

**PRACTICAL FILE**

**(COMPUTER NETWORKS)**

**Submitted to:**

**MR. DEEPAK MITTAL**

**Submitted by:**

**DIKSHA YADAV**

**Exam Roll no. 20015587008**

**Class Roll no. 20MTS5707**

**BSc. Mathematical Sciences**

**HTML PRACTICAL**

1. **Write a HTML program to design a form which should allow to enter your personal data. (Hint: make use of text field, password field, e-mail, lists, radio buttons, checkboxes, submit button).**

<html>

<head>

<title>

Personal DataForm

</title>

</head>

<body>

<h1>Personal Information Form</h1>

<form>

<h3> Name: <input> </h3>

<h3> Contact Number: <input type="Number"> </h3>

<h3> Email: <input> </h3>

<label for="password">Password:</label>

<input type="password" id="password" name="password">

<h3> Gender:

<input type="radio" id="Male" name="Gender" value="Male">

<label for="Male">Male</label><br>

<input type="radio" id="Female" name="Gender" value="Female">

<label for="Female">Female</label><br>

<input type="radio" id="Others" name="Gender" value="Others">

<label for="Others">Others</label><br>

</select><br>

<label for="state">Select a state:</label>

<select id="state" name="state">

<option value="Andhra Pradesh">Andhra Pradesh</option>

<option value="Arunachal Pradesh">Arunachal Pradesh</option>

<option value="Assam">Assam</option>

<option value="Bihar">Bihar</option>

<option value="Chhattisgarh">Chhattisgarh</option>

<option value="Goa">Goa</option>

<option value="Gujarat">Gujarat</option>

<option value="Haryana">Haryana</option>

<option value="Himachal Pradesh">Himachal Pradesh</option>

<option value="Jharkhand">Jharkhand</option>

<option value="Karnataka">Karnataka</option>

<option value="Kerala">Kerala</option>

<option value="Madhya Pradesh">Madhya Pradesh</option>

<option value="Maharashtra">Maharashtra</option>

<option value="Manipur">Manipur</option>

<option value="Meghalaya">Meghalaya</option>

<option value="Mizoram">Mizoram</option>

<option value="Nagaland">Nagaland</option>

<option value="Odisha">Odisha</option>

<option value="Punjab">Punjab</option>

<option value="Rajasthan">Rajasthan</option>

<option value="Sikkim">Sikkim</option>

<option value="Tamil Nadu">Tamil Nadu</option>

<option value="Telangana">Telangana</option>

<option value="Tripura">Tripura</option>

<option value="Uttar Pradesh">Uttar Pradesh</option>

<option value="Uttarakhand">Uttarakhand</option>

<option value="West Bengal">West Bengal</option>

</select><br><br>

<label for="interests">Select your interests:</label>

<input type="checkbox" id="sports" name="interests" value="sports">

<label for="sports">Sports</label>

<input type="checkbox" id="music" name="interests" value="music">

<label for="music">Music</label>

<input type="checkbox" id="movies" name="interests" value="movies">

<label for="movies">Movies</label>

<input type="checkbox" id="books" name="interests" value="books">

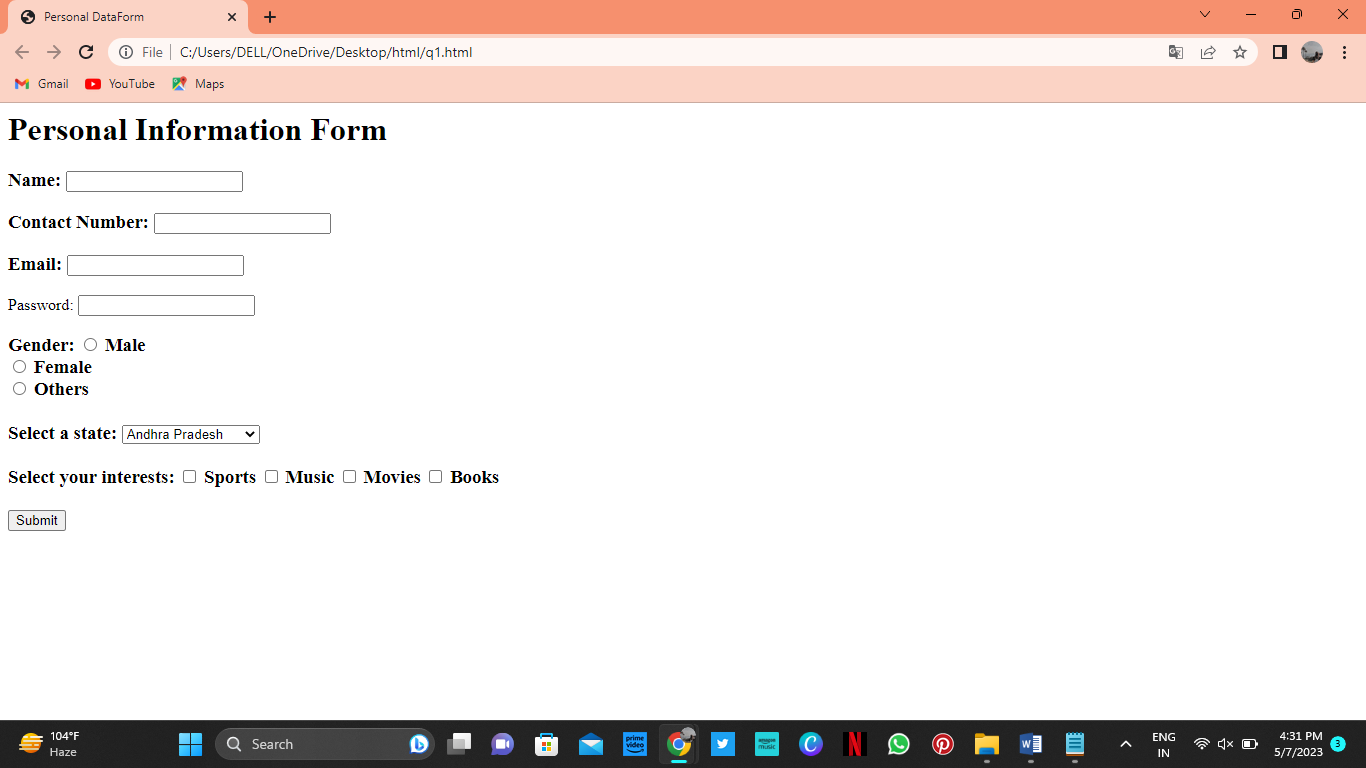
<label for="books">Books</label><br><br>

<input type="submit" value="Submit">

</form>

</body>

</html>



1. **Write html code to generate following output.**

**• Coffee**

**• Tea**

**o Black Tea**

**o Green Tea**

**• Milk**

<html>

<head>

<title>QUESTION 2 </title>

</head>

<body>

<ul>

<li>Coffee</li>

<li>Tea

<ul>

<li>Black Tea</li>

<li>Green Tea</li>

</ul>

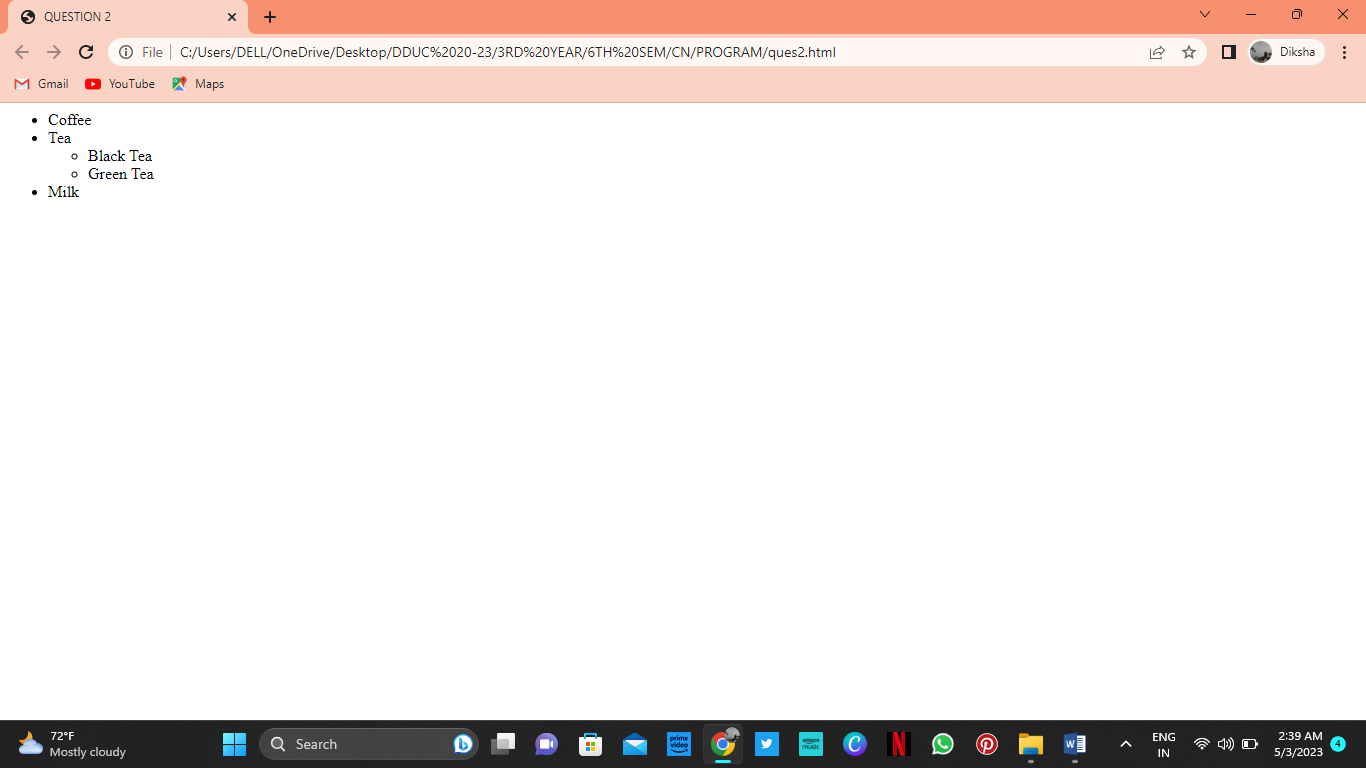
</li>

<li>Milk</li>

</ul>

</body>

</html>



1. **Design an html form to take the information of a customer visiting a departmental store such as name, contact phone no, preferred days of purchasing, favorite item (to be selected from a list of items), suggestions etc. One should provide button to Submit as well as Reset the form contents.**

<html>

<head>

<title>

Customer Feedback

</title>

</head>

<body>

<h1> Customer Feedback </h1>

<form>

<h3> Name: <input> </h3>

<h3> Contact Number: <input type="Number"> </h3>

<h3> Preferred days of purchasing:</h3>

<h4> Monday <input type="checkbox"><br>

Tuesday <input type="checkbox"><br>

Wednesday <input type="checkbox"><br>

Thursday <input type="checkbox"><br>

Friday <input type="checkbox"><br>

Saturday <input type="checkbox"><br>

Sunday <input type="checkbox"> </h4>

<h3> Favourite Items:</h3>

<h4> Groceries <input type="checkbox"> <br>

Stationary <input type="checkbox"> <br>

Garments <input type="checkbox"> <br>

Footwear <input type="checkbox"> </h4>

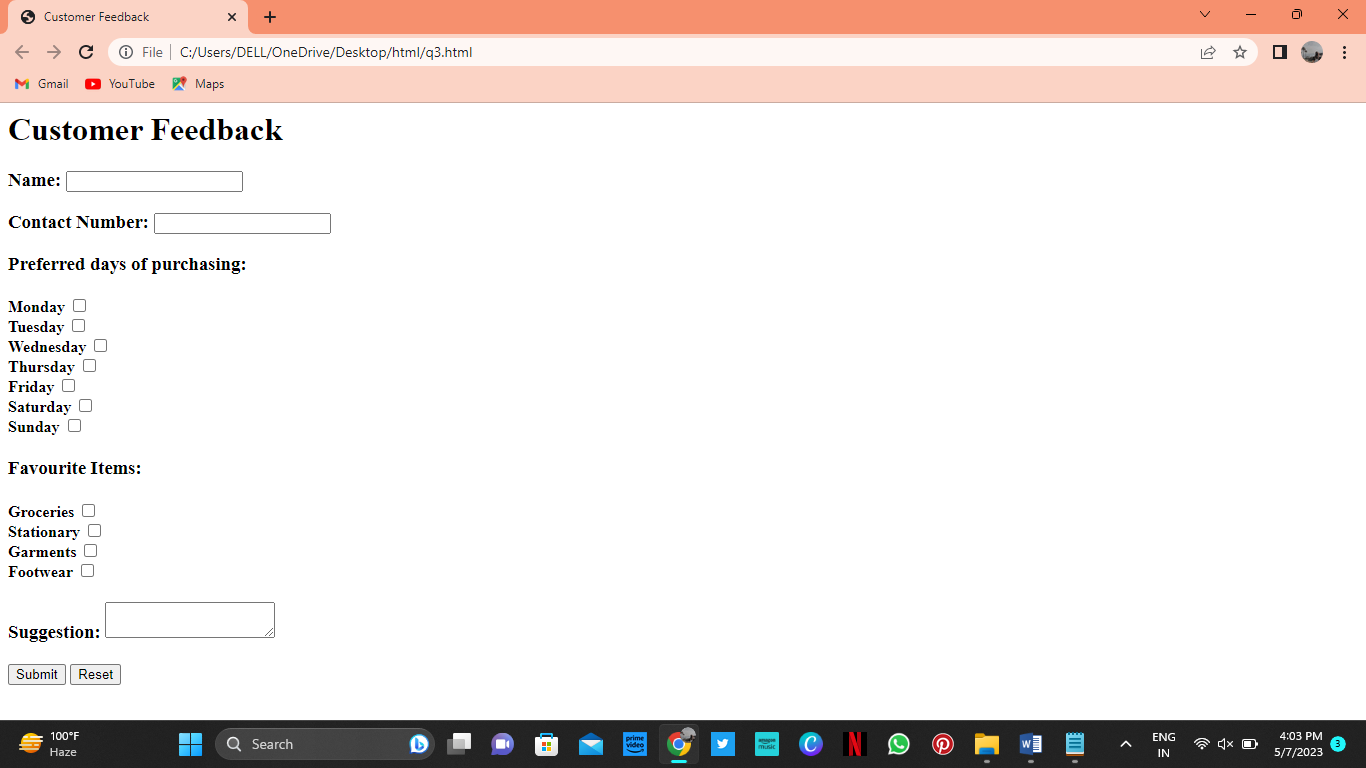
<h3> Suggestion: <textarea> </textarea> </h3>

<h3> <input type="SUBMIT"> <input type="RESET"> </h3>

</form>

</body>

</html>



1. **Design an html form to take the information of an article to be uploaded such as file path, author name, type (technical, literary, general), subject topic (to be selected from a list) etc. One should provide button to Submit as well as Reset the form contents.**

<html>

<head>

<title>

Article Information

</title>

</head>

<body>

<h1> Article Information </h1>

<form>

<h3> File: <input type="text" id="file" name="file"></h3>

<h3> Author Name:<input type="text" id="author" name="author"> </h3>

<h3> Type:</h3>

<input type="radio" id="technical" name="type" value="technical">

<label for="technical">Technical</label>

<input type="radio" id="literary" name="type" value="literary">

<label for="literary">Literary</label>

<input type="radio" id="general" name="type" value="general">

<label for="general">General</label><br>

<h3>Subject:

<select id="subject" name="subject">

<option value="technology">Technology</option>

<option value="science">Science</option>

<option value="health">Health</option>

<option value="politics">Politics</option>

<option value="economy">Economy</option>

</select>

</h3><br>

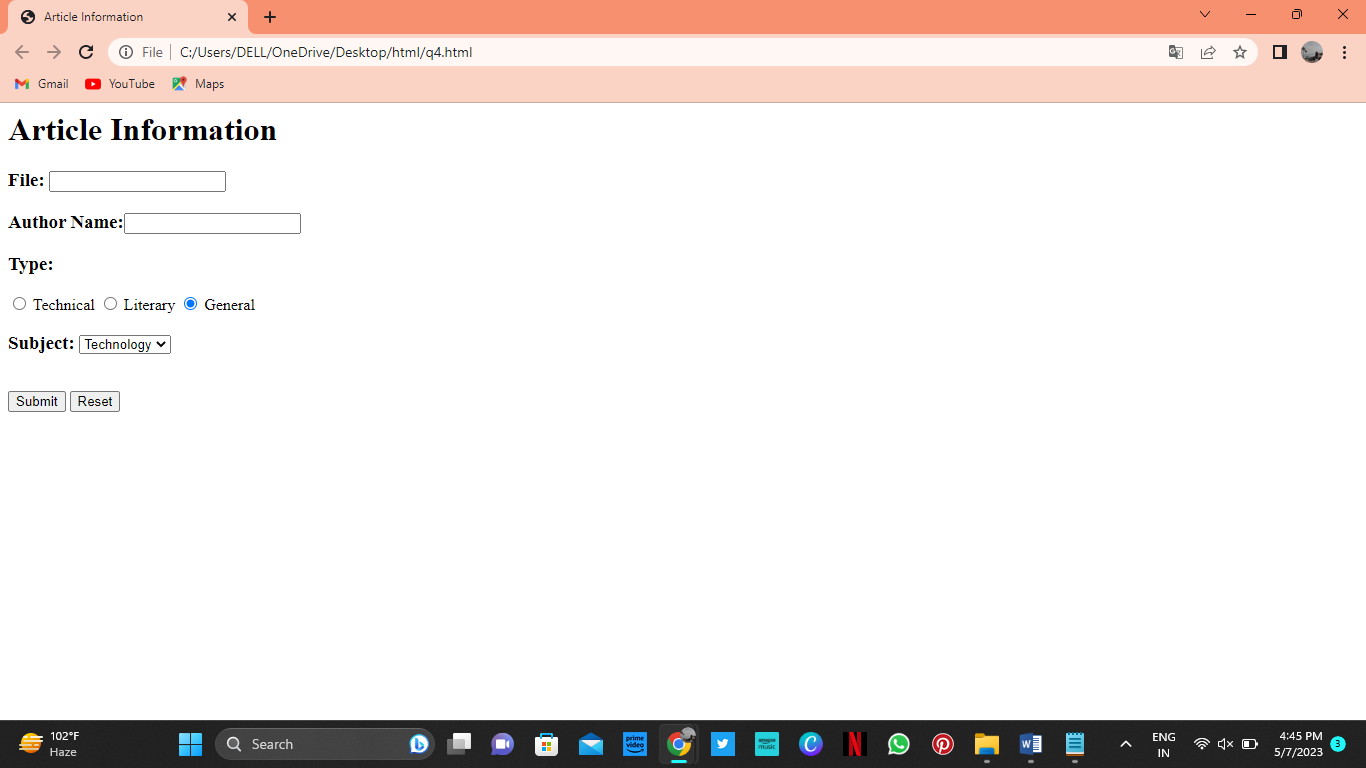
<input type="submit" value="Submit">

<input type="reset" value="Reset">

</form>

</body>

</html>



1. **Write a HTML code to generate following output. **

<html>

<head>

<title>

Question 5

</title>

</head>

<body>

<form>

<h3>Enter Name of your friend <input></h3>

<h3>Choose the file you want to post to your friend</h3> <h3><input type="text"><input type="file"></h3>

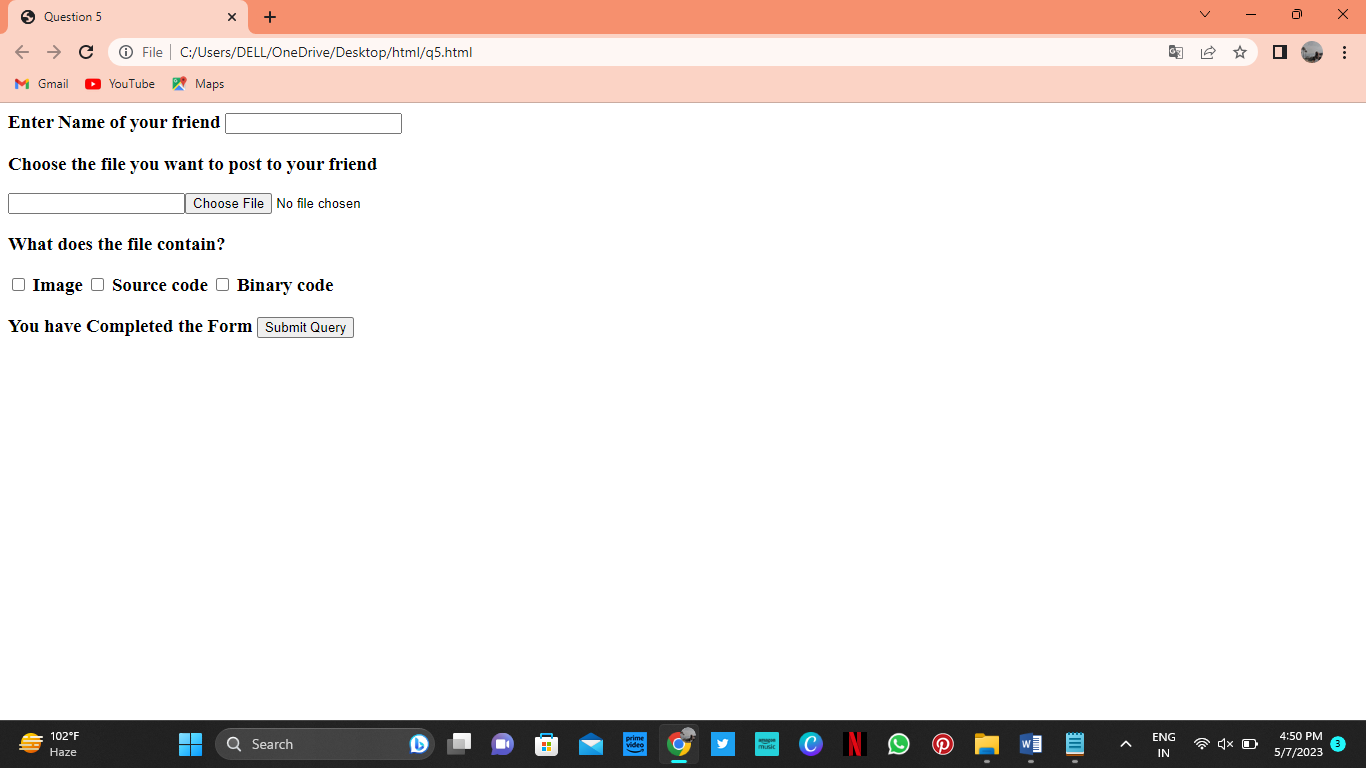
<h3>What does the file contain?</h3>

<h3><input type="checkbox"> Image <input type="checkbox"> Source code <input type="checkbox"> Binary code </h3> <h3>You have Completed the Form <input type= "button" value= "Submit Query"> </h3>

</form>

</body>

</html>



**NETWORK ALGORITHMS PRACTICAL LIST**

1. **Simulate Cyclic Redundancy Check(CRC) error detection algorithm for noisy channel.**

#include <iostream>

#include <cstring>

using namespace std;

const int CRC\_LEN = 16; // Length of the CRC polynomial

const int DATA\_LEN = 32; // Length of the data to be transmitted

// Function to perform CRC error detection

void crcErrorDetection(char data[], char crc[])

{

// Calculate the CRC remainder

char rem[CRC\_LEN];

memset(rem, 0, sizeof(rem));

for (int i = 0; i < DATA\_LEN; i++) {

rem[0] ^= data[i];

for (int j = 0; j < CRC\_LEN - 1; j++) {

if (rem[j] & 0x80) {

rem[j] = (rem[j] << 1) ^ 0x07;

rem[j+1] ^= 0x80;

} else {

rem[j] = (rem[j] << 1);

}

}

}

// Copy the remainder to the CRC

for (int i = 0; i < CRC\_LEN; i++) {

crc[i] = rem[i];

}

}

int main()

{

char data[DATA\_LEN] = "10111001010110000101100010011101"; // Data to be transmitted

char crc[CRC\_LEN] = "110000000000001"; // CRC polynomial

char noisyData[DATA\_LEN]; // Data transmitted over noisy channel

char receivedCRC[CRC\_LEN]; // Received CRC

// Simulate transmission over noisy channel

for (int i = 0; i < DATA\_LEN; i++) {

if (rand() % 100 < 10) { // Simulate 10% error rate

noisyData[i] = (data[i] == '0') ? '1' : '0';

} else {

noisyData[i] = data[i];

}

}

// Perform CRC error detection on received data

crcErrorDetection(noisyData, receivedCRC);

// Check if the received CRC matches the original CRC

if (strcmp(receivedCRC, crc) == 0) {

cout << "Transmission successful!" << endl;

} else {

cout << "Error detected in transmission!" << endl;

}

return 0;

}

1. **Simulate and implement stop and wait protocol for noisy channel.**

#include <iostream>

#include <time.h>

#include <unistd.h>

using namespace std;

int main() {

int frames[] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };

unsigned long seconds = 5000, to;

bool delay = false;

srand(time(NULL));

cout << "Sender has to send frames : ";

for (int i = 0; i < 10; i++)

cout << frames[i] << " ";

cout << endl << "timeout : 5s";

cout << endl << endl << "Sender\t\t\t\t\tReceiver" << endl;

int count = 0;

do {

bool timeout = false;

cout << "Sending Frame : " << frames[count]<< " ";

cout.flush();

cout << "\t\t";

to = rand() % 5500;

usleep(to \* 1000);

if (to <= seconds)

{

cout << "Received Frame : " << frames[count] << " ";

if (delay)

{

cout << "Duplicate";

delay = false;

}

cout << endl;

count++;

}

else

{

cout << "---" << endl;

cout << "Timeout" << endl;

timeout = true;

}

to = rand() % 5500;

usleep(to \* 1000);

if (to > seconds)

{

cout << "Delayed Ack" << endl;

count--;

delay = true;

}

else if (!timeout)

cout << "Acknowledgement : " << frames[count-1] << endl;

} while (count != 10);

return 0;

}

1. **Simulate and implement go back n sliding window protocol.**

#include <iostream>

#include <vector>

#include <random>

#include <chrono>

#include <thread>

using namespace std;

const int WINDOW\_SIZE = 4;

const int MAX\_SEQ\_NUM = 7;

const int TIMEOUT = 2000; // 2 seconds

struct Packet {

int seq\_num;

bool ack\_received;

};

// Function to simulate sending a packet over a noisy channel

bool send\_packet(Packet packet) {

// Simulate packet loss with a probability of 25%

random\_device rd;

mt19937 gen(rd());

uniform\_real\_distribution<> dis(0, 1);

double p = dis(gen);

if (p < 0.25) {

cout << "Packet " << packet.seq\_num << " lost" << endl;

return false;

}

// Simulate packet corruption with a probability of 10%

p = dis(gen);

if (p < 0.1) {

cout << "Packet " << packet.seq\_num << " corrupted" << endl;

return false;

}

cout << "Packet " << packet.seq\_num << " sent" << endl;

return true;

}

int main() {

vector<Packet> send\_buffer(WINDOW\_SIZE); // sliding send window

int send\_base = 0; // sequence number of oldest unacknowledged packet

int next\_seq\_num = 0; // next sequence number to use

int expected\_ack = 0; // expected acknowledgement number

chrono::steady\_clock::time\_point last\_sent\_time; // time when last packet was sent

while (send\_base < MAX\_SEQ\_NUM) {

// Send new packets if there is room in the window and not waiting for ACKs

while (next\_seq\_num < send\_base + WINDOW\_SIZE && next\_seq\_num <= MAX\_SEQ\_NUM) {

Packet packet = { next\_seq\_num, false };

send\_buffer[next\_seq\_num % WINDOW\_SIZE] = packet;

send\_packet(packet);

if (send\_base == next\_seq\_num) {

// Start timer for oldest unacknowledged packet

last\_sent\_time = chrono::steady\_clock::now();

}

next\_seq\_num++;

}

// Check for ACKs

Packet ack\_packet;

while (ack\_packet.ack\_received = true) {

// Check if timeout has expired

auto current\_time = chrono::steady\_clock::now();

auto time\_diff = chrono::duration\_cast<chrono::milliseconds>(current\_time - last\_sent\_time);

if (time\_diff.count() > TIMEOUT) {

cout << "Timeout expired for packet " << send\_base << endl;

// Resend all unacknowledged packets

next\_seq\_num = send\_base;

break;

}

// Receive ACKs

cout << "Waiting for ACK..." << endl;

// Simulate receiving an ACK packet

random\_device rd;

mt19937 gen(rd());

uniform\_int\_distribution<> dis(expected\_ack, send\_base + WINDOW\_SIZE - 1);

int ack\_num = dis(gen);

ack\_packet = send\_buffer[ack\_num % WINDOW\_SIZE];

if (ack\_packet.seq\_num == ack\_num) {

ack\_packet.ack\_received = true;

cout << "Received ACK for packet " << ack\_packet.seq\_num << endl;

expected\_ack = ack\_packet.seq\_num + 1;

send\_base = expected\_ack;

}

}

}

cout << "All packets sent successfully" << endl;

return 0;

}

1. **Simulate and implement selective repeat sliding window protocol.**

#include <iostream>

#include <vector>

#include <chrono>

#include <thread>

using namespace std;

using namespace chrono\_literals;

int main()

{

int windowSize, numFrames;

cout << "Enter the window size: ";

cin >> windowSize;

cout << "Enter the number of frames to transmit: ";

cin >> numFrames;

vector<int> frames(numFrames);

cout << "Enter the " << numFrames << " frames: ";

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

cin >> frames[i];

}

cout << "Sliding window protocol with selective repeat:\n\n";

vector<bool> acked(numFrames, false);

int totalTransmissions = 0;

for (int i = 0; i < numFrames;) {

bool hasTimeout = false;

for (int j = i; j < min(i + windowSize, numFrames); j++) {

if (!acked[j]) {

cout << "Sending Frame " << frames[j] << endl;

totalTransmissions++;

this\_thread::sleep\_for(1s); // simulate transmission time

if (rand() % 2 == 0) {

cout << "Frame " << frames[j] << " lost!\n";

hasTimeout = true;

} else {

cout << "Frame " << frames[j] << " received.\n";

acked[j] = true;

}

}

}

if (hasTimeout) {

cout << endl << "Timeout! Resending frames...\n\n";

continue;

}

i = min(i + windowSize, numFrames);

if (i == numFrames) {

break;

}

cout << endl;

for (int j = i - windowSize; j < i; j++) {

if (acked[j]) {

cout << "Frame " << frames[j] << " already acked.\n";

} else {

cout << "Frame " << frames[j] << " needs to be resent.\n";

totalTransmissions++;

}

}

cout << endl;

}

cout << "Transmission complete!\n\n";

cout << "Total transmissions attempted: " << totalTransmissions << endl;

return 0;

}

1. **Shortest Path algorithm.**

#include <iostream>

#include <limits.h>

using namespace std;

#define V 9

int minDistance(int dist[], bool sptSet[]) {

int min = INT\_MAX, min\_index;

for (int v = 0; v < V; v++) {

if (sptSet[v] == false && dist[v] <= min) {

min = dist[v];

min\_index = v;

}

}

return min\_index;

}

void printPath(int parent[], int j) {

if (parent[j] == -1) {

cout << j;

return;

}

printPath(parent, parent[j]);

cout << " -> " << j;

}

void printSolution(int dist[], int parent[], int src) {

cout << "Vertex \t Distance \t Path" << endl;

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

cout << i << " \t\t " << dist[i] << " \t\t ";

printPath(parent, i);

cout << endl;

}

}

void dijkstra(int graph[V][V], int src) {

int dist[V];

int parent[V];

bool sptSet[V];

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

dist[i] = INT\_MAX;

sptSet[i] = false;

parent[i] = -1;

}

dist[src] = 0;

for (int count = 0; count < V - 1; count++) {

int u = minDistance(dist, sptSet);

sptSet[u] = true;

for (int v = 0; v < V; v++) {

if (!sptSet[v] && graph[u][v] && dist[u] != INT\_MAX &&

dist[u] + graph[u][v] < dist[v]) {

dist[v] = dist[u] + graph[u][v];

parent[v] = u;

}

}

}

printSolution(dist, parent, src);

}

int main() {

int graph[V][V] = {

{ 0, 4, 0, 0, 0, 0, 0, 8, 0 },

{ 4, 0, 8, 0, 0, 0, 0, 11, 0 },

{ 0, 8, 0, 7, 0, 4, 0, 0, 2 },

{ 0, 0, 7, 0, 9, 14, 0, 0, 0 },

{ 0, 0, 0, 9, 0, 10, 0, 0, 0 },

{ 0, 0, 4, 14, 10, 0, 2, 0, 0 },

{ 0, 0, 0, 0, 0, 2, 0, 1, 6 },

{ 8, 11, 0, 0, 0, 0, 1, 0, 7 },

{ 0, 0, 2, 0, 0, 0, 6, 7, 0 }

};

int source = 0;

dijkstra(graph, source);

return 0;

}