Lab 1: 29th Aug 2025 - Caesar's Cipher

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>

void encrypt(char *in, int k, char *out) {
    int i = 0;
    while (in[i]) {
        char c = in[i];
        if (isalpha(c)) {
            char b = islower(c) ? 'a' : 'A';
            out[i] = ((c - b + k) % 26) + b;
        } else {
            out[i] = c;
        }
        i++;
    }
    out[i] = '\0';
}

void decrypt(char *in, int k, char *out) {
    encrypt(in, 26 - (k % 26), out);  // use encryption logic with inverse key
}

int main() {
    char txt[1000], enc[1000], dec[1000];
    int k;

    printf("Text: ");
    fgets(txt, sizeof(txt), stdin);
    txt[strcspn(txt, "\n")] = '\0';  // remove newline

    printf("Key: ");
    scanf("%d", &k);
    k %= 26;

    encrypt(txt, key, enc);
    decrypt(enc, key, dec);

    printf("Original Text : %s\n", txt);
    printf("Key           : %d\n", key);
    printf("Encrypted     : %s\n", enc);
    printf("Decrypted     : %s\n", dec);


    return 0;
}
```

O/P:

Original Text : Amrita
Key : 3
Encrypted : Dpulwd
Decrypted : Amrita

////

Lab 2: 12th Sept 2025 - Hill Cipher

////

Lab 3: 19th Sept 2025 - Railfence Encryption

Railfence Encryption:

P N N
O A A
N N

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
void main(){
  int i,j, len, rails, count, code[100][100];
  char str[100];
  printf("Enter the secret message: \n");
  gets(str);
  len=strlen(str);
  printf("Enter Number of rails: \n");
  scanf("%d",&rails);
  for(i=0;i<rails;i++){
    for(j=0;j<rails;j++){
            code[i][j]=0;
    }
```

```
 }
 count=0;
 j=0;
 while(j<len){
   if(count%2==0){
     for(i=0;i<rails;i++){
              code[i][j]=(int)str[j];
              j++;
     }
   } else {
            for(i=rails-2;i>0;i--){
               code[i][j]=(int)str[j];
               j++;
            }
   }
 }
 count++;

 for(i=0;i<rails;i++){
   for(j=0;j<len;j++){
           if(code[i][j]!=0){
             printf("%c ",code[i][j]);
           }
   }
 }
 printf("\n");
 getch();
}
```

O/p:

```
Enter the secret message: PONNANNA
Enter number of rails: 3
P N N O A A N N
```

//

Linux Commands:

```
Hello world.


┌──(root㉿kali)-[/home/kali/Desktop]
└─# exit

┌──(kali㉿kali)-[~/Desktop]
└─$
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ mkdir -p lab/file

┌──(kali㉿kali)-[~/Desktop]
└─$ ls
amrita     ashoka.pcapng   CyberSec   demo2.txt  kalitorify  lab       test1pwd.txt
amrita.pub  authorized_keys  demo1.txt  FLAG1.txt  kali.txt   Sublist3r  wordlist.txt

┌──(kali㉿kali)-[~/Desktop]
└─$ ls -ld lab
drwxrwxr-x 3 kali kali 4096 Sep 19 17:10 lab

┌──(kali㉿kali)-[~/Desktop]
└─$ cd lab

┌──(kali㉿kali)-[~/Desktop/lab]
└─$ ls
file

┌──(kali㉿kali)-[~/Desktop/lab]
└─$ ls -ld file
drwxrwxr-x 2 kali kali 4096 Sep 19 17:10 file

┌──(kali㉿kali)-[~/Desktop/lab]
└─$ sudo chown -R kali:kali file

┌──(kali㉿kali)-[~/Desktop/lab]
└─$ ls -ld file
drwxrwxr-x 2 kali kali 4096 Sep 19 17:10 file
```

////

Lab-4: 3rd Oct 2025 - Shell Scripting in Linux

bash/Zsh f1

```
┌──(kali㉿kali)-[~/Desktop]
└─$ echo $SHELL
/usr/bin/zsh

┌──(kali㉿kali)-[~/Desktop]
└─$ ec
echo         ecryptfs2john  ectool
```

NOTE: USE DOUBLE TAB TO GET INTELLIGENT CODE COMPLETION RESPONSE FOR UNKNOWN COMMANDS

```
┌──(kali㉿kali)-[~/Desktop]
└─$ echo
```

////

f1.sh>

```
echo "Welcome to shell scripting"
echo "Current shell is... $SHELL"
echo "System Security Lab"
echo "Display today's date"
date
echo "MAC ADDR:"
getmac
echo "System Host name is:"
hostname
echo "Mapping of logical to physical addresses using ARP"
arp -a
echo "Display all user names:"
uname
echo "Current path is..."
pwd
echo "Interface configuration:"
ifconfig
echo "Ping Google.com:"
ping www.google.com
```

////

```
┌──(kali㉿kali)-[~/Desktop]
└─$ bash f1.sh
Welcome to shell scripting
Current shell is... /usr/bin/zsh
System Security Lab
Display today's date
Fri Oct  3 04:28:54 PM IST 2025
MAC ADDR:
f1.sh: line 7: getmac: command not found
System Host name is:
kali
Mapping of logical to physical addresses using ARP
? (192.168.92.254) at 00:50:56:eb:eb:25 [ether] on eth0
? (192.168.92.2) at 00:50:56:e8:28:5b [ether] on eth0
Display all user names:
Linux
Current path is...
/home/kali/Desktop
Interface configuration:
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
```

```
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 7a:08:27:da:c3:7e  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 6 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.92.128  netmask 255.255.255.0  broadcast 192.168.92.255
        inet6 fe80::f01c:49c6:266d:6702  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:27:5b:50  txqueuelen 1000  (Ethernet)
        RX packets 645637  bytes 957605177 (913.2 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 29072  bytes 1803429 (1.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 8  bytes 480 (480.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 480 (480.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

Ping Google.com:
PING www.google.com (142.250.67.164) 56(84) bytes of data.
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=1 ttl=128 time=67.7 ms
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=2 ttl=128 time=110 ms
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=3 ttl=128 time=120 ms
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=4 ttl=128 time=68.5 ms
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=5 ttl=128 time=138 ms
64 bytes from bom12s07-in-f4.1e100.net (142.250.67.164): icmp_seq=6 ttl=128 time=156 ms
^C
--- www.google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 67.705/109.990/156.466/32.997 ms
```

bash f2.sh

f2.sh>

```
echo "Welcome to shell scripting"
echo "Enter values of a,b,c:"
read a b c
echo "a=$a b=$b c=$c"
echo a b c
```

```
sum=$((a + b + c))
echo "Sum is $sum"

if [ $a -gt $b ]
then
   echo "$a is greater"
else
   echo "$b is greater"
fi

echo "System Security Lab"

i=1
while [ $i -le 5 ]
do
   echo $i
   i=$((i + 1))
done

j=1
until [ $j -gt 5 ]
do
   echo $j
   j=$((j + 1))
done

for((k=0;k<10;k++))
do
   echo $k
done
```

bash f3.sh: Even/Odd, Positive-Negative, Fibonacci Series:

```
┌──(kali㉿kali)-[~/Desktop]
└─$ bash f3.sh
Even check of numbers, positive/negative, and Fibonacci
Enter a number to be checked:
2
Even
Enter a number:
50
Number is positive
Enter number of elements to be printed in Fibonacci series:
3
0
1
1
```

////

f3.sh>

```
echo "Even check of numbers, positive/negative, and Fibonacci"

# Even check
echo "Enter a number to be checked:"
read even
if [ $((even % 2)) -ne 0 ]
then
   echo "Not even"
else
   echo "Even"
fi

# Positive/Negative check
echo "Enter a number:"
read num
if [ $num -eq 0 ]
then
   echo "Number is zero"
elif [ $num -gt 0 ]
then
   echo "Number is positive"
else
   echo "Number is negative"
fi

# Fibonacci Series
echo "Enter number of elements to be printed in Fibonacci series:"
read eleno

a1=0
a2=1

echo "$a1"
echo "$a2"

i=3
while [ $i -le $eleno ]
do
   next=$((a1 + a2))
   echo "$next"
   a1=$a2
   a2=$next
   i=$((i + 1))
done
```

////

Environmental variables:

```
┌──(kali㉿kali)-[~/Desktop]
└─$ env
SHELL=/usr/bin/zsh
SESSION_MANAGER=local/kali:@/tmp/.ICE-unix/1679,unix/kali:/tmp/.ICE-unix/1679
WINDOWID=0
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg
XDG_SESSION_PATH=/org/freedesktop/DisplayManager/Session0
XDG_MENU_PREFIX=xfce-
POWERSHELL_UPDATECHECK=Off
LANGUAGE=
LESS_TERMCAP_se=
LESS_TERMCAP_so=
POWERSHELL_TELEMETRY_OPTOUT=1
SSH_AUTH_SOCK=/tmp/ssh-Nhf51lSvFUIX/agent.1781
DOTNET_CLI_TELEMETRY_OPTOUT=1
XDG_CONFIG_HOME=/home/kali/.config
NMAP_PRIVILEGED=
DESKTOP_SESSION=lightdm-xsession
SSH_AGENT_PID=1782
GTK_MODULES=gail:atk-bridge
XDG_SEAT=seat0
PWD=/home/kali/Desktop
XDG_SESSION_DESKTOP=lightdm-xsession
LOGNAME=kali
QT_QPA_PLATFORMTHEME=qt5ct
XDG_SESSION_TYPE=x11
XAUTHORITY=/home/kali/.Xauthority
XDG_GREETER_DATA_DIR=/var/lib/lightdm/data/kali
COMMAND_NOT_FOUND_INSTALL_PROMPT=1
HOME=/home/kali
LANG=en_US.UTF-8
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31
;01:mi=00:su=37;41:sg=30;43:ca=00:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.7z=01;31:*.ace=01;31:*.alz=01;
31:*.apk=01;31:*.arc=01;31:*.arj=01;31:*.bz=01;31:*.bz2=01;31:*.cab=01;31:*.cpio=01;31:*.crate=01;31:*.deb
=01;31:*.drpm=01;31:*.dwm=01;31:*.dz=01;31:*.ear=01;31:*.egg=01;31:*.esd=01;31:*.gz=01;31:*.jar=01;31:*.l
ha=01;31:*.lrz=01;31:*.lz=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.lzo=01;31:*.pyz=01;31:*.rar=01;31:*.r
pm=01;31:*.rz=01;31:*.sar=01;31:*.swm=01;31:*.t7z=01;31:*.tar=01;31:*.taz=01;31:*.tbz=01;31:*.tbz2=01;31:
*.tgz=01;31:*.tlz=01;31:*.txz=01;31:*.tz=01;31:*.tzo=01;31:*.tzst=01;31:*.udeb=01;31:*.war=01;31:*.whl=01;3
1:*.wim=01;31:*.xz=01;31:*.z=01;31:*.zip=01;31:*.zoo=01;31:*.zst=01;31:*.avif=01;35:*.jpg=01;35:*.jpeg=01;3
5:*.jxl=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.
tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;
35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm=01;35:*.webp=01
;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:
*.asf=01;35:*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xc
```

```
f=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:*~=00;90:*#=00;90:*.bak=00;90:*.crdownload=00;90:*.dpkg-dist=00;90:*.dpkg-new=00;90:*.dpkg-old=00;90:*.dpkg-tmp=00;90:*.old=00;90:*.orig=00;90:*.part=00;90:*.rej=00;90:*.rpmnew=00;90:*.rpmorig=00;90:*.rpmsave=00;90:*.swp=00;90:*.tmp=00;90:*.ucf-dist=00;90:*.ucf-new=00;90:*.ucf-old=00;90::ow=30;44:
XDG_CURRENT_DESKTOP=XFCE
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
XDG_CACHE_HOME=/home/kali/.cache
XDG_SESSION_CLASS=user
TERM=xterm-256color
LESS_TERMCAP_mb=
LESS_TERMCAP_me=
LESS_TERMCAP_md=
USER=kali
COLORFGBG=15;0
DISPLAY=:0.0
LESS_TERMCAP_ue=
SHLVL=2
LESS_TERMCAP_us=
XDG_VTNR=7
XDG_SESSION_ID=2
XDG_RUNTIME_DIR=/run/user/1000
QT_AUTO_SCREEN_SCALE_FACTOR=0
XDG_DATA_DIRS=/usr/share/xfce4:/usr/local/share/:/usr/share/:/usr/share
PATH=/home/kali/.local/bin:/usr/local/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/home/kali/.dotnet/tools
GDMSESSION=lightdm-xsession
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
OLDPWD=/home/kali/Desktop
_=/usr/bin/env
```

f4.sh> Switch Case

```
echo "Even check of numbers, positive/negative, and Fibonacci"

echo "1. Addition 2. Subtraction 3. Multiplication 4. Division"
echo "Enter option:"
read option
echo "option = $option"
echo "Enter two numbers"
read a b

case $option in
1) echo "Case 1:"
sum=$((a + b ))
echo "Addition=$sum";;
```

```
2) echo "Case 2:"
sub=$((a - b ))
echo "Subtraction=$sub";;
3) echo "Case 3:"
prod=$((a * b ))
echo "Multiplication=$prod";;
4) echo "Case 4:"
div=$((a / b ))
echo "Division=$div";;
*) echo "Invalid case";;
esac
```

////

```
┌──(kali㉿kali)-[~/Desktop]
└─$ bash f4.sh
Even check of numbers, positive/negative, and Fibonacci
1. Addition 2. Subtraction 3. Multiplication 4. Division
Enter option:
1
option = 1
Enter two numbers
10 20
Case 1:
Addition=30
```

/////

Lab-5: 17th Oct 2025 - Patterns in Shell Scripting and CRC check

Patterns in Shell Scripting:

f5.sh>

```
#!/bin/bash

# Pattern 1:
# * * * *
# * * * *
# * * * *
# * * * *
```

```
echo "Pattern 1:"
for ((i=0; i<4; i++)); do
    echo "* * * *"
done
echo ""

# Pattern 2:
# *
# * *
# * * *
# * * * *
echo "Pattern 2:"
for ((i=1; i<=4; i++)); do
    for ((j=1; j<=i; j++)); do
        echo -n "* "
    done
    echo ""
done
echo ""

# Pattern 3:
# *
# * * *
# * * * * *
# * * * * * * *
echo "Pattern 3:"
count=1
for ((i=1; i<=4; i++)); do
    for ((j=1; j<=count; j++)); do
        echo -n "* "
    done
    echo ""
    count=$((count + 2))
done
echo ""

# Pattern 4:
# * * * *
# * * *
# * *
# *
echo "Pattern 4:"
for ((i=4; i>=1; i--)); do
    for ((j=1; j<=i; j++)); do
        echo -n "* "
    done
    echo ""
done
echo ""

# Pattern 5:
# 1
```

```
# 1 2
# 1 2 3
# 1 2 3 4
echo "Pattern 5:"
for ((i=1; i<=4; i++)); do
    for ((j=1; j<=i; j++)); do
        echo -n "$j "
    done
    echo ""
done
echo ""

# Pattern 6:
# 1
# 2 3
# 4 5 6
# 7 8 9 10
echo "Pattern 6:"
count=1
for ((i=1; i<=4; i++)); do
    for ((j=1; j<=i; j++)); do
        echo -n "$count "
        count=$((num + 1))
    done
    echo ""
done
```

////

CRC check

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#define N strlen(gen_poly)

char data[28];          // Data to be transmitted
char check_value[28];   // Check value (CRC)
char gen_poly[10];      // Generator polynomial
int data_length, i, j;

// Function to XOR the check_value with the generator polynomial
void XOR() {
    for (j = 1; j < N; j++) {
        check_value[j] = (check_value[j] == gen_poly[j]) ? '0' : '1';
    }
```

```
}

// Function to perform the CRC division
void crc() {
    for (i = 0; i < N; i++) {
        check_value[i] = data[i];
    }

    do {
        if (check_value[0] == '1') {
            XOR();
        }
        for (j = 0; j < N - 1; j++) {
            check_value[j] = check_value[j + 1];
        }
        check_value[j] = data[i++];
    } while (i <= data_length + N - 1);
    check_value[N - 1] = '\0';  // Null-terminate the CRC check value
}

// Function to check if the received data has an error
void receiver() {
    printf("Enter the received data: ");
    scanf("%27s", data);  // Prevent buffer overflow

    // Validate received data to ensure it contains only '0' or '1'
    for (i = 0; i < strlen(data); i++) {
        if (data[i] != '0' && data[i] != '1') {
            printf("Invalid input. Please enter a valid binary string.\n");
            return;
        }
    }

    printf("Data received: %s\n", data);
    crc();

    // Check if there is an error in the received data
    for (i = 0; (i < N - 1) && (check_value[i] != '1'); i++);
    if (i < N - 1) {
        printf("\nError detected\n\n");
    } else {
        printf("\nNo error detected\n\n");
    }
}

int main() {
    printf("Enter data to be transmitted: ");
    scanf("%27s", data);  // Prevent buffer overflow

    // Validate the input to ensure it contains only '0' or '1'
    for (i = 0; i < strlen(data); i++) {
        if (data[i] != '0' && data[i] != '1') {
```

```
            printf("Invalid input. Please enter a valid binary string.\n");
            return 1;
        }
    }

    printf("\nEnter the generating polynomial: ");
    scanf("%9s", gen_poly);  // Prevent buffer overflow

    // Validate the generating polynomial
    for (i = 0; i < strlen(gen_poly); i++) {
        if (gen_poly[i] != '0' && gen_poly[i] != '1') {
            printf("Invalid generating polynomial. Please enter a valid binary string.\n");
            return 1;
        }
    }

    data_length = strlen(data);

    // Pad the data with n-1 zeros
    for (i = data_length; i < data_length + N - 1; i++) {
        data[i] = '0';
    }
    printf("\nData padded with n-1 zeros: %s\n", data);

    crc();

    // Display the CRC or check value
    printf("\nCRC or check value is: %s", check_value);

    // Append the check value to the original data to form the final data to be sent
    for (i = data_length; i < data_length + N - 1; i++) {
        data[i] = check_value[i - data_length];
    }
    printf("\nFinal data to be sent: %s\n", data);

    receiver();

    return 0;
}
```

O/P:

```
Enter data to be transmitted: 100100
Enter the generating polynomial: 1101

Data padded with n-1 zeros: 100100000
CRC or check value is: 001
Final data to be sent: 100100001
```

Enter the received data: 100100001
Data received: 100100001

No error detected

////

Lab-6: 7th Nov 2025 - Linear Search

Linear Search Bash

```
echo enter the number of elements
read n
echo enter the array elements
for ((i=1;i<=n;i++))
do
read a[$i]
done
echo enter the element to be searched
read item
j=1
while [ $j -lt $n  -a $item -ne ${a[$j]} ]
do
j=$(($j+1))
done
if [ $item -eq ${a[$j]} ]
then
echo $item is present at the location $j
else
echo $item is not present
fi
```

O/P:

```
enter the number of elements
3
enter the array elements
1
2
3
```

enter the element to be searched
3
3 is present at the location 3

////

Commands:

- hostname
- whoami
- uname
- logname
- set computername
- man set
- finger asas
- cat /etc/issue
- cat /etc/*-release
- ifconfig
- ifconfig /allcompartments /all
- route -n
- finger ns3
- route print

////