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

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh distance.sh
Enter the distance in kilometers  :
2
 Distance in meter is :  2000
 Distance in centimeter is :  200000
 Distance in feet is :  6561.68
 Distance in inches is :  78740.16
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

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

Q3)

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.

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

```
#area of rectangle and the circle
```

```
echo "Enter the length of the rectangle : "
read len
echo "Enter he breathe of the rectangle : "
read breath
area=`echo "scale = 2; $len * $breath " | bc`
echo "area of the rectangle is : " $area
peri=`echo "scale = 2; $area * 2 " | bc`
echo "perimeter of the rectangle is  :" $peri
echo "Enter the radius of the circle : "
read r
a=`echo " scale = 2 ; 3.14 * $r * $r " | bc`
echo "The area of the circle is : " $a
cir=`echo " scale = 2 ;  2 * 3.14 * $r " | bc`
echo "The circumference of the circle is : " $cir
```

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh area.sh
Enter the length of the rectangle :
10
Enter he breathe of the rectangle :
2.5
area of the rectangle is :  25.0
perimeter of the rectangle is  : 50.0
Enter the radius of the circle :
5.5
The area of the circle is :  94.98
The circumference of the circle is :  34.54
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

Q4)

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

****

```
# Sum of all the digit of the number

echo "Enter a five digit number : "
read num
number=$num
```

```
sum=0
while [ "$num" -gt 0 ]
do
	r=$(($num % 10))
	sum=`expr $sum + $r`
	num=`expr $num / 10`
done
echo " The sum of " $number " is : " $sum
```

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh sum.sh
Enter a five digit number :
12345
 The sum of  12345  is :  15
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

Q5)

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 -f1,3,4,6,7 -d":" /etc/passwd | tail -n 1
```

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_5.sh
neel:1000:1000:/home/neel:/bin/bash
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

****
Q6)

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 file

ls -l $file | cut -d' ' -f 1,2,3,4,5,6,7,8,9 -s | awk '{print $9 ":" $1 ":" $2 ":" $3 ":" $4 ":" $5 ":" $6 ":" $7 ":" $8}'
```

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

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_6.sh
Enter file name :
sum.sh
sum.sh:-rwxrwxrwx:1:neel:neel:239:Nov:21:10:33
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****
****
****
Q7)

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 the cost price of an item :"
read cost_price
echo "Enter the selling price of an item :"
read sell_price
if [ $sell_price -gt $cost_price ]
```

```
then
	echo "The seller made profit and thai is : " `expr $sell_price - $cost_price`
else
	echo "The seller made loss and that is : " `expr $cost_price - $sell_price`
fi
```

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_7.sh
Enter the cost price of an item :
10000
Enter the selling price of an item :
12000
The seller made profit and thai is :  2000
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_7.sh
Enter the cost price of an item :
10000
Enter the selling price of an item :
8000
The seller made loss and that is :  2000
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

Q8)
Check whether the entered no. is odd or even.

****

```
# check the given no is odd or not

echo "Enter the number  : "
read num

rem=$(($num % 2))
if [ $rem -eq 0 ]
then
	echo "The number is even "
else
	echo "The number is odd "
fi
```

****

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_8.sh
Enter the number  :
12
The number is even
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

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

Q9)
Check whether the entered no. is prime or not.

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

```
# check the no is prime or not

echo "Enter the number you want to check the prime or not : "
read num

i=2
flag=0

while test $i -le `expr $num / 2`
do
	if test `expr $num % $i` -eq 0
	then
		flag=1
	fi
	i=`expr $i + 1`
done

if [ $num -eq 1 ]
then
	echo $num " is niether prime nor composite "
else
	if [ $flag -eq 0 ]
	then
		echo $num " is prime number "
	else
		echo $num " is not prime number "
	fi
fi
```

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

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_9.sh
Enter the number you want to check the prime or not :
5
5  is prime number
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_9.sh
Enter the number you want to check the prime or not :
6
6  is not prime number
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

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

Q10)
Check whether the entered year is a leap year or not.

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

```
echo "Enter the year : "
read year

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

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

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_10.sh
Enter the year :
2020
 Year 2020 is leap year :
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_10.sh
Enter the year :
2019
 Year 2019 is not leap year :
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

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

Q11)
The script receives two file names as arguments, the script must check whether the files are same or not, if they are s
imilar
then delete the second file.

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

the script must check whether the files are same or not,
if they are similar then delete the second file.

```
#! /bin/bash

if [ ! -f $1 ]; then
       echo "$1 not found!"
       exit
fi
if [ ! -f $2 ]; then
       echo "$2 not found!"
       exit
fi


my_var=$(cmp -b $1 $2)
if test -z "$my_var"
then
     echo "Files are same"
     rm $2
     echo $2 "Deleted"
else
       echo "Files are not same"
fi
```

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

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_11.sh new.txt new1.txt
Files are same
new1.txt Deleted
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

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

****
Q12)

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

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

```
echo "Enter user name "
read name

who | grep $name > /dev/null

if [ $? -eq 0 ]
then
        echo "User is logged in "
        echo "Please enter a message to send him "
        read msg
        echo $msg
else
        echo "User is not logged in "
fi
```

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

Output:

```
neel@neel-VirtualBox:~/neel$ sh s_s_12.sh
Enter user name
neel
User is logged in
Please enter a message to send him
Hii !! How are you ?
Hii !! How are you ?
neel@neel-VirtualBox:~/neel$ sh s_S_12.sh
Enter user name
Piyu
User is not logged in
neel@neel-VirtualBox:~/neel$
```

****
****
****
****
Q13)

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

how many terminals has this user logged in.

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

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

Output

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

Q14)
Write a shell script to display the date with the format :- 25th October 2005 is a Tuesday.

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

```
# 9th December 2020 is a Wednesday.
d=`date +%d\ %B\ %Y\ is\ a\ %A`
echo $d
```

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

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_14.sh
10 December 2020 is a Thursday
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

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

Q15)
Write a shell script to display the appropriate message like : Good Morning / Good Afternoon / Good Evening

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

```
#get the current hour in (24 hrs format)
```

```
hour=$(date +"%H")

if [ $hour -ge 0 -a $hour -lt 12 ]
then
	echo " Good Morning , $USER "
elif [ $hour -ge 12 -a $hour -lt 17 ]
then
	echo " Good Afternoon , $USER "
elif [ $hour -ge 17 -a $hour -lt 19 ]
then
	echo " Good Evening , $USER "
else
	echo "Good Night , $USER "
fi
```

****

Output:

```
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$ sh s_s_15.sh
Good Night , neel
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$
```

****

Q16)
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".

****

```
echo "Press 1 for list all files of the current directory "
echo "Press 2 for print the current directory "
echo "Press 3 for print the date "

ch=1

while [ ch -ne 0 ]
do

 echo "Enter your choice from the above : "
 read ch

 case $ch in
 1)
  ls -l
```

```
 ;;

 2)
   pwd
 ;;
 3)
  date
 ;;
 *)
  echo "Invalid date : "
 ;;
done
```

****

Output:

```
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$ sh s_s_16.sh
Press 1 for list all files of the current directory
Press 2 for print the current directory
Press 3 for print the date
Enter your choice from the above :
1
total 0
-rwxrwxrwx 1 neel neel 550 Nov 21 12:05 area.sh
-rwxrwxrwx 1 neel neel 360 Nov 21 10:31 distance.sh
-rwxrwxrwx 1 neel neel 243 Nov 21 10:29 grosspay.sh
-rwxrwxrwx 1 neel neel 214 Nov 21 21:05 s_s_10.sh
-rwxrwxrwx 1 neel neel 227 Nov 21 21:58 s_s_12.sh
-rwxrwxrwx 1 neel neel 325 Nov 23 19:56 s_s_15.sh
-rwxrwxrwx 1 neel neel 391 Nov 23 20:17 s_s_16.sh
-rwxrwxrwx 1 neel neel 299 Nov 22 17:22 s_s_18.sh
-rwxrwxrwx 1 neel neel  23 Nov 22 17:24 s_s_19.sh
-rwxrwxrwx 1 neel neel 555 Nov 23 19:46 s_s_34.sh
-rwxrwxrwx 1 neel neel  46 Nov 21 12:19 s_s_5.sh
-rwxrwxrwx 1 neel neel 161 Nov 21 19:16 s_s_6.sh
-rwxrwxrwx 1 neel neel 326 Nov 21 19:34 s_s_7.sh
-rwxrwxrwx 1 neel neel 183 Nov 21 18:23 s_s_8.sh
-rwxrwxrwx 1 neel neel 422 Nov 21 20:37 s_s_9.sh
-rwxrwxrwx 1 neel neel 246 Nov 21 19:11 sum.sh
Enter your choice from the above :
2
/mnt/e/Rolwala/sem-3/o.s/practicles
Enter your choice from the above :
3
Mon Nov 23 20:18:02 IST 2020
Enter your choice from the above :
0
Invalid Option  :
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$
```

****
****
Q17)
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
ch=1
while [ $ch -ne 0 ]
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..";;

	*) echo "Invalid Choice.."

	esac

done
```

****
****
Output:

```
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$ sh s_s_17.sh
Enter 1st Integer :
20
Enter 2nd Integer :
```

```
10

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

Enter Your Choice :
1
Addition =  30

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

Enter Your Choice :
2
Subtraction =  10

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

Enter Your Choice :
3
Multiplication =  200

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 :
0
Exit..
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$
```

****

****
****
****
Q18)
Find the factorial of any number.
****
****
****

```
#shell scr:wqipt for factorial of a number
#factorial using while loop

echo "Enter a number"
read num

fact=1

while [ $num -gt 1 ]
do
        fact=$(($fact * $num))  #fact = fact * num
        num=$(($num - 1))       #num = num - 1
done
echo "FActorial of the number " $num " is " $fact
```

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

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_18.sh
Enter a number
5
FActorial of the number  5  is  120
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****
****
****
Q19)
Display the fibonacci series upto some number.
****
****
****

```
#Fibonnaci series :
echo "How many terms you want to generated  "
read n

x=0
```

```
y=1
i=2
echo "Fibonacci series upto $n numbers  :"
echo "$x"
echo "$y"

while [ $i -lt $n ]
do
	i=`expr $i + 1 `
	z=`expr $x + $y `
	echo "$z"
	x=$y
	y=$z
done
```

****

Output:

```
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$ sh s_s_19.sh
How many terms you want to generated
5
Fibonacci series upto 5 numbers  :
0
1
1
2
3
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$
```

****

Q20)
Two numbers are entered through the keyboard, find the power, one number raised to another.

****

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

i=0
ans=1

while [ $i -lt $pow ]
do
```

```
        ans=`expr $ans \* $exp`
        i=`expr $i + 1`
done


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

****

Output:

```
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$ sh s_s_20.sh
Enter Exponent  :
5
Enter Power :
2
5 ^ 2  = 25
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$ sh s_s_20.sh
Enter Exponent  :
2
Enter Power :
5
2 ^ 5  = 32
neel@NeelRana:/mnt/e/Rolwala/sem-3/o.s/practicles$
```

****

Q22)
Write a script which reports name and size of all files in a directory. whose sizes exceed 1000. The filenames shoul
d be printed in the descending
order of their sizes. The total no. of files must be reported.

****

```
 # 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.

 #! /bin/bash
 ls --sort=size -l | awk '$5 >= 500 {print $5,$9}'
```

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_22.sh
151552 which
4096 mydir
1994 s_s_17.sh
1650 s_s_16.sh
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

Q24)
Print the prime nos. from 1 to 300.

****

```
        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)) -eq 0 ]
        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
```

Output :

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_24.sh
2
3
5
7
11
13
17
19
23
25
29
31
35
37
41
43
47
49
53
55
59
61
65
67
71
73
77
79
83
85
89
91
95
97
101
103
107
```

109
113
115
119
121
125
127
131
133
137
139
143
145
149
151
155
157
161
163
167
169
173
175
179
181
185
187
191
193
197
199
203
205
209
211
215
217
221
223
227
229
233
235
239
241
245
247
251
253
257
259
263
265
269

271
275
277
281
283
287
289
293
295
299

neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
****
****
****
****
Output:

****
****
****
****
Q25)

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 $i ]
			then
				if [ $j -le $i ]
				then
					echo $i $j $k
				fi
			fi
		done
	done
done
```

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

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_25.sh
1 1 1
2 1 1
2 1 2
2 2 1
2 2 2
3 1 1
3 1 2
3 1 3
3 2 1
3 2 2
3 2 3
3 3 1
3 3 2
3 3 3
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

Q26)

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
```

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles/mydir$ ls -l
total 0
-rwxrwxrwx 1 neel neel 13 Dec  9 16:38 Hello.sh.82
-rwxrwxrwx 1 neel neel 58 Dec  9 16:38 file1.txt.82
-rwxrwxrwx 1 neel neel  0 Dec  9 14:42 neel.sh.82
-rwxrwxrwx 1 neel neel  0 Dec  9 16:43 s_s_26.sh
-rwxrwxrwx 1 neel neel 55 Dec  9 16:39 s_s_26.sh.82
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles/mydir$
```

****

Q27)
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.

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

```
# 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

 #! /bin/bash

 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:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_27.sh
Usage:
./27-findWordFromFile.sh <wordFile> <findFile ...>
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****
****

****

Q28)
Write a shell script which deletes all the lines containing the word "unix"
in the files supplied as arguments to it
****
****
****
****

```
#! /bin/bash
# ========================================================================
# sed '/foo/d' deleteFromFile.txt

#  the substring foo inside the foobar string is also replaced.
# If this is not the wanted behavior, use the word-boundary expression (\b)
# at both ends of the search string.
# This ensures the partial words are not matched.

# ========================================================================

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" $1
done
```

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

```
#Output:

#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi s_s_28.sh
#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_28.sh unix.txt
#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ cat unix.txt
#hello
#nee
l@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi unix.txt
#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ cat unix.txt
#hello
#unix is the best
#i love linux
#unix is my favourite os
#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_28.sh unix.txt
#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ cat unix.txt
#hello
#i love linux
```

#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$

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

Q29)
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.
****
****
****
****

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

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

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

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi s__29.sh
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s__29.sh file.txt
Searching file : file.txt...
Found in file.txt
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

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

Q30)
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
****
****
****
****

```
#zero arguments

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
```

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ touch new1.txt
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ cat new1.txt
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi new.txt
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ cat new.txt
Hello
My name is Neel
class MCA-3
Have a good day
Nice to meet you
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_30.sh new.txt new1.txt
s_s_30.sh: 19: [: !1: unexpected operator
s_s_30.sh: 19: [: !2: unexpected operator
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi s_s_30.sh
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_30.sh new.txt new1.txt
s_s_30.sh: 19: Syntax error: "(" unexpected (expecting "then")
```

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi s_s_30.sh
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_30.sh new.txt new1.txt
s_s_30.sh: 19: Syntax error: "(" unexpected (expecting "then")
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi s_s_30.sh
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_30.sh new.txt new1.txt
s_s_30.sh: 19: 1%2: not found
cp: 'new.txt' and 'new.txt' are the same file
'new.txt' copied to -> new.txt
s_s_30.sh: 19: 2%2: not found
'new.txt' copied to -> new1.txt
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ cat new.txt
Hello
My name is Neel
class MCA-3
Have a good day
Nice to meet you
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ cat new1.txt
Hello
My name is Neel
class MCA-3
Have a good day
Nice to meet you
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi s_S_30.sh
```

****

Q31)
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 curent directory."
echo "Which have Read,Write and Execute permisions. "
for file in *
do
 if [ -f $file ]
 then
  if [ -r $file -a -w $file -a -x $file ]
  then
   ls -l $file
  fi
 fi
done
```

****

Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_S_31.sh
The list of File Names in the curent directory.
Which have Read,Write and Execute permisions.
-rwxrwxrwx 1 neel neel 550 Nov 21 12:05 area.sh
-rwxrwxrwx 1 neel neel 360 Nov 21 10:31 distance.sh
-rwxrwxrwx 1 neel neel 243 Nov 21 10:29 grosspay.sh
-rwxrwxrwx 1 neel neel 227 Nov 23 22:09 s_S_31.sh
-rwxrwxrwx 1 neel neel 214 Nov 21 21:05 s_s_10.sh
-rwxrwxrwx 1 neel neel 227 Nov 21 21:58 s_s_12.sh
-rwxrwxrwx 1 neel neel 325 Nov 23 19:56 s_s_15.sh
-rwxrwxrwx 1 neel neel 391 Nov 23 20:17 s_s_16.sh
-rwxrwxrwx 1 neel neel 670 Nov 23 20:39 s_s_17.sh
-rwxrwxrwx 1 neel neel 299 Nov 22 17:22 s_s_18.sh
-rwxrwxrwx 1 neel neel 246 Nov 23 21:17 s_s_19.sh
-rwxrwxrwx 1 neel neel 181 Nov 23 21:41 s_s_20.sh
-rwxrwxrwx 1 neel neel 235 Nov 23 21:57 s_s_24.sh
-rwxrwxrwx 1 neel neel 555 Nov 23 19:46 s_s_34.sh
-rwxrwxrwx 1 neel neel 46 Nov 21 12:19 s_s_5.sh
-rwxrwxrwx 1 neel neel 161 Nov 21 19:16 s_s_6.sh
-rwxrwxrwx 1 neel neel 326 Nov 21 19:34 s_s_7.sh
-rwxrwxrwx 1 neel neel 183 Nov 21 18:23 s_s_8.sh
-rwxrwxrwx 1 neel neel 422 Nov 21 20:37 s_s_9.sh
-rwxrwxrwx 1 neel neel 246 Nov 21 19:11 sum.sh
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

Q32)
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

#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practiclis$ sh s_s_32.sh file.txt
#file.txt -> file with lines: 2
#neel@NeIiiana:/mnt/e/ROLWALA/sem-3/o.s/practicles$

****
****
****
****
Q33)
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.
****
****
****
****

```
# 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.


 #! /bin/bash

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

```
        fi
done
```

****
Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_33.sh new.txt
new.txt exists
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****
Q34)
Accept the marks of 5 subjects and calculate the percentage and grade.
****

```
#calculate the percentage and marks of 5 subjects

echo "Enter the five subject marks for the student"
read m1 m2 m3 m4 m5
sum1=`expr $m1 + $m2 + $m3 + $m4 + $m5`
echo "Sum of 5 subjects are: " $sum1
per=`expr $sum1 / 5`
echo "The percentage of the marks of five subject is : " $per

if [ $per -ge 80 ]
then
      echo " you got Grade : A "
elif [ $per -ge 70 ]
then
      echo " You got Grade : B "
elif [ $per -ge 60 ]
then
      echo " You got grade : C "
elif [ $per -ge 50 ]
then
      echo "You got Grade : D "
else
      echo "FAIL :"
fi
```

****
Output:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_34.sh
```

```
Enter the five subject marks for the student
50 80 70 90 60
Sum of 5 subjects are:  350
The percentage of the marks of five subject is :  70
 You got Grade : B
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

Q35)

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:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi s_s_35.sh
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_35.sh
Armstrong number :  1
Armstrong number :  153
Armstrong number :  370
Armstrong number :  371
Armstrong number :  407
```

****

Q36)
Accept the measure (angles) of a triangle and display 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:

```
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi s_s_36.sh
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_36.sh
Enter angle
120
Obtuse angle
neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

Q37)
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

echo "The following no is devisible by 7 \n"

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

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

	num=$((num+1))

done

echo "\n"
```

****

Output:

```
#output

#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ vi s_s_37.sh
#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_37.sh
#The following no is devisible by 7

#7 14 21 28 35 42 49 56 63 70 77 84 91 98

#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

Q38)
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:

```
#output

#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_38.sh
#Enter 1st Number :
#3
#Enter 2nd Number :
#6
#Enter 3rd Number :
#1
#6 is Largest..
#1 is Smallest...
#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

****

Q39)

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:

```
#output

#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_39.sh
#Enter first number : 15
#Enter second number : 55
#HCF of 15 and 55 = 5
#LCM of 15 and 55 = 165
#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$
```

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

Q40)

Display the dates falling on Sundays of the current month.

****

o "Sundays in current month are:"

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

****

Output:

```
#output

#neel@NeelRana:/mnt/e/ROLWALA/sem-3/o.s/practicles$ sh s_s_40.sh
#s_s_40.sh: 1: o: not found
# ----- Using AWK -----
# 1
# 6
# 13
# 20
# 27
```

****