Centenary Celebrated Sharnbasveshwar Vidya Vardhaka Sangha's

ಶರಣಬಸವ ವಿಶ್ವವಿದ್ಯಾಲಯ

SHARNBASVA UNIVERSITY

A State Private University approved by Govt. of Karnataka vide Notification No. ED 144 URC 2016 dated 29-07-2017
Recognised by UGC under Section 2f vide No. F.8-29/2017 (CPP-I/PU), dated 20-12-2017 & AICTE, CoA, PCI New Delhi

**FACULTY OF ENGINEERING AND TECHNOLOGY (Co-Ed)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

LAB MANUAL

# Computer Network Laboratory

**(22CSL57)**

**2024-2025**

| Computer Networks Laboratory<br>[As per Choice Based Credit System (CBCS) scheme]<br>(Effective from the academic year 2022-2023)<br>SEMESTER – V | | | |
|---|---|---|---|
| **Subject Code** | 22CSL57 | **CIE Marks** | 50 |
| **Number of Laboratory Hours/Week** | 0:0:2 | **SEE Marks** | 50 |
| **Total Teaching hours** | 30 | **Exam Hours** | 03 |
| **CREDITS – 01** | | | |
| **Course objectives:** This course will enable students | | | |

Course Objectives:
1. Explain the ns3 simulator, installation and its application.
2. Illustrate the creation of point-to-point link, TCP, UDP protocols its connection.
3. Demonstrate the connection establishment of network computing devices.
4. Discuss tracking, testing, analyzing the network.

1. Using TCP/IP Socket programming implement a program to transfer the contents of a requested file from server to the client using TCP/IP sockets.
2. Implement the data link layer farming methods such as character, character stuffing and bit stuffing.
3. Implement on a data of set of characters the three CRC polynomials-CRC 12, CRC 16 and CRC CCIP.
4. Write a program for frame sorting techniques used in buffers.
5. Write a program for Hamming Code generation for error detection and correction.
6. Take an example subnet graph with weights indicating delay between nodes. Now obtain routing table at each node using distance vector routing algorithm.
7. Using bucket algorithm, design a program to achieve traffic management at flow level by implementing closed loop control technique.
8. Using RSA algorithm encrypt a text data and decrypt the same.
9. a) Write a NS3 program to connect two nodes with a point-to-point link, which have unique interface. Analyze the network performance using UDP client server.
   b) Write NS 3 Program to configure two nodes on an 802.11b physical layer, with802.11b NICs in Ad hoc mode, and by default, sends one packet of 1000 (application) bytes to the other node. The physical layer is configured to receive at a fixed RSS (regardless of the distance and transmit power); therefore, changing position of the nodes has no effect. Analyze the performance.
10. a Configure network topology using switch and router (LAN, Internet).
    b Configure network topology to implement VLAN using packet tracer.

## 1. Using TCP/IP Socket programming, implement a program to transfer the contents of a requested file from server to the client using TCP/IP Sockets.

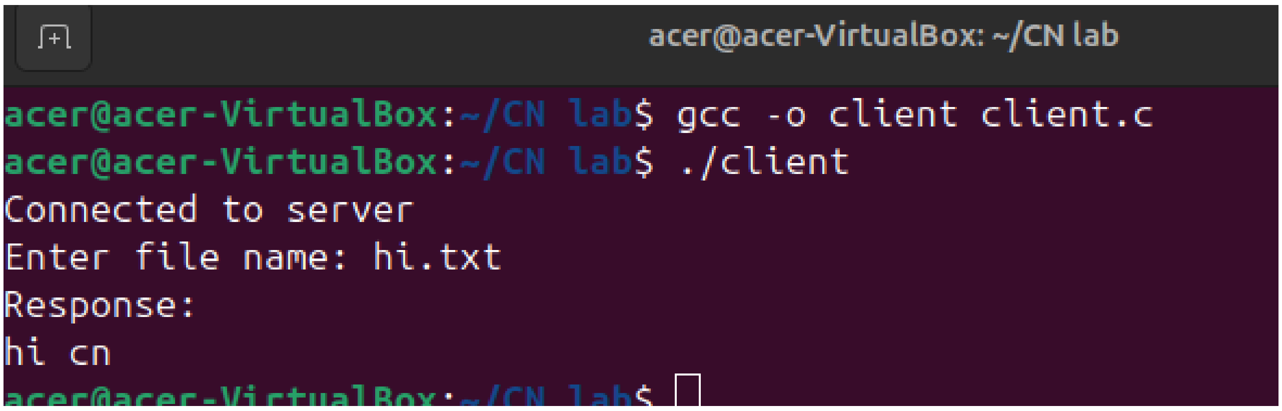
```
// client.c
#include <stdio.h>
#include <arpa/inet.h>
#include <unistd.h>

int main() {
    int soc = socket(PF_INET, SOCK_STREAM, 0);
    if (soc < 0) return 1; // Exit if socket creation fails

    struct sockaddr_in addr = {
        .sin_family = AF_INET,
        .sin_port = htons(7891),
        .sin_addr.s_addr = inet_addr("127.0.0.1")
    };

    if (connect(soc, (struct sockaddr *) &addr, sizeof(addr)) == 0) {
        printf("Connected to server\nEnter file name: ");
        char fname[50], buffer[1024];
        scanf("%s", fname);

        send(soc, fname, sizeof(fname), 0);
        printf("Response:\n");
        while (recv(soc, buffer, sizeof(buffer), 0) > 0)
            printf("%s", buffer);
    } else {
        perror("Connection failed");
    }

    close(soc);
    return 0;
}
```

//**server.c**

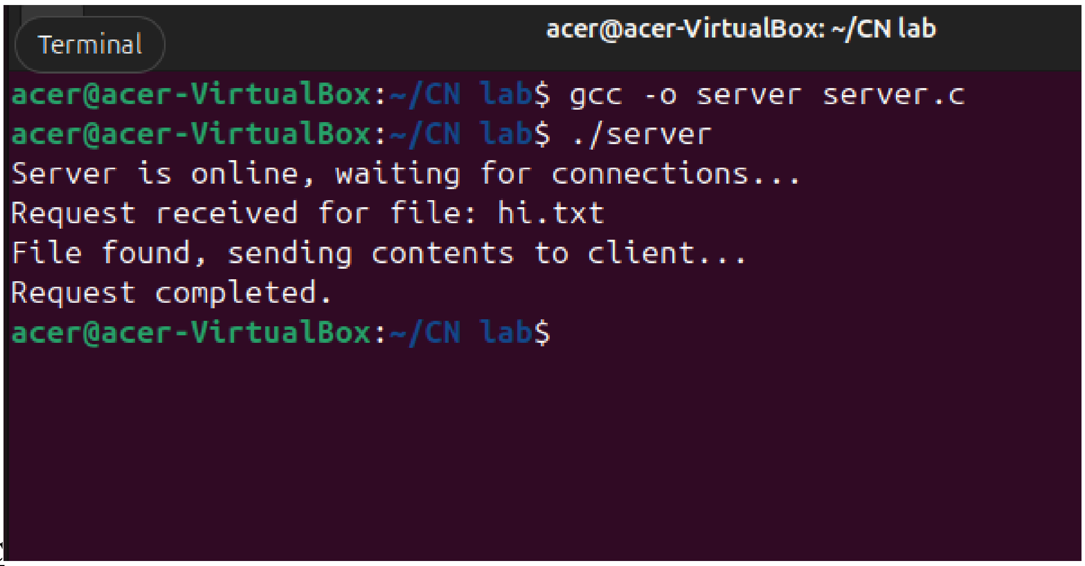
```
#include <stdio.h>
#include <arpa/inet.h>
#include <fcntl.h>
#include <unistd.h>

int main() {
    int welcome = socket(PF_INET, SOCK_STREAM, 0);
    struct sockaddr_in addr = { .sin_family = AF_INET, .sin_port = htons(7891), .sin_addr.s_addr =
inet_addr("127.0.0.1") };

    bind(welcome, (struct sockaddr*)&addr, sizeof(addr));
    printf("Server is online, waiting for connections...\n");

    listen(welcome, 5);
    int new_soc = accept(welcome, NULL, NULL);

    char fname[50], buffer[1024];
    recv(new_soc, fname, sizeof(fname), 0);
    printf("Request received for file: %s\n", fname);
    int fd = open(fname, O_RDONLY);
    if (fd < 0) {
        send(new_soc, "File not found\n", 15, 0);
        printf("File not found, notified client.\n");
    } else {
        printf("File found, sending contents to client...\n");
        int n;
        while ((n = read(fd, buffer, sizeof(buffer))) > 0)
            send(new_soc, buffer, n, 0);
    }

    close(fd);
    close(new_soc);
    close(welcome);
    printf("Request completed.\n");
    return 0;
}
```

**OUTPUT**

**Client. C**



```
acer@acer-VirtualBox:~/CN lab$ gcc -o client client.c
acer@acer-VirtualBox:~/CN lab$ ./client
Connected to server
Enter file name: hi.txt
Response:
hi cn
acer@acer-VirtualBox:~/CN lab$
```



```
acer@acer-VirtualBox:~/CN lab$ gcc -o server server.c
acer@acer-VirtualBox:~/CN lab$ ./server
Server is online, waiting for connections...
Request received for file: hi.txt
File found, sending contents to client...
Request completed.
acer@acer-VirtualBox:~/CN lab$
```

**SERVER.C**

**2. Implement the data link layer farming methods such as character, character stuffing and bit stuffing.**

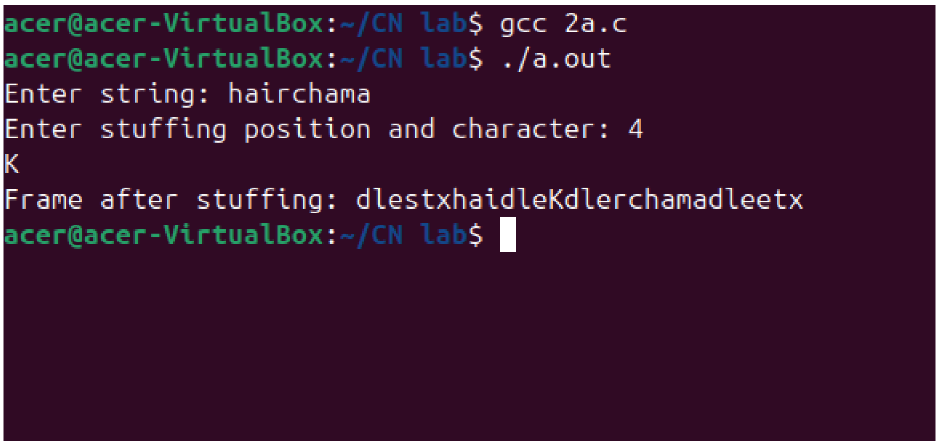
**(a) Character stuffing**

```
//program for character stuffing
#include <stdio.h>
#include <string.h>

void character_stuffing(const char *input, char *output, char ch, int pos) {
   const char *dle = "dle";
   strcpy(output, "dlestx");
   int j = 6;

   for (int i = 0; input[i] != '\0'; i++) {
      if (i == pos - 1) {
         strcat(output + j, dle);
         output[j + 3] = ch;
         strcat(output + j + 4, dle);
         j += 7;
      } else if (!strncmp(&input[i], dle, 3)) {
         strcat(output + j, dle);
         j += 3;
      }
      output[j++] = input[i];
   }
   strcat(output + j, "dleetx");
}

int main() {
   char input[20], output[50];
   int pos;
   char ch;

   printf("Enter string: ");
```

```
    scanf("%s", input);
    printf("Enter stuffing position and character: ");
    scanf("%d %c", &pos, &ch);

    character_stuffing(input, output, ch, pos);
    printf("Frame after stuffing: %s\n", output);
    return 0;
}
```

**OUTPUT:**

```
Enter String:
haiarchana
Enter position:
4
Enter the Character
K
Frame after stuffing:
Dlestxhaidlekdlearchanadleetx
```

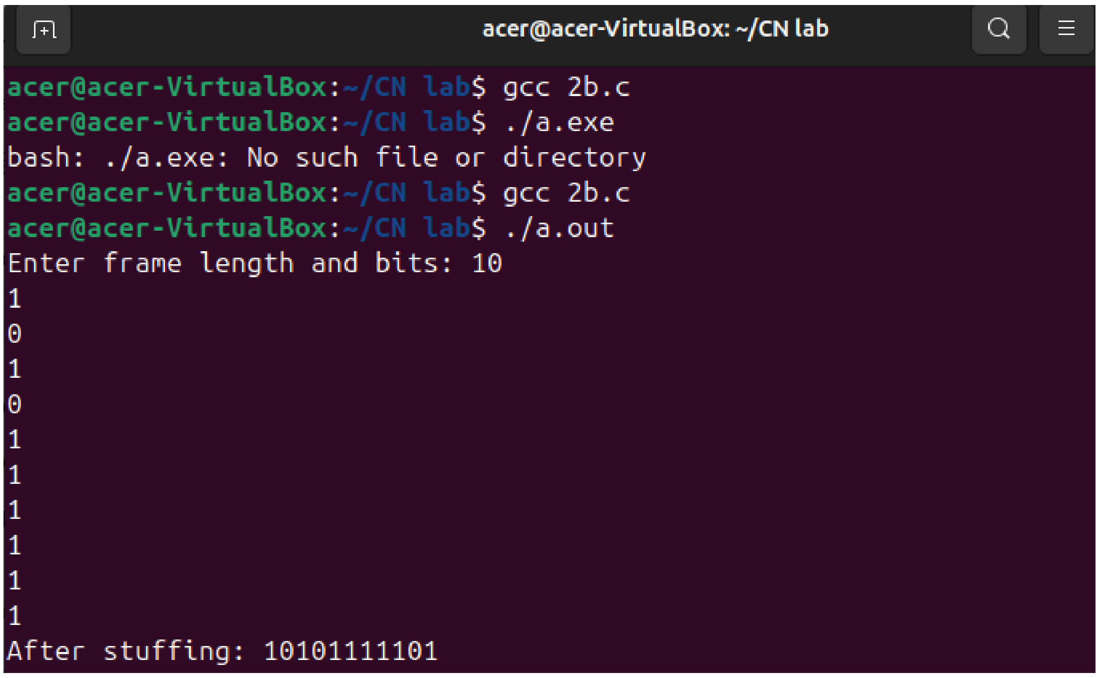


```
acer@acer-VirtualBox:~/CN lab$ gcc 2a.c
acer@acer-VirtualBox:~/CN lab$ ./a.out
Enter string: hairchama
Enter stuffing position and character: 4
K
Frame after stuffing: dlestxhaidleKdlerchamadleetx
acer@acer-VirtualBox:~/CN lab$ 
```

**(b) Bit stuffing**

```
#include <stdio.h>

void bit_stuffing(int *input, int n) {
    int output[50], j = 0, count = 0;
    for (int i = 0; i < n; i++) {
        output[j++] = input[i];
        if (input[i] == 1) {
            count++;
            if (count == 5) {
                output[j++] = 0;
                count = 0;
            }
        } else {
            count = 0;
        }
    }
    printf("After stuffing: ");
    for (int i = 0; i < j; i++) printf("%d", output[i]);
    printf("\n");
}

int main() {
    int n, input[20];
    printf("Enter frame length and bits: ");
    scanf("%d", &n);
    for (int i = 0; i < n; i++) scanf("%d", &input[i]);

    bit_stuffing(input, n);
    return 0;
}
```

**OUTPUT**



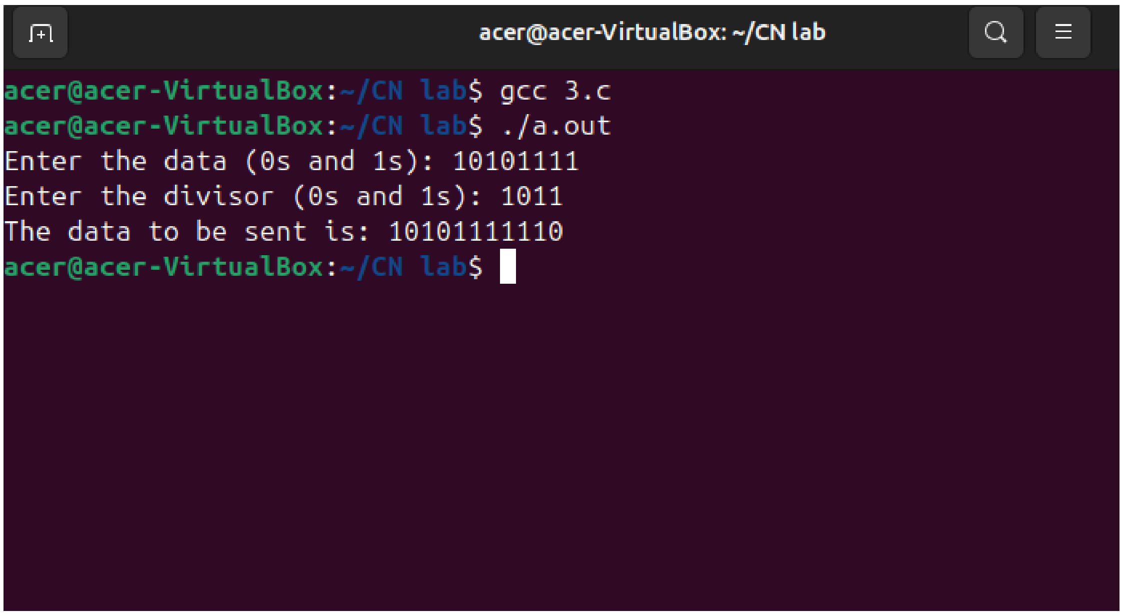
```
acer@acer-VirtualBox:~/CN lab$ gcc 2b.c
acer@acer-VirtualBox:~/CN lab$ ./a.exe
bash: ./a.exe: No such file or directory
acer@acer-VirtualBox:~/CN lab$ gcc 2b.c
acer@acer-VirtualBox:~/CN lab$ ./a.out
Enter frame length and bits: 10
1
0
1
0
1
1
1
1
1
1
After stuffing: 10101111101
```

**3. Write a C program to implement on a data set characters the three CRC polynomials – CRC 12, CRC 16, and CRC CCIP.**

```
// program for Cyclic Redundancy Check
#include <stdio.h>

int main() {
    int data[50], div[16], rem[16];
    int datalen = 0, divlen = 0;

    // Input data
    printf("Enter the data (0s and 1s): ");
    char ch;
    while ((ch = getchar()) != '\n')
        data[datalen++] = ch - '0';

    // Input divisor
    printf("Enter the divisor (0s and 1s): ");
    while ((ch = getchar()) != '\n')
        div[divlen++] = ch - '0';

    // Append zeros to data for CRC
    for (int i = 0; i < divlen - 1; i++)
        data[datalen + i] = 0;

    // Initialize remainder with the first part of the data
    for (int i = 0; i < divlen; i++)
        rem[i] = data[i];

    // Perform division
    for (int i = divlen; i <= datalen + divlen - 1; i++) {
        if (rem[0] == 1) {
            for (int j = 1; j < divlen; j++)
                rem[j - 1] = rem[j] ^ div[j];
        } else {
            for (int j = 1; j < divlen; j++)
                rem[j - 1] = rem[j];
        }
        rem[divlen - 1] = data[i];
    }

    // Combine remainder with data for the final message
    for (int i = 0; i < divlen - 1; i++)
        data[datalen + i] = rem[i];

    // Display final data
    printf("The data to be sent is: ");
    for (int i = 0; i < datalen + divlen - 1; i++)
        printf("%d", data[i]);
    printf("\n");

    return 0;
}
```

**OUTPUT:**

```
Enter the data(0s and 1s): 10101111
Enter the divisor(0s and 1s): 1011
The data to be sent is:10101111110
```



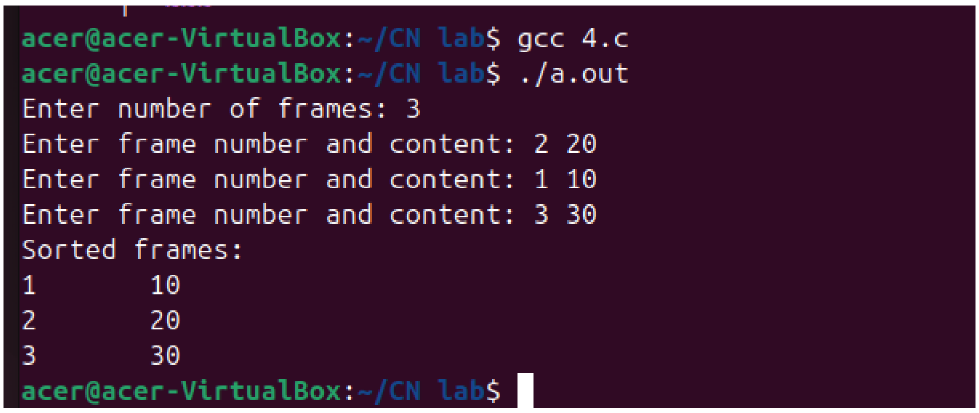
```
acer@acer-VirtualBox:~/CN lab$ gcc 3.c
acer@acer-VirtualBox:~/CN lab$ ./a.out
Enter the data (0s and 1s): 10101111
Enter the divisor (0s and 1s): 1011
The data to be sent is: 10101111110
acer@acer-VirtualBox:~/CN lab$
```

**4. Write a program for frame sorting technique used in buffers.**

```
#include <stdio.h>

typedef struct {

   int num;

   char str[50];

} Frame;

void sort(Frame arr[], int n) {

   for (int i = 0; i < n - 1; i++) {

       for (int j = i + 1; j < n; j++) {

           if (arr[i].num > arr[j].num) {

               Frame temp = arr[i];

               arr[i] = arr[j];

               arr[j] = temp;

           }

       }

   }

}

int main() {

   int n;
```

```
    printf("Enter number of frames: ");
    scanf("%d", &n);
    Frame arr[n];
    for (int i = 0; i < n; i++) {
        printf("Enter frame number and content: ");
        scanf("%d %s", &arr[i].num, arr[i].str);
    }
    sort(arr, n);
    printf("Sorted frames:\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t%s\n", arr[i].num, arr[i].str);
    }
    return 0;
}
```

**OUTPUT**

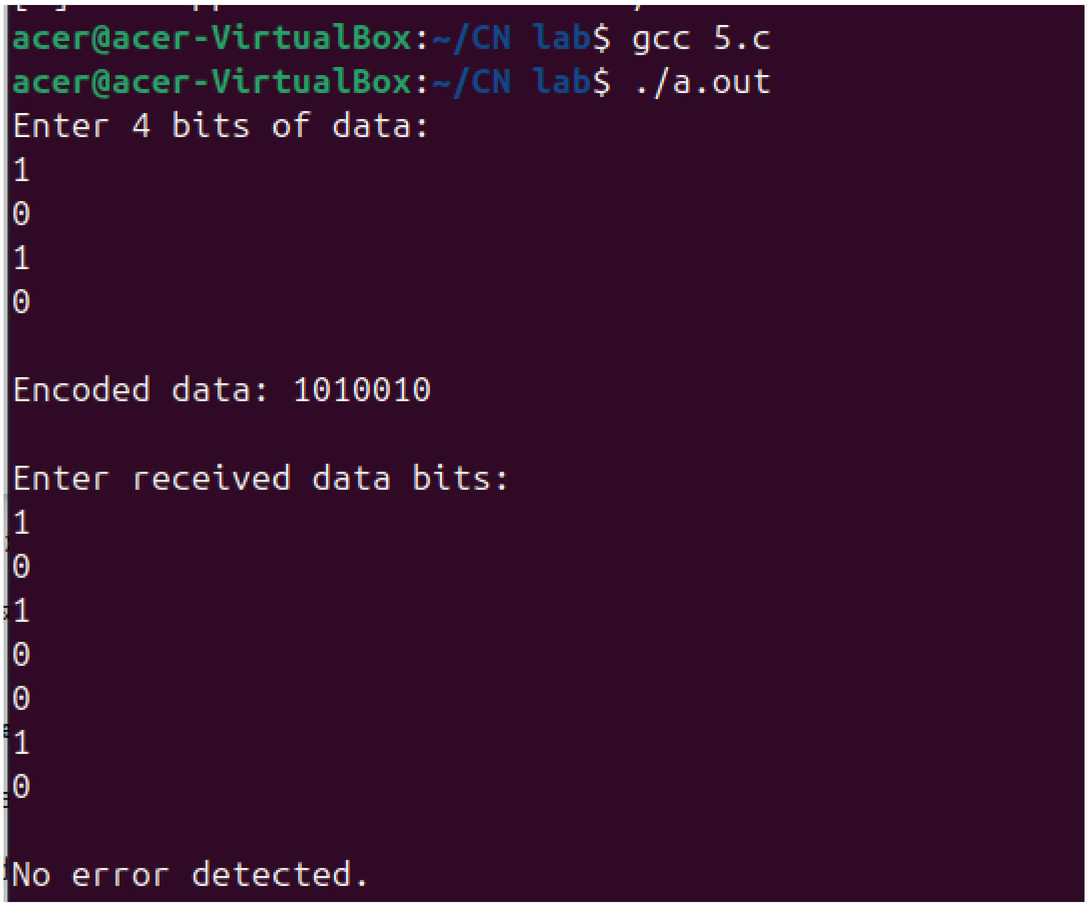


```
acer@acer-VirtualBox:~/CN lab$ gcc 4.c
acer@acer-VirtualBox:~/CN lab$ ./a.out
Enter number of frames: 3
Enter frame number and content: 2 20
Enter frame number and content: 1 10
Enter frame number and content: 3 30
Sorted frames:
1       10
2       20
3       30
acer@acer-VirtualBox:~/CN lab$
```

**5. Write a program for Hamming Code generation for error detection and correction.**

```
#include<stdio.h>

void main() {
    int data[7], dataAtRec[7], c1, c2, c3, c, i;

    printf("Enter 4 bits of data:\n");
    for (i = 0; i < 4; i++) {
        scanf("%d", &data[i]);
    }

    // Calculate parity bits
    data[3] = data[0] ^ data[1] ^ data[2];
    data[5] = data[0] ^ data[1] ^ data[4];
    data[6] = data[0] ^ data[2] ^ data[4];

    printf("\nEncoded data: ");
    for (i = 0; i < 7; i++) {
        printf("%d", data[i]);
    }
    printf("\n");

    printf("\nEnter received data bits:\n");
    for (i = 0; i < 7; i++) {
        scanf("%d", &dataAtRec[i]);
    }

    // Calculate parity check bits
    c1 = dataAtRec[6] ^ dataAtRec[4] ^ dataAtRec[2] ^ dataAtRec[0];
    c2 = dataAtRec[5] ^ dataAtRec[4] ^ dataAtRec[1] ^ dataAtRec[0];
    c3 = dataAtRec[3] ^ dataAtRec[2] ^ dataAtRec[1] ^ dataAtRec[0];
    c = c3 * 4 + c2 * 2 + c1;

    if (c == 0) {
        printf("\nNo error detected.\n");
    } else {
        printf("\nError at position %d\n", c + 1);
        // Correct the error
        dataAtRec[7 - c] = (dataAtRec[7 - c] == 0) ? 1 : 0;

        printf("\nCorrected data: ");
        for (i = 0; i < 7; i++) {
            printf("%d", dataAtRec[i]);
        }
        printf("\n");
    }
}
```

**OUTPUT**



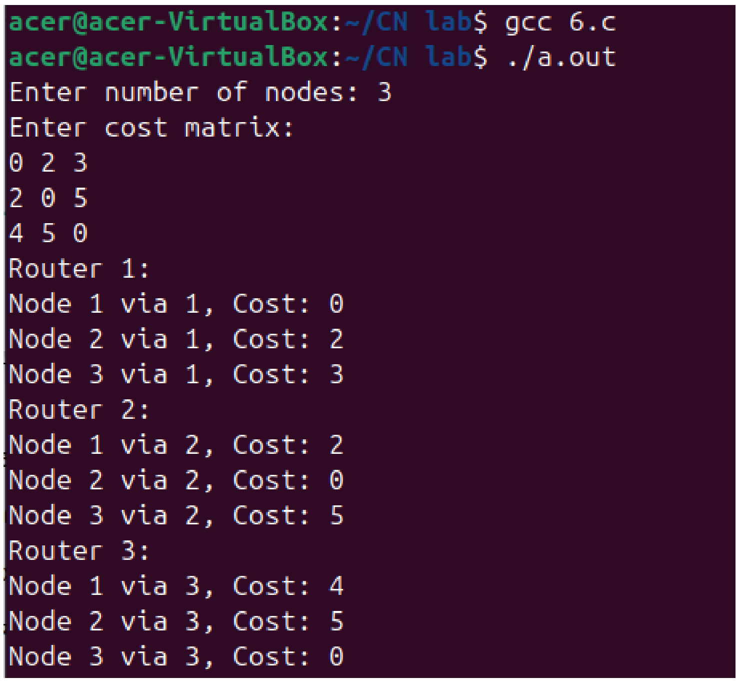
```
acer@acer-VirtualBox:~/CN lab$ gcc 5.c
acer@acer-VirtualBox:~/CN lab$ ./a.out
Enter 4 bits of data:
1
0
1
0

Encoded data: 1010010

Enter received data bits:
1
0
1
0
0
1
0

No error detected.
```

**6. Take an example subnet graph with weights indicating delay between nodes. Now obtain Routing table at each node using distance vector routing algorithm.**

```
#include <stdio.h>

int main() {
   int nodes, cost[10][10];
   printf("Enter number of nodes: ");
   scanf("%d", &nodes);
   printf("Enter cost matrix:\n");
   for (int i = 0; i < nodes; i++)
      for (int j = 0; j < nodes; j++)
scanf("%d", &cost[i][j]);

   for (int i = 0; i < nodes; i++) {
      printf("Router %d:\n", i + 1);
      for (int j = 0; j < nodes; j++)
printf("Node %d via %d, Cost: %d\n", j + 1, i + 1, cost[i][j]);
   }
   return 0;
```

}

**OUTPUT**

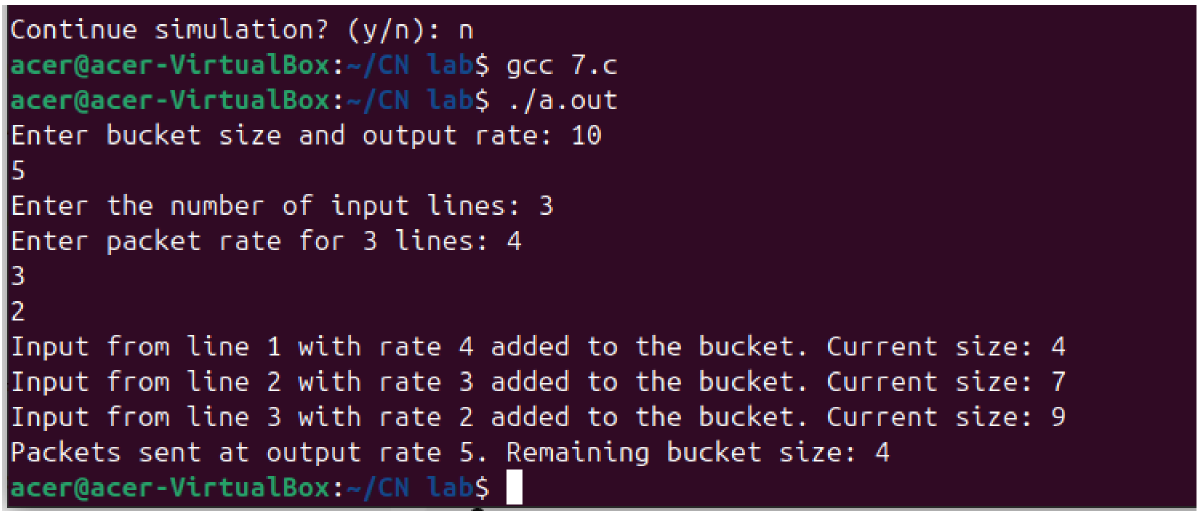


```
acer@acer-VirtualBox:~/CN lab$ gcc 6.c
acer@acer-VirtualBox:~/CN lab$ ./a.out
Enter number of nodes: 3
Enter cost matrix:
0 2 3
2 0 5
4 5 0
Router 1:
Node 1 via 1, Cost: 0
Node 2 via 1, Cost: 2
Node 3 via 1, Cost: 3
Router 2:
Node 1 via 2, Cost: 2
Node 2 via 2, Cost: 0
Node 3 via 2, Cost: 5
Router 3:
Node 1 via 3, Cost: 4
Node 2 via 3, Cost: 5
Node 3 via 3, Cost: 0
```

**7. Using Leaky Bucket Algorithm, Design a program to achieve Traffic management at Flow level by implementing Closed Loop Control technique.**

```
#include <stdio.h>

int main() {

    int bucketSize, outputRate, line, packetRate[25], currentSize = 0;

    // Input bucket size and output rate

    printf("Enter bucket size and output rate: ");

    scanf("%d %d", &bucketSize, &outputRate);


    // Input number of lines

    printf("Enter the number of input lines: ");

    scanf("%d", &line);

    // Input the packet rate for each line

    printf("Enter packet rate for %d lines: ", line);

    for (int i = 0; i < line; i++) {

        scanf("%d", &packetRate[i]);
    }
    // Process each packet rate
    for (int i = 0; i < line; i++) {
        currentSize += packetRate[i];  // Add packet to the bucket
        // Check if bucket overflows
        if (currentSize <= bucketSize) {
            printf("Input from line %d with rate %d added to the bucket. Current size: %d\n", i + 1, packetRate[i],
currentSize);
        } else {
            currentSize -= packetRate[i];  // Discard packet if overflow
            printf("Input from line %d with rate %d discarded. Current size: %d\n", i + 1, packetRate[i], currentSize);
        }
    }
    // Output the packets at the specified output rate
    if (currentSize <= outputRate) {
        printf("All packets sent at output rate %d. Bucket empty.\n", currentSize);
        currentSize = 0;
    } else {
        currentSize -= outputRate;
        printf("Packets sent at output rate %d. Remaining bucket size: %d\n", outputRate, currentSize);
    }
    return 0;
}
```

**OUTPUT**



```
Continue simulation? (y/n): n
acer@acer-VirtualBox:~/CN lab$ gcc 7.c
acer@acer-VirtualBox:~/CN lab$ ./a.out
Enter bucket size and output rate: 10
5
Enter the number of input lines: 3
Enter packet rate for 3 lines: 4
3
2
Input from line 1 with rate 4 added to the bucket. Current size: 4
Input from line 2 with rate 3 added to the bucket. Current size: 7
Input from line 3 with rate 2 added to the bucket. Current size: 9
Packets sent at output rate 5. Remaining bucket size: 4
acer@acer-VirtualBox:~/CN lab$ 
```

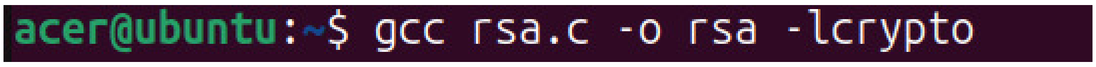
8) Using RSA algorightm Encrypt the text and Decrypt the Same.

```
#include <openssl/rsa.h>
#include <openssl/pem.h>
#include <openssl/err.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
void handle_errors(const char *msg) {
    fprintf(stderr, "%s\n", msg);
    ERR_print_errors_fp(stderr);
    exit(1);
}
RSA* generate_rsa_keypair(int bits) {
    RSA *rsa = RSA_new();
    BIGNUM *bn = BN_new();

    if (!BN_set_word(bn, RSA_F4)) handle_errors("Error setting BIGNUM word.");
    if (!RSA_generate_key_ex(rsa, bits, bn, NULL)) handle_errors("Error generating RSA key pair.");
    BN_free(bn);
    return rsa;
}
int main() {
    // Step 1: Generate RSA Key Pair
    int bits = 2048;
    RSA *rsa = generate_rsa_keypair(bits);
    // Step 2: Prepare the message to encrypt
    const char *message = "Hello, RSA encryption!";
    unsigned char encrypted[256]; // Buffer for encrypted message
    unsigned char decrypted[256]; // Buffer for decrypted message
    int encrypted_length, decrypted_length;
    printf("Original message: %s\n", message);
    // Step 3: Encrypt the message using the public key
    encrypted_length = RSA_public_encrypt(strlen(message), (unsigned char *)message, encrypted, rsa,
RSA_PKCS1_OAEP_PADDING);
    if (encrypted_length == -1) handle_errors("Error encrypting message.");
    printf("Encrypted message (hex): ");
    for (int i = 0; i < encrypted_length; i++) {
        printf("%02x", encrypted[i]);
    }
    printf("\n");
    // Step 4: Decrypt the message using the private key
    decrypted_length = RSA_private_decrypt(encrypted_length, encrypted, decrypted, rsa,
RSA_PKCS1_OAEP_PADDING);
    if (decrypted_length == -1) handle_errors("Error decrypting message.");
    decrypted[decrypted_length] = '\0'; // Null-terminate the decrypted message
    printf("Decrypted message: %s\n", decrypted);
    // Clean up
    RSA_free(rsa);
    return 0;

}
```

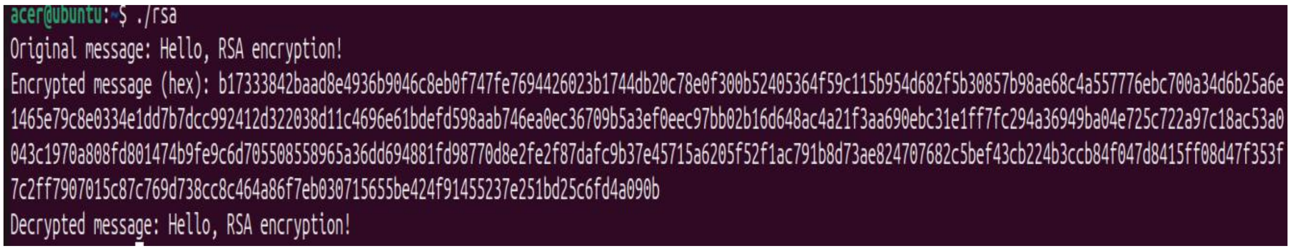
**OUTPUT**

**LIBRARIES:**

```
sudo apt update

sudo apt install libssl-dev
```



```
acer@ubuntu:~$ gcc rsa.c -o rsa -lcrypto
```



```
acer@ubuntu:~$ ./rsa
Original message: Hello, RSA encryption!
Encrypted message (hex): b17333842baad8e4936b9046c8eb0f747fe7694426023b1744db20c78e0f300b52405364f59c115b954d682f5b30857b98ae68c4a557776ebc700a34d6b25a6e
1465e79c8e0334e1dd7b7dcc992412d322038d11c4696e61bdefd598aab746ea0ec36709b5a3ef0eec97bb02b16d648ac4a21f3aa690ebc31e1ff7fc294a36949ba04e725c722a97c18ac53a0
043c1970a808fd801474b9fe9c6d705508558965a36dd694881fd98770d8e2fe2f87dafc9b37e45715a6205f52f1ac791b8d73ae824707682c5bef43cb224b3ccb84f047d8415ff08d47f353f
7c2ff7907015c87c769d738cc8c464a86f7eb030715655be424f91455237e251bd25c6fd4a090b
Decrypted message: Hello, RSA encryption!
```

9) a) Write a NS3 program to connect two nodes with a point-to-point link, which have a unique interface. Analyze the network performance using UDP client server.

```
#include "ns3/core-module.h"

#include "ns3/network-module.h"

#include "ns3/internet-module.h"

#include "ns3/point-to-point-module.h"

#include "ns3/applications-module.h"


using namespace ns3;


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

    CommandLine cmd;

    cmd.Parse(argc, argv);


    // Create two nodes
```

```
NodeContainer nodes;
nodes.Create(2);

// Set up the point-to-point link
PointToPointHelper pointToPoint;
pointToPoint.SetDeviceAttribute("DataRate", StringValue("5Mbps"));
pointToPoint.SetChannelAttribute("Delay", StringValue("2ms"));

// Install network devices
NetDeviceContainer devices;
devices = pointToPoint.Install(nodes);

// Install Internet stack
InternetStackHelper internet;
internet.Install(nodes);

// Assign IP addresses
Ipv4AddressHelper address;
address.SetBase("10.1.1.0", "255.255.255.0");
Ipv4InterfaceContainer interfaces = address.Assign(devices);

// Set up the UDP server on node 1
uint16_t port = 9; // Well-known port for echo
UdpServerHelper server(port);
ApplicationContainer serverApp = server.Install(nodes.Get(1));
serverApp.Start(Seconds(1.0));
serverApp.Stop(Seconds(10.0));
```

```
    // Set up the UDP client on node 0
    UdpClientHelper client(interfaces.GetAddress(1), port);
    client.SetAttribute("MaxPackets", UintegerValue(320));
    client.SetAttribute("Interval", TimeValue(Seconds(0.05)));
    client.SetAttribute("PacketSize", UintegerValue(1024));

    ApplicationContainer clientApp = client.Install(nodes.Get(0));
    clientApp.Start(Seconds(2.0));
    clientApp.Stop(Seconds(10.0));

    // Enable tracing
    pointToPoint.EnablePcapAll("point-to-point-udp");

    // Run the simulation
    Simulator::Run();
    Simulator::Destroy();

    return 0;
}
```

9) b) Write NS3 program to configure two nodes on an 802.11b physical layer, 802.11b NICs in AD hoc mode, and by default, sends one packet of 1000(applications) bytes to the other node. The physical layer is configured to

receive at a fixed RSS (regardless of distance and the transmit power); therefore changing the position of the nodes has no effect. Analyze the performance.

```
#include "ns3/core-module.h"

#include "ns3/network-module.h"

#include "ns3/internet-module.h"

#include "ns3/yans-wifi-helper.h"

#include "ns3/mobility-helper.h"

#include "ns3/wifi-module.h"

#include "ns3/applications-module.h"


using namespace ns3;


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

  CommandLine cmd;

  cmd.Parse(argc, argv);


  // Create two nodes

  NodeContainer nodes;

  nodes.Create(2);


  // Configure the WiFi physical layer with 802.11b standard

  YansWifiChannelHelper channel = YansWifiChannelHelper::Default();

  YansWifiPhyHelper phy = YansWifiPhyHelper::Default();

  phy.SetChannel(channel.Create());

  phy.Set("RxGain", DoubleValue(0));

  phy.Set("CcaMode1Threshold", DoubleValue(-64));

  phy.Set("TxPowerStart", DoubleValue(16));

  phy.Set("TxPowerEnd", DoubleValue(16));
  phy.Set("RxNoiseFigure", DoubleValue(7));

  // Fixed RSS irrespective of distance
  phy.Set("FixedRssDbm", DoubleValue(-50));

  // Configure the WiFi MAC layer
  WifiHelper wifi;
  wifi.SetStandard(WIFI_STANDARD_80211b);
  WifiMacHelper mac;
  mac.SetType("ns3::AdhocWifiMac");

  // Install WiFi devices
  NetDeviceContainer devices = wifi.Install(phy, mac, nodes);

  // Install Internet stack
  InternetStackHelper internet;
  internet.Install(nodes);

  // Assign IP addresses
  Ipv4AddressHelper address;
  address.SetBase("10.1.2.0", "255.255.255.0");
  Ipv4InterfaceContainer interfaces = address.Assign(devices);

  // Mobility model - fixed positions
  MobilityHelper mobility;
  mobility.SetPositionAllocator("ns3::GridPositionAllocator",
                          "MinX", DoubleValue(0.0),
                          "MinY", DoubleValue(0.0),
                          "DeltaX", DoubleValue(5.0),
                          "DeltaY", DoubleValue(0.0),
                          "GridWidth", UintegerValue(2),
                          "LayoutType", StringValue("RowFirst"));
    mobility.SetMobilityModel("ns3::ConstantPositionMobilityModel");
    mobility.Install(nodes);

    // UDP Application setup
    UdpServerHelper server(9);
    ApplicationContainer serverApp = server.Install(nodes.Get(1));
    serverApp.Start(Seconds(1.0));
    serverApp.Stop(Seconds(10.0));

    UdpClientHelper client(interfaces.GetAddress(1), 9);
    client.SetAttribute("MaxPackets", UintegerValue(1));
    client.SetAttribute("Interval", TimeValue(Seconds(1.0)));
    client.SetAttribute("PacketSize", UintegerValue(1000));

    ApplicationContainer clientApp=client.Install(nodes.Get(0));
    clientApp.Start(Seconds(2.0));
    clientApp.Stop(Seconds(10.0));
   // Enable tracing
   phy.EnablePcap("wifi-adhoc", devices);

    // Run simulation
    Simulator::Run();
```

```
Simulator::Destroy();


return 0
```