

B.M.S. COLLEGE OF ENGINEERING BENGALURU
Autonomous Institute, Affiliated to VTU



Lab Record

Computer Networks – 23CS5PCCON

Submitted in partial fulfillment for the 5th Semester Laboratory

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by:

Deepthi M

(1BM23CS088)

Department of Computer Science and Engineering
B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
August 2025-December 2025

B.M.S. COLLEGE OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING



CERTIFICATE

This is to certify that the Computer Networks (23CS5PCCON) laboratory has been carried out by Deepthi M (1BM23CS088) during the 5th Semester August 2025-December 2025

Signature of the Faculty Incharge:

Sarala D V
Assistant Professor
Department of Computer Science and Engineering
B.M.S. College of Engineering, Bangalore

Table of Contents

PART - A	
Serial No.	Name of Experiment
1.	Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message.
2.	Configure DHCP within a LAN and outside LAN.
3.	Configure Web Server, DNS within a LAN.
4.	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.
5.	Configure default route, static route to the Router.
6.	Configure RIP routing Protocol in Routers.
7.	Configure OSPF routing protocol.
8.	To construct a VLAN and make the PC's communicate among a VLAN.
9.	To construct a WLAN and make the nodes communicate wirelessly.
10.	Demonstrate the TTL/ Life of a Packet.
11.	To understand the operation of TELNET by accessing the router in server room from a PC in IT office.
12.	To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP).

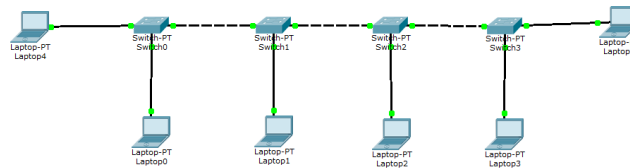
PART – B	
Serial No.	Name of Experiment
1.	Write a program for congestion control using Leaky bucket algorithm.
2.	Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.
3.	Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.
4.	Write a program for error detecting code using CRC-CCITT (16-bits).

PART - A

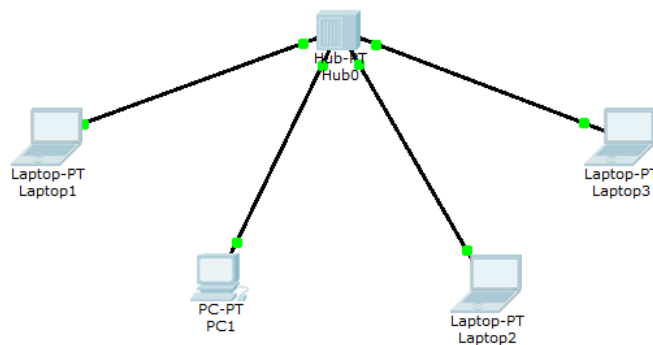
Program 1: Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message.

Network diagram:

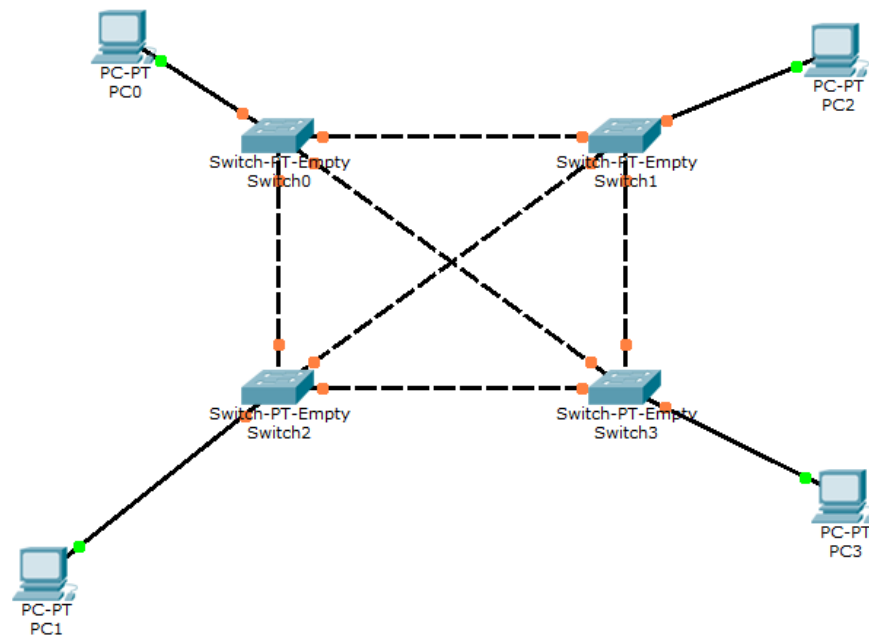
Bus



Hub

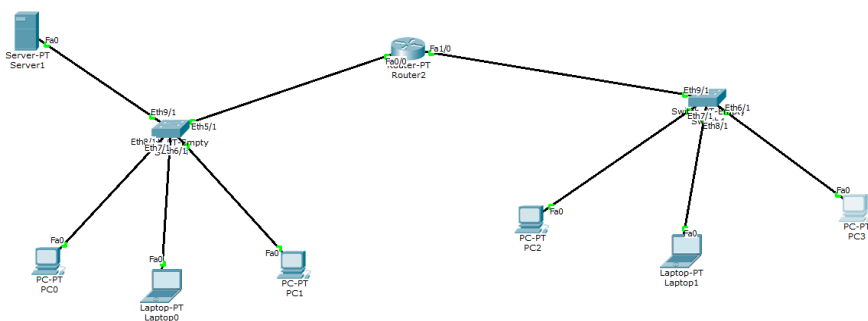


Mesh

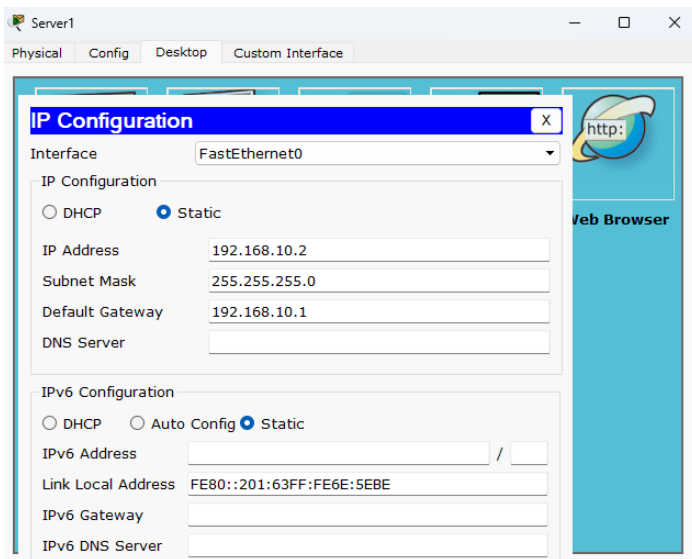


Program 2: Configure DHCP within a LAN and outside LAN.

Network diagram:



Configuration:



```

PT 1001 (PTSC2005) processor (revision 0x200) with 60416K/5120K bytes of memory
.
Processor board ID FT0123 (0123)
PT2005 processor: part number 0, mask 01
Bridging software.
X.25 software, Version 3.0.0.
4 FastEthernet/IEEE 802.3 interface(s)
3 Low-speed serial(sync/async) network interface(s)
32K bytes of non-volatile configuration memory.
63488K bytes of ATA CompactFlash (Read/Write)

--- System Configuration Dialog ---

Continue with configuration dialog? [yes/no]: no

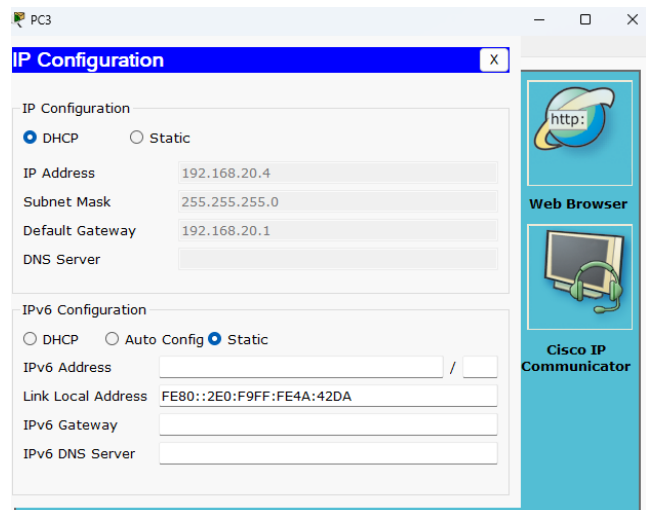
Press RETURN to get started!

Router>enable
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#int Fa0/0
Router(config-if)#ip address 192.168.10.1 255.255.255.0
Router(config-if)#ip helper-address 192.168.10.2
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up
do write memory
Building configuration...
[OK]
Router(config-if)#exit
Router(config)#int Fa1/0
Router(config-if)#ip address 192.168.20.1 255.255.255.0
Router(config-if)#ip helper-address 192.168.10.2
Router(config-if)#no shutdown

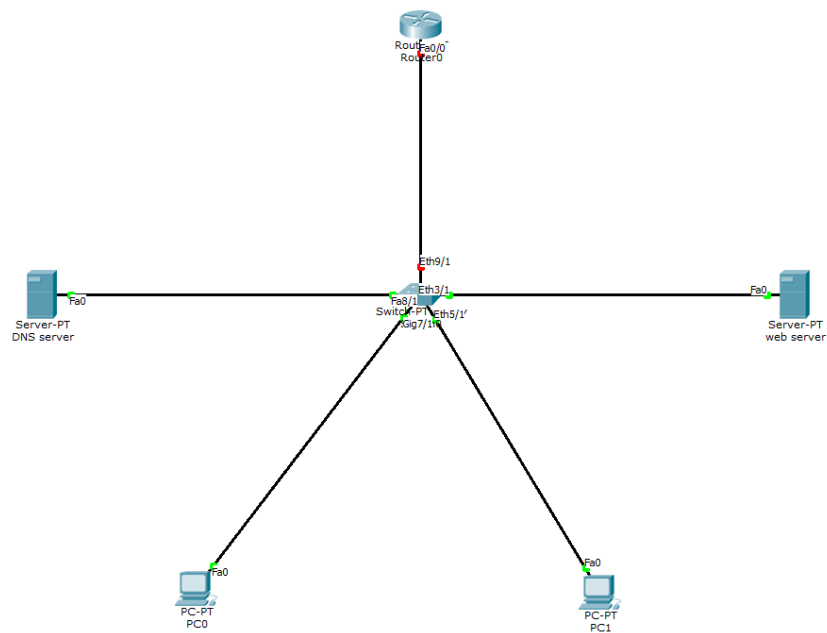
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet1/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to up
do write memory
Building configuration...
[OK]
Router(config-if)#exit
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
write memory
Building configuration...
[OK]
Router#IP-4-DUPADDR: Duplicate address 192.168.10.1 on FastEthernet0/0, sourced
by 000A.4166.1664

```

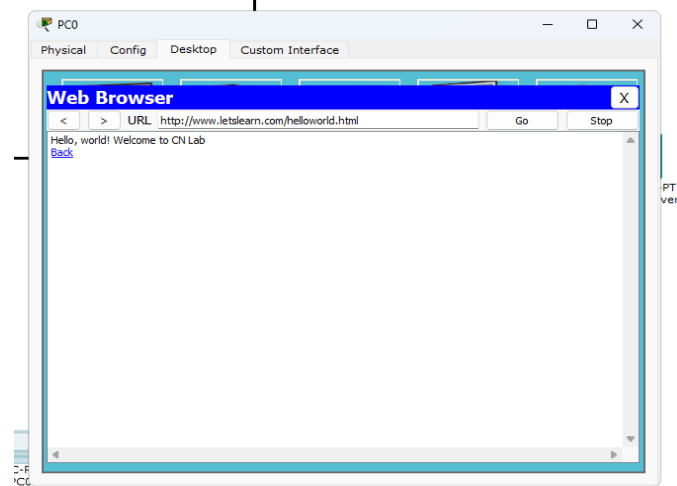


Program 3: Configure Web Server, DNS within a LAN.

Network diagram:

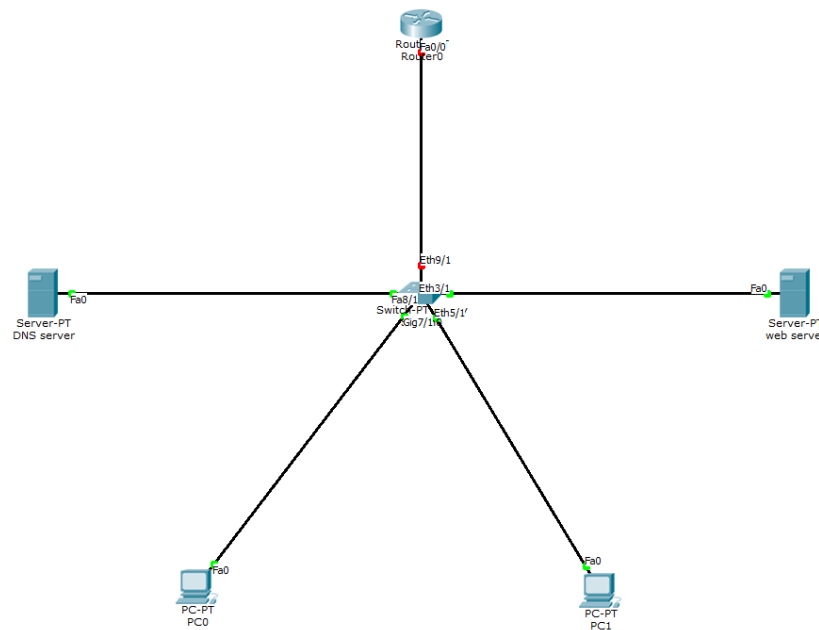


Configuration:



Program 4: Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.

Network diagram:



Configuration:

```
PC>ping 192.168.1.101
Pinging 192.168.1.101 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

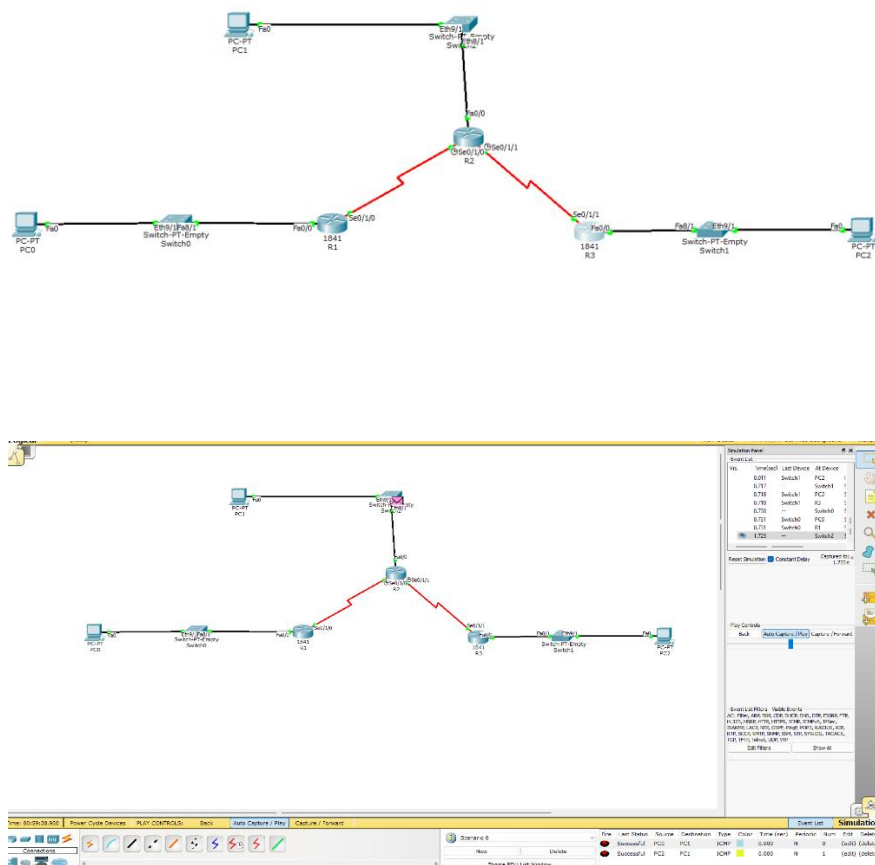
Ping statistics for 192.168.1.101:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

PC>ping 192.168.1.6
Pinging 192.168.1.6 with 32 bytes of data:
Reply from 192.168.1.6: bytes=32 time=5ms TTL=128
Reply from 192.168.1.6: bytes=32 time=0ms TTL=128
Reply from 192.168.1.6: bytes=32 time=0ms TTL=128
Reply from 192.168.1.6: bytes=32 time=0ms TTL=128

Ping statistics for 192.168.1.6:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 5ms, Average = 1ms
```


Program 5: Configure default route, static route to the Router.

Network diagram:



Configuration:

PC0

IP Configuration

IP Configuration

☐ DHCP ☒ Static

IP Address

Subnet Mask

Default Gateway

DNS Server

IPv6 Configuration

☐ DHCP ☐ Auto Config ☒ Static

IPv6 Address

Link Local Address

IPv6 Gateway

IPv6 DNS Server

PC1

IP Configuration

IP Configuration

☐ DHCP ☒ Static

IP Address

Subnet Mask

Default Gateway

DNS Server

IPv6 Configuration

☐ DHCP ☐ Auto Config ☒ Static

IPv6 Address

Link Local Address

IPv6 Gateway

IPv6 DNS Server

PC2

IP Configuration

IP Configuration

☐ DHCP ☒ Static

IP Address

Subnet Mask

Default Gateway

DNS Server

IPv6 Configuration

☐ DHCP ☐ Auto Config ☒ Static

IPv6 Address

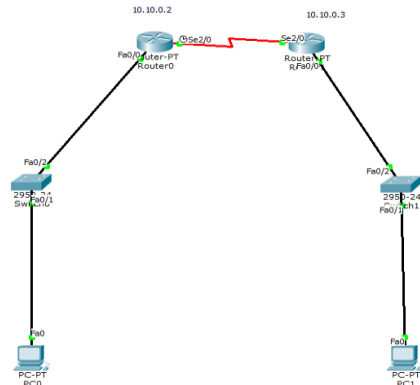
Link Local Address

IPv6 Gateway

IPv6 DNS Server

Program 6: Configure RIP routing Protocol in Routers.

Network diagram:



Configuration:

PC0 IP Configuration

IP Configuration: ☐ DHCP ☒ Static

IP Address: 192.168.1.2

Subnet Mask: 255.255.255.0

Default Gateway: 192.168.1.1

DNS Server:

IPv6 Configuration: ☐ DHCP ☐ Auto Config ☒ Static

IPv6 Address: /

Link Local Address: FE80::20B:BEFF:FE9B:6648

IPv6 Gateway:

IPv6 DNS Server:

Router0 FastEthernet0/0 Configuration

Port Status: ☒ On

Bandwidth: ☐ 10 Mbps ☒ 100 Mbps

Duplex: ☒ Full Duplex ☐ Half Duplex

MAC Address: 0009.7CEA.A6D0

IP Address: 192.168.1.1

Subnet Mask: 255.255.255.0

Tx Ring Limit: 10

Equivalent IOS Commands

```
Router#configure terminal
Router(config)#interface FastEthernet0/0
Router(config-if)#
```

Router0 Global Settings

Display Name: Router0

Hostname: Router

NVRAM: Erase Save

Startup Config: Load... Export...

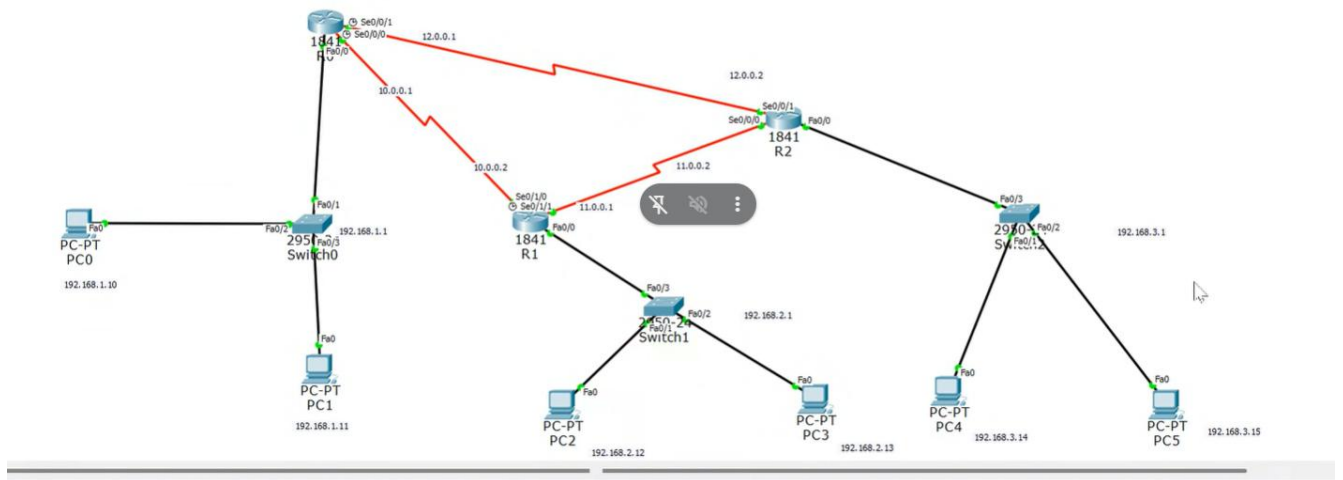
Running Config: Merge... Export...

Equivalent IOS Commands

```
Router(config)#router rip
Router(config)#router rip
Router(config-router)#
Router(config-router)#exit
Router(config)#router rip
```

Program 7: Configure OSPF routing protocol.

Network diagram:



Configuration:

```
* 2019/06/06 10:00:00
Router(config)#exit
Router#
$SYS-5-CONFIG_I: Configured from console by console
enable
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#network 192.168.1.0 0.0.0.255 area0
% Invalid input detected at '^' marker.

Router(config-router)#network 192.168.1.1 0.0.0.255 area0
% Invalid input detected at '^' marker.

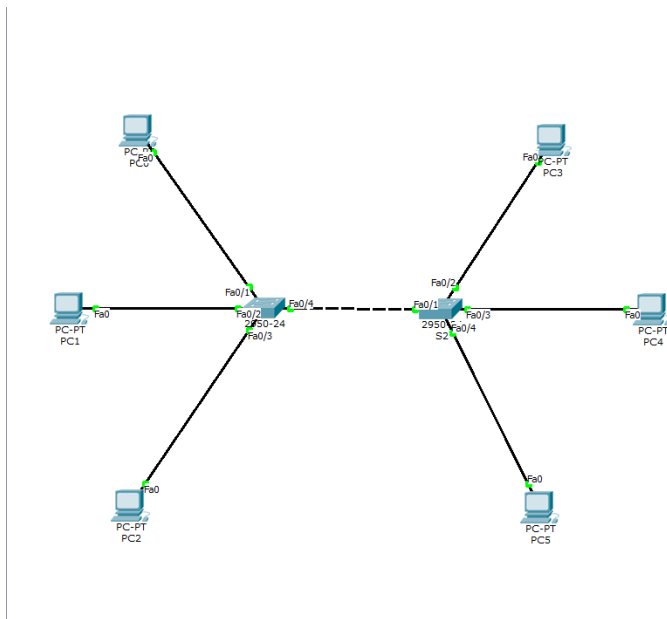
Router(config-router)#exit
Router(config)#exit
Router#
$SYS-5-CONFIG_I: Configured from console by console
enable
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#
Router(config-router)#
```

```
!LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#ip address 11.0.0.1 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
!LINE-5-CHANGED: Interface Serial2/0, changed state to up
!LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up
Router(config)#enable
% Incomplete command.
Router(config)#exit
Router#
$SYS-5-CONFIG_I: Configured from console by console
enable
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#network 192.168.2.0 0.0.0.255 area 0
Router(config-router)#network 10.0.0.0 0.255.255.255 area 0
Router(config-router)#network 11.0.0.0 0.255.255.255 area 0
% Invalid input detected at '^' marker.

Router(config-router)#network 11.0.0.0 0.255.255.255 area 0
Router(config-router)#exit
Router#
$SYS-5-CONFIG_I: Configured from console by console
we
Building configuration...
[OK]
Router#
00:10:31: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.3.1 on Serial2/0 from LOADING t
o FULL, Loading Done
```

Program 8: To construct a VLAN and make the PC's communicate among a VLAN.

Network diagram:



Configuration:

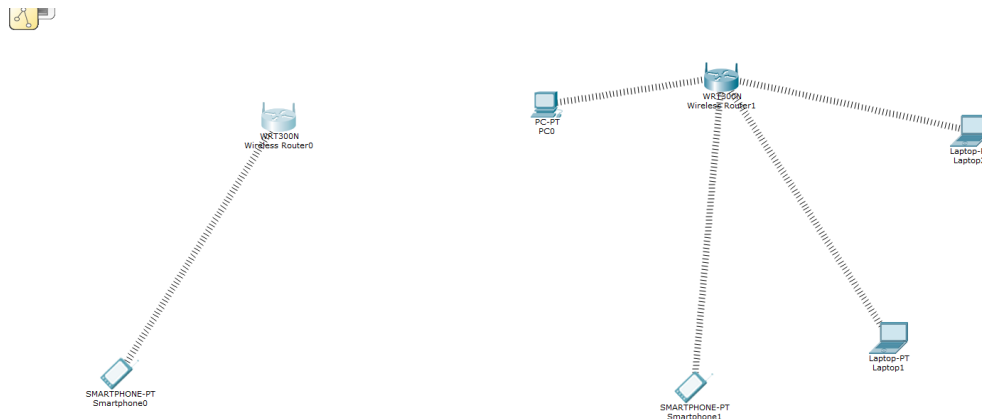
```

S2
Physical Config CLI
IOS Command Line Interface
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/4, changed state t
o up
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state t
o down
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/1, changed state t
o up
Switch(enable)
Switch(conf t)
Enter configuration commands, one per line. End with CTRL/Z.
Switch(config)#int fa0/2
Switch(config-if)#switchport access vlan 10
% Access VLAN does not exist. Creating vlan 10
Switch(config-if)#int fa0/3
Switch(config-if)#switchport access vlan 20
% Access VLAN does not exist. Creating vlan 20
Switch(config-if)#int fa0/4
Switch(config-if)#switchport access vlan 30
% Access VLAN does not exist. Creating vlan 30
Switch(config-if)#int fa0/1
Switch(config-if)#switchport mode trunk
Switch(config-if)#
  
```

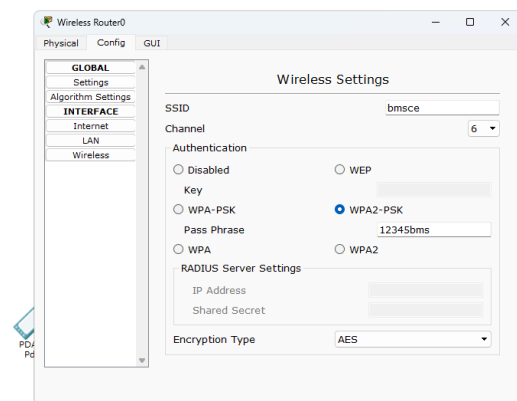
Fire	Last Status	Source	Destination	Type	Color	Time (sec)	Periodic	Num	Edit	Delete
	Failed	PC0	PC4	ICMP		2.028	N	0	(edit)	(delete)

Program 9: To construct a WLAN and make the nodes communicate wirelessly.

Network diagram:



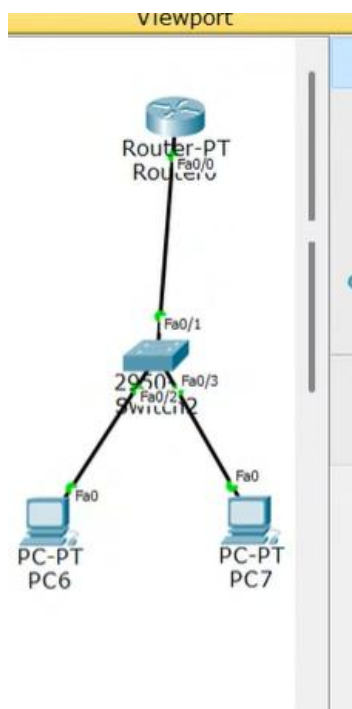
Configuration:



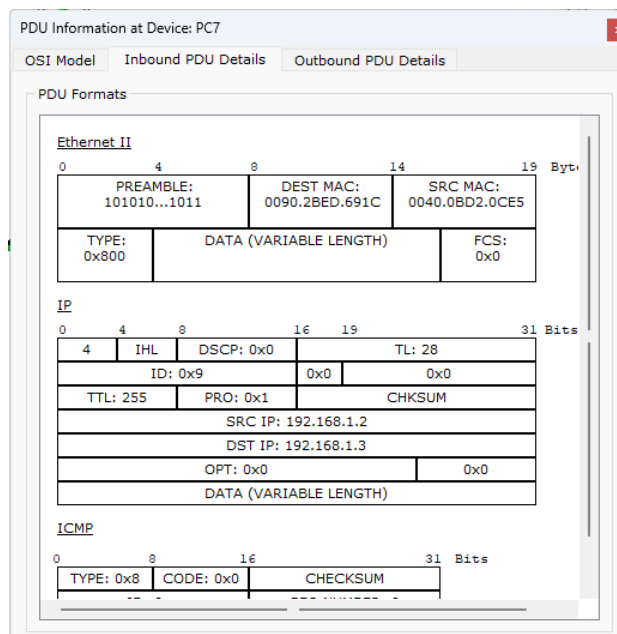
Realtime										
Fire	Last Status	Source	Destination	Type	Color	Time (sec)	Periodic	Num	Edit	Delete
	Successful	Pda0	Laptop1	ICMP		0.000	N	0	(edit)	(delete)

Program 10: Demonstrate the TTL/ Life of a Packet.

Network diagram:

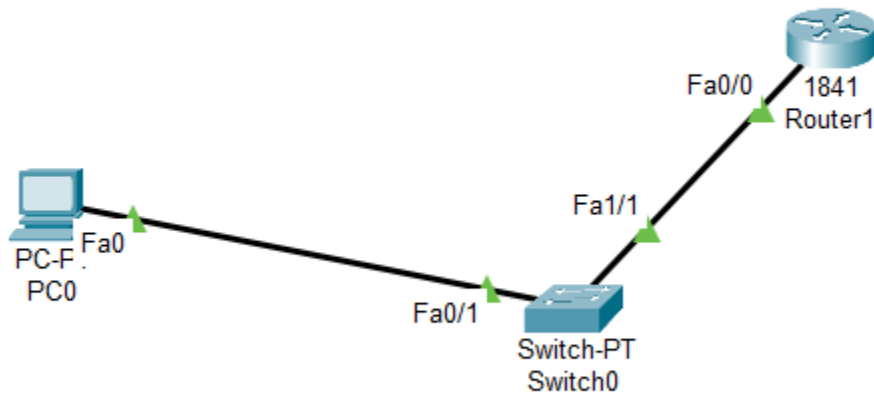


Configuration:



Program 11: To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

Network diagram:



Configuration:

```
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#enable secret rp
Router(config)#int Fa 0/0
Router(config-if)#ip address 192.168.1.1 255.255.255.0
^
% Invalid input detected at '^' marker.

Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

Router(config-if)#
Router(config-if)#line vty
^
% Invalid input detected at '^' marker.

Router(config-if)#line vty
^
% Invalid input detected at '^' marker.

Router(config-if)#line vty
^
% Invalid input detected at '^' marker.

Router(config-if)#exit
Router(config)#line vty
% Incomplete command.
Router(config)#login
% Incomplete command.
Router(config)#enable secret rp
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
enable
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#enable secret rp
```


Program 12: To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP).

Network diagram and configuration:

The screenshot shows the Cisco Packet Tracer interface with several windows open:

- ARP Table for Server0:**

IP Address	Hardware Address	Interface
192.168.1...	0001.434B.6550	FastEthernet0
- ARP Table for PC0:**

IP Address	Hardware Address	Interface
192.168.1...	0060.2FE0.AC...	FastEthernet0
192.168.1...	0060.47D7.A2...	FastEthernet0
192.168.1...	0001.C9A7.A7...	FastEthernet0
- PDU Information at Device: PC1:**

Ethernet II

PREAMBLE:	DEST MAC:	SRC MAC:
101010...	101...	101...

OSI Model Inbound PDU Details

PDU Formats

TYP	DATA (VARIABLE LENGTH)	FCS
E:		

ARP

0	8	16	31	bits
HARDWARE		PROTOCOL		
HLEN: PLEN:		OPCODE: 0x2		
SOURCE MAC: 0060.47D7.A241		SOURCE IP (32)		
192.168.11.3		SOURCE IP (32)		
TARGET MAC: 0060.2FE0.AC8A		TARGET IP: 192.168.11.2 (32)		
- Simulation Panel:**

Event List

Vis.	Time(sec)	Last De	At Dev	Type	Info
	0.002	Switch0	PC2	ARP	
	0.002	Switch0	Serve...	ARP	
	0.003	PC2	Switch...	ARP	
	0.004	Switch0	PC1	ARP	
	0.004	--	PC1	ICMP	

Reset Simulation ☒ Constant Delay Captured to: 0.004 s

Play Controls: Back Auto Capture / Play Capture / Forward

Event List Filters - Visible Events

ACL Filter, ARP, BGP, CD, DHCP, DHCPv6, DNS, DTP, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPsec, ISAKMP, LACP, NDP, NETFLOW, VTP, OSPF, OSPFv6, PAg, POP3, RADIUS, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP

Edit Filters Show All/None

The screenshot shows the Cisco Packet Tracer interface with the following windows:

- PC0 IP Configuration:**

IP Configuration

☐ DHCP ☒ Static

IP Address: 192.168.11.1

Subnet Mask: 255.255.255.0

Default Gateway:

DNS Server: 192.168.11.4

IPv6 Configuration

☐ DHCP ☐ Auto Config ☒ Static

IPv6 Address: /

Link Local Address: FE80::260:70FF:FE86:B1A2

IPv6 Gateway:

IPv6 DNS Server:
- Network Diagram:**

The diagram shows a central switch (Switch0) connected to three PCs (PC1, PC2, PC3). The switch is labeled "Switch0 (S2600)". The PCs are labeled "PC1 (C2950)", "PC2 (C2950)", and "PC3 (C2950)".
- ARP Table for PC0:**

IP Address	Hardware Address	Interface
192.168.1...	0060.2FE0.AC...	FastEthernet0

PART - B

Program 1: Write a program for congestion control using Leaky bucket algorithm.

Code: #include <stdio.h>

#include <string.h>

```
int min(int x, int y) {  
    if (x < y)  
        return x;  
    else  
        return y;  
}
```

```
int main() {  
    int drop = 0, mini, nsec, cap, count = 0, i, inp[25], process;  
    printf("Enter the bucket size:\n");  
    scanf("%d", &cap);  
    printf("Enter the processing rate:\n");  
    scanf("%d", &process);  
    printf("Enter the number of seconds you want to simulate:\n");  
    scanf("%d", &nsec);  
  
    for (i = 0; i < nsec; i++) {  
        printf("Enter the size of the packet entering at %d sec:\n", i + 1);  
        scanf("%d", &inp[i]);  
    }  
  
    printf("\nSecond | Packet received | Packet sent | Packet left | Dropped\n");  
    printf("-----\n");  
  
    for (i = 0; i < nsec; i++) {  
        count += inp[i];  
        if (count > cap) {  
            drop = count - cap;  
            count = cap;  
        }  
        printf("%d", i + 1);  
        printf("\t%d", inp[i]);  
        mini = min(count, process);
```

```

    printf("\t\t%d", mini);
    count -= mini;
    printf("\t\t%d", count);
    printf("\t\t%d\n", drop);
    drop = 0;
}

while (count != 0) {
    if (count > cap) {
        drop = count - cap;
        count = cap;
    }
    i++;
    printf("%d", i);
    printf("\t0");
    mini = min(count, process);
    printf("\t\t%d", mini);
    count -= mini;
    printf("\t\t%d", count);
    printf("\t\t%d\n", drop);
}

return 0;
}
OUTPUT:

```

```

Enter the bucket size:
5
Enter the processing rate:
2
Enter the number of seconds you want to simulate:
3
Enter the size of the packet entering at 1 sec:
5
Enter the size of the packet entering at 2 sec:
2
Enter the size of the packet entering at 3 sec:
3

```

Second	Packet received	Packet sent	Packet left	Dropped
1	5	2	3	0
2	2	2	3	0
3	3	2	3	1
4	0	2	1	0
5	0	1	0	0

Program 2: Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.

Code:

Server:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
int main(int argc, char *argv[])
{
    int sockfd, newsockfd, portno, n;
    char buffer[256], line[2000], filedata[20000];
    struct sockaddr_in serv, cli;
    socklen_t len;
    FILE *fp;
    if (argc < 2) {
        printf("Error: No port number provided.\nUsage: ./server <port>\n");
        exit(1);
    }
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd < 0) {
        perror("Socket creation failed");
        exit(1);
    }
```

```

}

memset(&serv, 0, sizeof(serv));
portno = atoi(argv[1]);
serv.sin_family = AF_INET;
serv.sin_addr.s_addr = INADDR_ANY;
serv.sin_port = htons(portno);
if (bind(sockfd, (struct sockaddr *)&serv, sizeof(serv)) < 0) {
    perror("Bind failed");
    exit(1);
}
listen(sockfd, 5);
printf("Server: Waiting for connection...\n");
len = sizeof(cli);
newsockfd = accept(sockfd, (struct sockaddr *)&cli, &len);
if (newsockfd < 0) {
    perror("Accept failed");
    exit(1);
}

memset(buffer, 0, sizeof(buffer));
n = read(newsockfd, buffer, sizeof(buffer) - 1);
if (n < 0) {
    perror("Error reading file name");
    exit(1);
}
printf("Server received file request: %s\n", buffer);
fp = fopen(buffer, "r");

```

```

if (fp == NULL) {
    printf("Server: File not found.\n");
    write(newsockfd, "File not found", 15);
    close(newsockfd);
    close(sockfd);
    return 0;
}
printf("Server: File found. Reading...\n");
memset(filedata, 0, sizeof(filedata));
while (fgets(line, sizeof(line), fp) != NULL) {
    strcat(filedata, line);
}
fclose(fp);
n = write(newsockfd, filedata, strlen(filedata));
if (n < 0)
    perror("Error writing to socket");
printf("Transfer complete.\n");

close(newsockfd);
close(sockfd);
return 0;
}

```

Client:

```

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>

```

```

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
int main(int argc, char *argv[])
{
    int sockfd, portno, n;
    char filename[256], filedata[20000];
    struct sockaddr_in serv;
    if (argc < 2) {
        printf("Error: No port number provided.\nUsage: ./client <port>\n");
        exit(1);
    }
    sockfd = socket(AF_INET, SOCK_STREAM, 0);
    if (sockfd < 0) {
        perror("Socket creation failed");
        exit(1);
    }
    memset(&serv, 0, sizeof(serv));
    portno = atoi(argv[1]);
    serv.sin_family = AF_INET;
    serv.sin_port = htons(portno);
    serv.sin_addr.s_addr = INADDR_ANY; // connecting to same machine
    if (connect(sockfd, (struct sockaddr *)&serv, sizeof(serv)) < 0) {
        perror("Server not responding");
        exit(1);
    }
    printf("Enter file path: ");

```

```

scanf("%s", filename);
n = write(sockfd, filename, strlen(filename));
if (n < 0)
    perror("Error writing filename");
memset(filedata, 0, sizeof(filedata));
n = read(sockfd, filedata, sizeof(filedata) - 1);
if (n < 0)
    perror("Error reading data from server");
printf("\n--- File Content ---\n%s\n", filedata);
close(sockfd);
return 0;
}

```

Output:

```
$ cc socketserver.c
```

```
$ ./a.out 1025
```

server:

waiting for connection

server received:/home/aps/cse.txt

server:/home/aps/cse.txt found

opening and reading..

reading..

..reading complete

transfer complete

```
$ cc socketclient.c
```

```
$ ./a.out 1025
```

Enter the file with complete path

/home/aps/cse.txt

Reading..

..

client: display content of /home/aps/cse.txt

..

Welcome to the CSE department.....

2)

\$ cc fserver.c

\$./a.out

error:no port no

usage:

/server port no

\$ cc fclient.c

\$./a.out

Err:no port no.

usage:

./client portno

ex:./client 7777

Program 3: Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.

Code:

Server:

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

```
#include <stdlib.h>
```

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

```
#include <unistd.h>
```

```

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>

int main(int argc, char *argv[])
{
    int sockfd, portno, n;
    char buffer[1024];
    struct sockaddr_in servaddr, cliaddr;
    socklen_t len;

    if (argc < 2) {
        printf("Usage: ./server <port>\n");
        exit(1);
    }

    portno = atoi(argv[1]);

    sockfd = socket(AF_INET, SOCK_DGRAM, 0);
    if (sockfd < 0) {
        perror("Socket creation failed");
        exit(1);
    }

    memset(&servaddr, 0, sizeof(servaddr));
    memset(&cliaddr, 0, sizeof(cliaddr));

```

```

servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = INADDR_ANY;
servaddr.sin_port = htons(portno);

if (bind(sockfd, (struct sockaddr *)&servaddr, sizeof(servaddr)) < 0) {
    perror("Bind failed");
    exit(1);
}

printf("UDP Server: Waiting for data...\n");

len = sizeof(cliaddr);
memset(buffer, 0, sizeof(buffer));

n = recvfrom(sockfd, buffer, sizeof(buffer)-1, 0,
             (struct sockaddr *)&cliaddr, &len);
if (n < 0) {
    perror("Receive failed");
    exit(1);
}

printf("Received from client: %s\n", buffer);

// Echo back the same data
n = sendto(sockfd, buffer, strlen(buffer), 0,
           (struct sockaddr *)&cliaddr, len);
if (n < 0)

```

```
perror("Send failed");
```

```
close(sockfd);
```

```
return 0;
```

```
}
```

Client:

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

```
#include <stdlib.h>
```

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

```
#include <unistd.h>
```

```
#include <sys/types.h>
```

```
#include <sys/socket.h>
```

```
#include <netinet/in.h>
```

```
int main(int argc, char *argv[])
```

```
{
```

```
    int sockfd, portno, n;
```

```
    char buffer[1024];
```

```
    struct sockaddr_in servaddr;
```

```
    socklen_t len;
```

```
    if (argc < 2) {
```

```
        printf("Usage: ./client <port>\n");
```

```
        exit(1);
```

```
    }
```

```
    portno = atoi(argv[1]);
```

```

sockfd = socket(AF_INET, SOCK_DGRAM, 0);
if (sockfd < 0) {
    perror("Socket creation failed");
    exit(1);
}

memset(&servaddr, 0, sizeof(servaddr));

servaddr.sin_family = AF_INET;
servaddr.sin_port = htons(portno);
servaddr.sin_addr.s_addr = INADDR_ANY; // local machine

printf("Enter message to send: ");
scanf("%[^\n]", buffer);

len = sizeof(servaddr);

n = sendto(sockfd, buffer, strlen(buffer), 0,
           (struct sockaddr *)&servaddr, len);
if (n < 0) {
    perror("Send failed");
    exit(1);
}

memset(buffer, 0, sizeof(buffer));

```

```

n = recvfrom(sockfd, buffer, sizeof(buffer)-1, 0,
              (struct sockaddr *)&servaddr, &len);
if (n < 0) {
    perror("Receive failed");
    exit(1);
}

printf("Server replied: %s\n", buffer);
close(sockfd);
return 0;
}

```

Output:

1)

UDP Server: Waiting for data...

Received from client: Hello UDP Server!

2)

Enter message to send: Hello UDP Server!

Server replied: Hello UDP Server!

Program 4: Write a program for error detecting code using CRC-CCITT (16-bits).

Code: #include <stdio.h>

#include <string.h>

#include <stdlib.h>

```

int main() {
    char rem[50], a[50], s[50], c, msj[50], gen[30];
    int i, genlen, t, j, flag = 0, k, n;

    printf("Enter the generator polynomial: ");
    fgets(gen, sizeof(gen), stdin);
    gen[strcspn(gen, "\n")] = '\0';
    printf("Generator polynomial is CRC: %s\n", gen);

    genlen = strlen(gen);

```

```

k = genlen - 1;

printf("Enter the message: ");
n = 0;
while ((c = getchar()) != '\n') {
    msj[n] = c;
    n++;
}
msj[n] = '\0';

for (i = 0; i < n; i++) {
    a[i] = msj[i];
}
for (i = 0; i < k; i++) {
    a[n + i] = '0';
}
a[n + k] = '\0';

printf("\nMessage polynomial appended with zeros:\n");
puts(a);

for (i = 0; i < n; i++) {
    if (a[i] == '1') {
        t = i;
        for (j = 0; j <= k; j++) {
            a[t] = (a[t] == gen[j]) ? '0' : '1';
            t++;
        }
    }
}

for (i = 0; i < k; i++) {
    rem[i] = a[n + i];
}
rem[k] = '\0';

printf("The checksum appended:\n");
puts(rem);
printf("\nThe message with checksum appended:\n");
for (i = 0; i < n; i++) {
    a[i] = msj[i];
}
for (i = 0; i < k; i++) {
    a[n + i] = rem[i];
}
a[n + k] = '\0';
puts(a);

n = 0;
printf("Enter the received message: ");

```

```

while ((c = getchar()) != '\n') {
    s[n] = c;
    n++;
}
s[n] = '\0';
for (i = 0; i < n; i++) {
    if (s[i] == '1') {
        t = i;
        for (j = 0; j <= k; j++, t++) {
            s[t] = (s[t] == gen[j]) ? '0' : '1';
        }
    }
}

for (i = 0; i < k; i++) {
    rem[i] = s[n + i];
}
rem[k] = '\0';
for (i = 0; i < k; i++) {
    if (rem[i] == '1') {
        flag = 1;
    }
}

if (flag == 0) {
    printf("Received polynomial is error-free.\n");
} else {
    printf("Received polynomial has an error.\n");
}

return 0;
}
OUTPUT:

```



```
Enter the generator polynomial: 10011
Generator polynomial is CRC: 10011
Enter the message: 1101011011

Message polynomial appended with zeros:
11010110110000
The checksum appended:
1110

The message with checksum appended:
11010110111110
Enter the received message: 11010110111110
Received polynomial is error-free.

Process returned 0 (0x0)   execution time : 20.417 s
Press any key to continue.
|
```