Linux OS

Basics

ASCII
a
b
2
_

alphanumeric
  a-z A-Z 0-9 _

Unix/Linux

History

GE	Bell/At&T	MIT

PDP7

Ken Thompson

Dennis Ritchie C
Brian Kerningham

PDP11

MULTICS
  multiplexed operating computing systems
UNICS
  uniplexed operating computing systems
UNIX

UNIX → C

university of california, berkeley
mid 70s BSD
  (berkley software distribution)

At&T	UNIX
HP	HPUX
IBM	AIX
Sun Microsystems	Solaris

Richard Stallman	(FSF)	free software foundation
Linus Torvalds		
	MINIX	
	Linux	

RedHat

standards

BSD
system V
POSIX (IEEE)

1a	OS
1b	RTOS
1c	threads (pthreads)

Shell

sh	shell
csh	c-shell
zsh	zeeshell
tsh	
tcsh	
bash	born-again

Linux flavors

ubuntu
debian
kali
raspbian
redhat
centOS
Fedora

misc

directory == folder

command	options	operand
mkdir		captain
ls	-l	
cd		captain

redirection operators

>	output redirection
>>	output redirection (append)

basic commands

clear	clear the screen
date	
cal	
who	
alias	
unalias	

more commands

echo

followed by a string

read

read from the input

ulimit

-a	limits related to the OS

command paths

/bin
/usr/bin

dir & file commands

pwd	
cd	change
	cd captain
	cd ..
	cd .
	cd ~ (home)
mkdir	create directory
ls	list
cat	
head	
	-n
tail	
	-n

less
more
uniq

cp	copying
mv	move
	renaming
rm	remove
rmdir	remove an empty directory
rm -r	remove a full directory

file system

various file systems

fat
    file allocation table
fat32
fat64
ntfs
    new tech file system
apfs
    apple file system
ext3/4
    journaling
    linux uses
procfs

inode

unique number given to every file
inode data struct
    all info file

types of files

-	regular
d	directory
p	pipe/fifo
b	block device driver
c	char device diver
s	socket

/dev

/dev/tty0
/dev/tty1
/dev/tty2

/dev/pts/1
/dev/pts/2

path

relative path
step by step

```
/amazon/hyd15/floor2/room7/chair5
cd ..
/amazon/hyd15/floor2/room7/
cd ..
/amazon/hyd15/floor2
cd ..
/amazon/hyd15
cd floor3
/amazon/hyd15/floor3/
cd room6
/amazon/hyd15/floor3/room6
cd chair2
/amazon/hyd15/floor3/room6/chair2
```

absolute path

```
/amazon/hyd15/floor2/room7/chair5
cd /amazon/hyd15/floor3/room6/chair2

/amazon/hyd15/floor2/room7/chair5
```

permissions

participants.xlsx

user	group	others
r w x	r w -	r - -
1 1 1	1 1 0	1 0 0
7	6	4

users & groups

every user belongs to a group

u	g	o	r	w	x

o+r	read for others
+r	read for all
g-w	remove write permissions for group

directory permissions

writing into folder	→	creating new files
reading a folder	→	viewing list of files & folders
executing folder	→	can’t even cd

root

super user

commands for adding

groupadd training
usermod -g

file descriptors

temporary number given to each open file
smallest available number is assigned

open file descriptors

stdin	0
stdout	1
stderr	2
hello.txt	3

editors

general

sublimetext
atom
vscode
notepad++

command based CLI

linux editors

vi (vim)
gedit
atom
sublimetext
vscode
emacs
notepad++
nano

vi

visual editor
vim

two modes

insertion

	i
	a

command

	esc
:w	save
:wq	save & quit
:q!	quit without saving
:w!	save as

yy	copy
p	paste
dd	cut
	delete
u	undo
ctrl+r	redo
4yy	copies four lines

specials (w.r.t strings)

wild cards

*
?

escape characters

\”	“
\\	\
\$	$

grep

-i	ignore-case
-v	invert the condition
-n	line number
-c	total counts
-w	match the entire word
-m	max counts
-A2	also displays 2 lines after the match
-B3	also displays 3 lines before the match
-C5	also displays 5 lines before & after match
-E	extended grep (includes regex)

regular expressions

meta characters

^	line starts with	
$	ends with	
[ ]	pick from within that range	
	[a-z]	all lower
	[A-Z]	all upper
	[a-zA-Z]	all alphabets
	[a-egtkl]	[abcdegtkl]
	[0-9]	any digit
	[0-39]	[01239]
.	place holder for one character (except newline)	
*	zero or more occurrences	
?	zero or one occurrence	
+	one or more	
{x, y}	min x number of times	
	max y number of times	
	{1,}	one or more (same as +)
	{1,4}	min 1 max 4
	{1}	only 1

specail sequences

\d	digits
	[0-9]

\D	any non digit
\s	whitespace (space tab newline)
\S	invert of \s
\w	alphanumeric [a-zA-Z_]
\W	
\\|	or "akash\\|akshat" looks for akash or akshat anywhere in the file
\b	word boundary
\\\\	\\
*	*
\[	[

more file commands

find

find	where	option	operand	more options
find	/home/nigam	-type	d	-exec

-empty	empty files	
-name	name of the file	
-type	type of the file	
	f	file
	d	directory
	p	pipe/fifo
-exec	execute something over find	
-size		
	c	bytes
	k	Kilobytes
	b	block(512bytes)
	M	mega
	G	giga
-delete		

```
find . -size 0 -delete
find . -empty -delete
find . -type f -empty -delete
Find . -type f -empty -exec chmod -r {} \;
```

wc

-l	line
-c	character

-w	word

sort

-r	reverse
-n	numeric
-k	column
-c	
-u	unique
-o	store it in an output file

uniq

-c	count
-d	only repeated lines
-f	skip some words
-s	skip some characters
-i	ignore case

links

hard links
symbolic link

ln

	create a hard link
-s	create symbolic link (soft)

shell scripts

variables

local
shell
env

local variables

can start with	a-z	A-Z	_	
can have	a-z	A-Z	_	0-9

hello23

max_temp
hi_34_XX

invalid:
- 23hello
- hi-we
- hy,iu

environment variables

SHELL
LOGNAME
HOME
PWD
PATH
IFS
- internal field separator

special variables

$0	filename
$n	command line arguments
	$1
	$2
$#	total number of arguments
$*	all the arguments
$@	
$?	exit status of the last command
$$	current pid
$!	pid of last process executed

arithmetic

let
- to ensure its a mathematical expressions

+	
-	
*	mul
/	div
%	modulus (remainder)
=	assignment

expr ` `
(( ))

conditional

if

```
if [ conditional expression ]
then
        set of commands
else
        other commands
fi
```

the square brackets are called as test
can also use the keyword test

case

```
case $data in
        )
                ;;
        )
                ;;
        *)      default
```

loops

- while
- for
- until

while

```
while [ condition ]
do
        commands
done
```

operators

arth

+	
-	
*	mul
/	div
%	modulus (remainder)
=	assignment
==	equality

!=

relational

-eq	equality
-ne	not equal to
-lt	lesser than
-le	less than or equal to
-gt	greater than
-ge	greater than or equal to

strings

=	equal to
!=	not equal to
-z	size is zero
-n	size is non-zero

logical

&&	and
||	or
-o	or
-a	and
!	not

file

-f	file exists & is a regular file
-d	file exists & is a directory
-e	file exists
-x	file is executable
-w	file is writable
-r	file is readable
-p	file is pipe
-S	file is socket
-s	file exists & is non zero in size

key words

if
elif
else
then
test
while
do
done

process

everything in linux → file
if it’s running its a process

pid

man

1	commands
2	system calls
3	functions
4	
5	
6	
7	misc , signals

scheduling algorithms

FCFS

First Come First Serve
simple to implement
queue
poor performance

SRT (SJF) (SJN)

shortest remaining time (shortest job first)
CPU time is known in advance

Priority Based Scheduling

Round Robin

polling

1. interrupts
2. priorities

    a. increase the frequency
    b. increase the quantum
3. time scheduling (quantum)

processors & processes & threads

multi cores

2 burner	2 core	40 minutes
4 burner	4 cores	30 minutes

multi processors

iPod

2 burner	2 core

knife, plate , water

thread

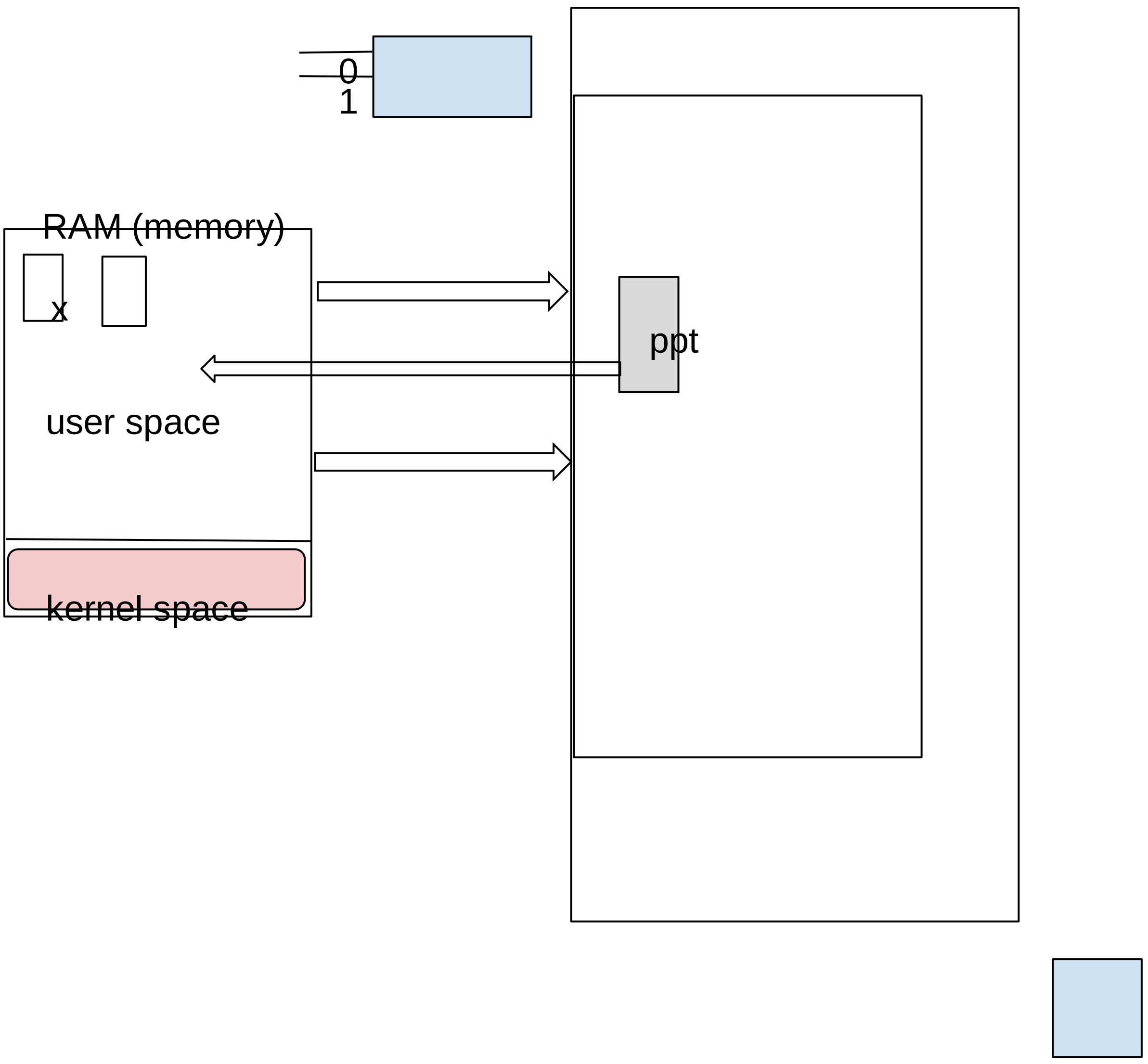
concurrency

appearance of simultaneous tasks

multi processes
multi threads

memory management

virtual memory



block memory

1 litre bottles		
0.7	milk	1
1.6	water	2
0.4	coffee	1
0.8	oil	1

list of bottles :

bot1	milk
bot2	water
bot3	water
bot4	coffee
bot5	oil

inventory of items:

milk	bot1
water	bot2, bot3
coffee	bot4
oil	bot5

2 litre bottles

0.7	milk	1
1.6	water	1
0.4	coffee	1
0.8	oil	1

list of bottles :

bot1	milk
bot2	water
bot3	coffee
bot4	oil

inventory of items:

milk	bot1
water	bot2
coffee	bot3
oil	bot4

0.1 litre bottles

0.7	milk	7
1.6	water	16
0.4	coffee	4
0.8	oil	8

list of bottles :

bot1	milk
bot2	milk
bot3	milk
bot4	milk

….
….

….

inventory of items:

milk	bot1, bot2, bot3, bot4, …..
water	bot8, …..
coffee	botx
oil	botx

0.4 litre bottles

0.7	milk	2
1.6	water	4
0.4	coffee	1
0.8	oil	2

list of bottles :

bot1	milk
bot2	milk
bot3	water
bot4	water
….	
….	
….	

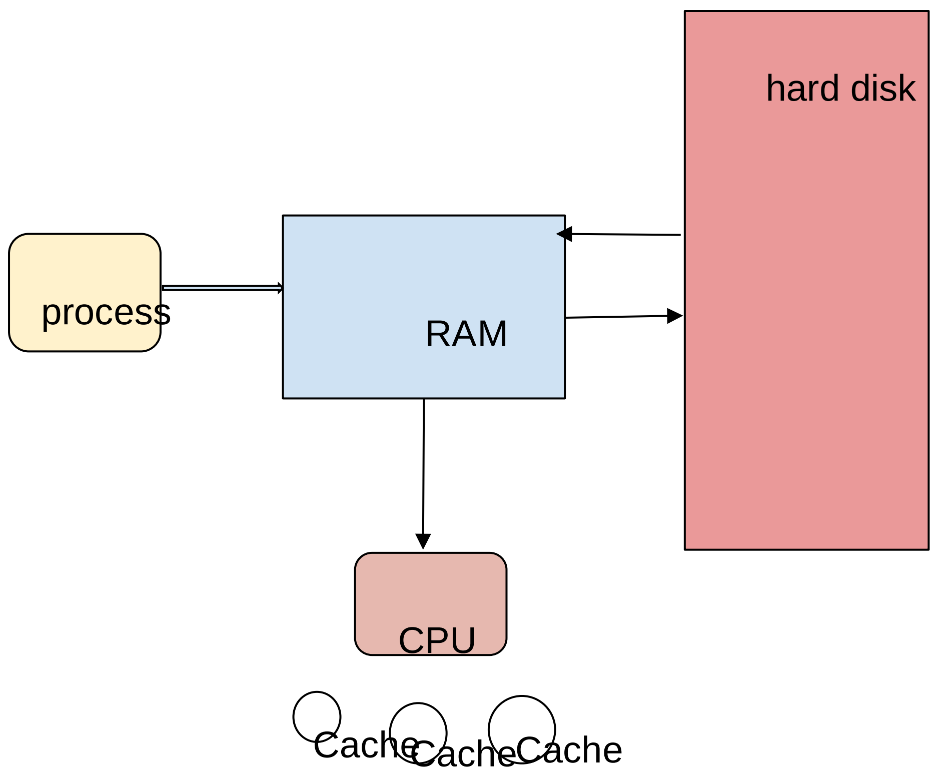
inventory of items:

milk	bot1, bot2
water	bot3, bot4, bot5, bot6
coffee	bot7
oil	bot8, bot9

terminologies & imp points

CPU
RAM
cache
- temporary (fast) memory
- in CPU is like tiny RAM in the CPU

opening a file
- Hard Disk
- modification → RAM
- after saving send to Hard Disk



Virtual Memory
Resident Memory
memory-over allocation

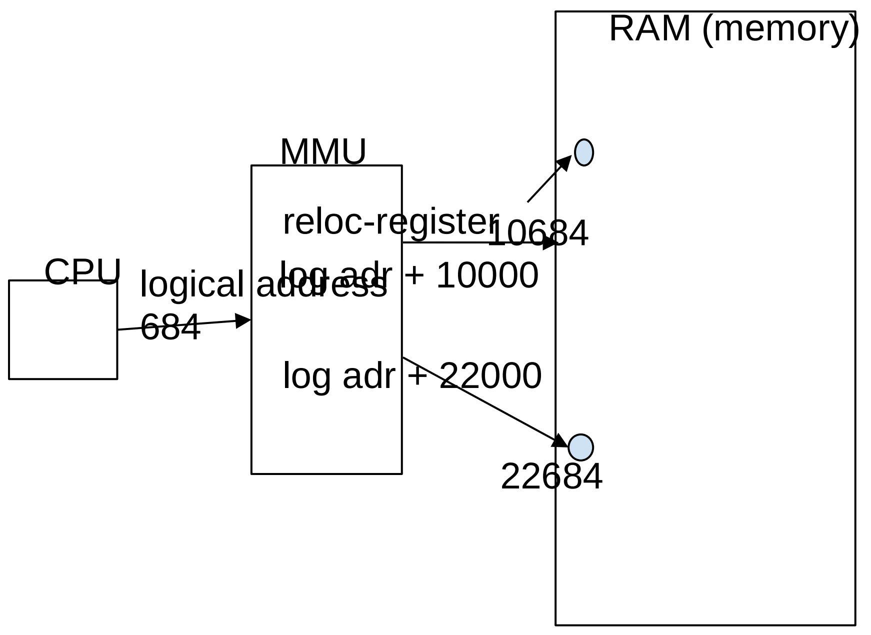
allocation of memory
- give
- used

OOM
- out-of-memory

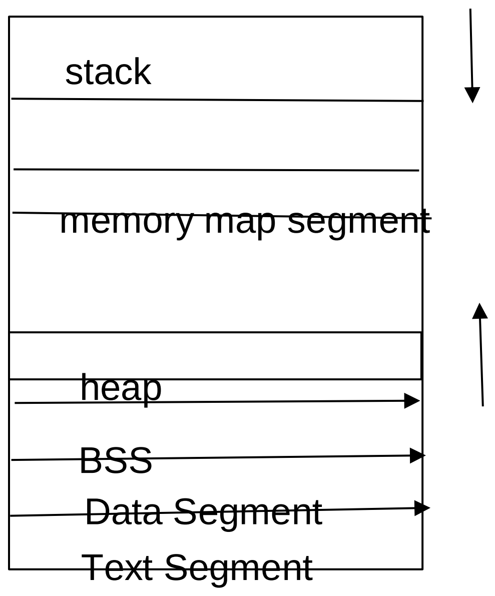
memory management
multiple levels:
- Hardware Assisted - MMU
- Software assisted -

Logical Address
- (virtual address)
- MMU will map logical address to its actual physical address
- generated by CPU

Physical address
- actual or real address in the RAM
- generated by MMU
- actual hardware stored
- listed in PTE
  - Page Table Entry



memory of a process



stack → local data from functions, auto deleted
heap → dynamically allocated data from malloc etc., have to be manually deleted

BSS
    uninitialised static data

Data
    initialised static data

Text
    code segment

paging

- PTE
- TBL
- demand paging
- journaling

fragmentation

4 floors
3 rooms

inventory register
20 green

F1 R1	12 red, 4 green
F1 R2	14 green, 2 blue
F1 R3	12 Blue, 4 Yellow
F2 R1	14 yellow, 2 green
F2 R2	2 green, 10 Blue, 4 yellow
F2 R3	2 yellow
green	F1 R1, F1 R2, F2 R1, F2 R2

defragment

F1 R1	12 red, 4 green
F1 R2	14 green, 2 green
F1 R3	2 green, 14 Blue,
F2 R1	8 blue, 8 yellow
F2 R2	12 yellow
F2 R3	2 yellow

linux

F1 R1	12 red
F1 R2	16 yellow
F1 R3	2 yellow
F2 R1	16 green
F2 R2	2 green, 14 green
F2 R3	
F3 R1	14 blue

F3 R2
F3 R3

```
myLinux$ mycat file1.txt

myLinux$
```

Networking

overview

MAC
- Media Access Control
- 48 bit
- hardware address

IP address
- 220.40.13.167
- Ipv4
- Ipv6

localhost
- 127.0.0.1
- loopback address

DNS
- Domain Name System
- www.google.com

host google.com
- similar:
  - dig
  - nslookup

whois
ping
- packet internet groper

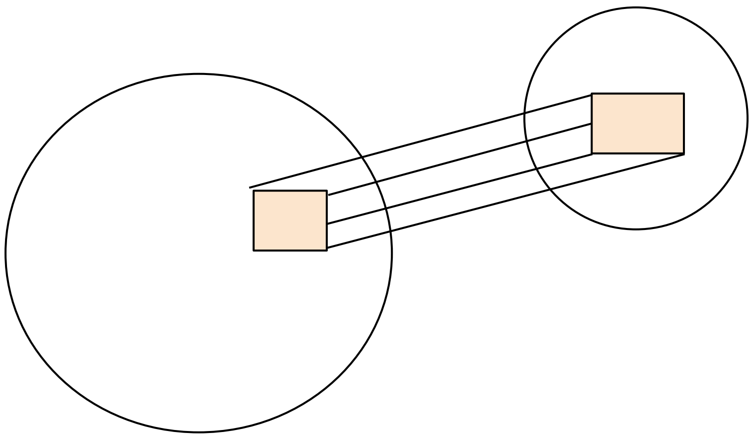
traceroute
netstat

remote connections

FTP	File Transfer Protocol
Telnet	
SSH	Secure Shell (Secure Socket Shell)
RDP	Remote Desktop Protocol
HTTP	

SSH
- OpenSSH
- sudo apt install openssh-client
- sudo apt install openssh-server
- ssh remote_username@remote_host

scp
- secure copy
- transfer of files



version control system

importantfile.doc
importantfile_final.doc
importantfile_final_1.doc

VCS
- SCM (source code management)
- RCS (revision control system)

popular VCS

- Git
- CVS
- Subversion
- Perforce
- Mercurial

Git Repository
- local
- remote

GitHub

- cloud for code
- code hosting service

- other:
  - GitLab
  - Bitbucket

clone
- (download or fetch)

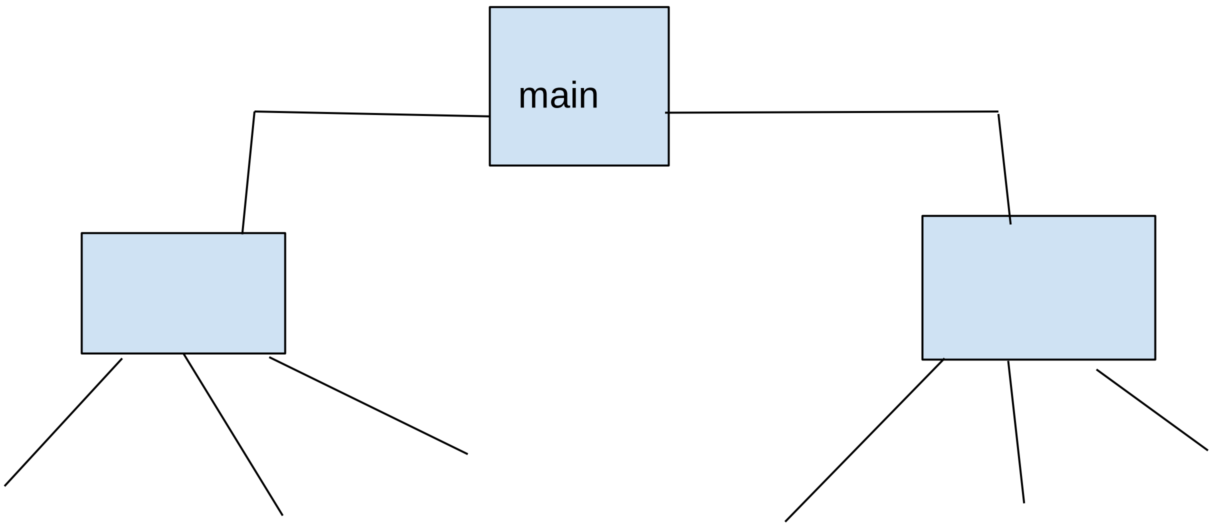
three spaces:
- working directory
  - actual files & folders
- staging area
  - update versions of files
  - modifications
- commit history

commands

git init
git add <files>
git status
git commit
- -m message

git diff

git config --global user.email “hi@how.com”



system programming

1. application programming
2. system programming
3. kernel programming

system calls

codes inside the kernel

file based system calls

open

opens a file

O_RDONLY	read only
O_WRONLY	write only
	previous content is deleted
O_RDWR	read & write
O_APPEND	append mode

read

(from where -fd, read into where - buffer, how many bytes/chars)
return → number of bytes/characters read

write

(where to write -fd, what to write - buffer, how many bytes/chars)
return → number of bytes/characters write

lseek

lseek(fd, OFFSET, from where)

fd	file desc
from where	

SEEK_CUR	(current)
SEEK_END	(end)
SEEK_SET	(start)

create()

unlink()

chmod()

close()

dup()

create a duplicate fd

dup2()

create a duplicate fd, with number specified by user

p
\n
k
a
a
a

printf is nothing but

write(1, …, …)

0	stdin
1	stdout
2	stderr

modes of operation

user mode

kernel mode

process

ps

information about process

pid	pid
ppid	ppid
mem	
virtual mem	
actual ram	
shared mem	
start time	stime
elapsed time	etime
running time	
cpu/kernel	time
uid	
gid	
name/command	cmd
cpu usage	
priority	pri
state	stat
tty	tty
num of threads	nlwp

init

pid 1
first process to run

fork()

creates a new process
returns 0 to child
child's pid to the parent
shared with child:
- code written after fork
- open file descriptors

wait(NULL)

wait for a child process to exit

waitpid()

wait for a particular pid to exit

state

R	Run	
		using CPU resources
S	Sleep	
		wait, delay
T	Stop	
		pause

Z	zombie
D	uninterruptible

I	kernel threads
+	needs a stdout
s	session leader

signals

kill

kill()

	action	keyboard	handled?
SIGTERM	end	no	yes
SIGINT	end	ctrl + c	yes
SIGKILL	end	no	no
SIGSTOP	stop	ctrl + z	no
SIGQUIT	end	ctrl + \	yes
SIGALARM	end self	no	yes

from keyboard:

SIGINT SIGSTOP SIGQUIT

two signals can not be handled:
SIGKILL SIGSTOP

other functions & system calls

memset()
ftok()
perror()

Inter Process Communication

types of communication

primitive

pipes/fifos

sys V

message queues
shared memory
semaphores

POSIX

message queues
shared memory
semaphores
mutex

pipes

pipe

P1 write → pipe → P2

1. unidirectional
2. read data is deleted
3. separate cursors for read & write
4. read process can not move ahead until write is done
5. only related processes are communicating
6. everything happens in the main memory (RAM)

fifos

named pipes

1. unidirectional
2. read data is deleted
3. separate cursors for read & write
4. both the ends of the fifo should be open
5. unrelated process can use it for communication
6. read process can not move ahead until write is done
7. everything happens in the main memory (RAM)

multithreading

pthread_t	structure
pthread_create()	(&ta, NULL, (void) (*) function, NULL)
	last parameter →
pthread_join()	
pthread_self()	thread id

sys V IPCS

key
unique id
xxxget()
- msgget()
- shmget()
- semget()

ipcs

list of IPCs in sys V

-q	msg queues
-m	shared memory
-s	semaphores
-l	limit

ipcrm

id:

-q	msg queues
-m	shared memory
-s	semaphores

Message Queue

1. key = ftok()
2. id = msgget(key)
3. msgsnd(id,)   or   msgrcv(id, ….)
4. msgctl(id, …..)

Shared Memory

broadcast
data remains until overwritten
synchronisation issues
    race around condition

1. key = ftok()
2. id = shmget(key)
3. shmat(id,)
4. shmctl(id,)

posix shared memory:
    gcc -lrt

Semaphore

counting semaphore
count = 1
    binary semaphores

posix semaphores
    gcc -lrt -lpthread

project submission

Way of submission: Email at submissions@learnoa.com
Deadline: Sunday (26th June)
Format: Doc file
late submissions will not be accepted.
Mention your batch name in the email

use amazon email IDs (discussion on hold)

revision

regex revist

^	start of the string
$	end
[ ]	
	[abcdghpw]
.	any character
{x}	x num of times
{x,y}	min x num of times, at most y num of times
{x,}	min x num of times, any num of times
+	one or more
?	zero or one
*	zero or more

\s	white spaces (space, tab, newline ……)
\d	numbers
	[0-9]
\w	alphanumeric character (including underscore)

a.a	(ata) (aTa) (a_a) (a6a)	
an?a	aa	ana

signals

SIGTERM	terminates
SIGINT	terminates, can be send from keyboard (ctrl +c)
SIGKILL	terminates, can not be handled
SIG	