#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#define FRAME\_SIZE 50 // Defines the maximum frame size

#define GEN\_SIZE 20 // Defines the maximum size of the generator polynomial

// Function declarations

void computeCRC();

void copyFrame();

// Global variables for computation

int i, j, k, gen\_size = 0;

int gen[GEN\_SIZE], frame[FRAME\_SIZE], rem[FRAME\_SIZE], temp[FRAME\_SIZE];

// Function to perform CRC division

void computeCRC() {

for (i = 0; i <= j - gen\_size; i++) {

if (rem[i] == 1) { // Perform XOR only if the leading bit is 1

for (k = 0; k < gen\_size; k++) {

rem[i + k] ^= gen[k]; // XOR operation for division

}

}

}

}

// Function to copy the frame for display

void copyFrame() {

for (i = 0; i < j; i++) {

rem[i] = temp[i]; // Copying temp values to rem array

printf("%d", temp[i]); // Printing the frame

}

printf("\n");

}

int main() {

char poly[50], gen\_str[GEN\_SIZE]; // Arrays to store input polynomials

printf("\t\t\*\*\* CYCLIC REDUNDANCY CHECK-12 \*\*\*\n\n");

// Initialize arrays with zero

memset(frame, 0, sizeof(frame));

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

memset(temp, 0, sizeof(temp));

memset(gen, 0, sizeof(gen));

// Input polynomial (binary string)

printf("Enter the polynomial in binary (e.g., 110101): ");

fgets(poly, sizeof(poly), stdin);

int poly\_len = strlen(poly) - 1; // Adjust for newline character

// Convert polynomial string to binary array

for (i = 0; i < poly\_len; i++) {

if (poly[i] == '0' || poly[i] == '1') {

frame[i] = poly[i] - '0';

} else {

printf("Invalid polynomial! Exiting...\n");

return 1;

}

}

// Input generator polynomial

printf("Enter the generator polynomial (e.g., 1101): ");

fgets(gen\_str, sizeof(gen\_str), stdin);

gen\_size = strlen(gen\_str) - 1; // Adjust for newline character

// Convert generator string to binary array

for (i = 0; i < gen\_size; i++) {

if (gen\_str[i] == '0' || gen\_str[i] == '1') {

gen[i] = gen\_str[i] - '0';

} else {

printf("Invalid generator! Exiting...\n");

return 1;

}

}

// Compute final frame size after appending zeros

j = poly\_len + gen\_size - 1;

for (i = 0; i < poly\_len; i++) {

temp[i] = frame[i]; // Copy original polynomial

}

for (i = poly\_len; i < j; i++) {

temp[i] = 0; // Append zeros

}

// Display frame after appending zeros

printf("\nFRAME after appending 0's: ");

copyFrame();

// Compute CRC remainder

computeCRC();

// Append remainder to the frame

printf("\nThe REMAINDER is: ");

for (i = poly\_len; i < j; i++) {

temp[i] = rem[i];

printf("%d", rem[i]);

}

printf("\n");

// Simulate data transmission

printf("\nTransmitting FRAME...\n\nTransmitted FRAME is: ");

copyFrame();

// Check for errors at receiver

printf("\nChecking for errors...\n\nReceived FRAME: ");

copyFrame();

computeCRC();

// Display remainder and check for errors

printf("\nThe remainder is: ");

int error = 0;

for (i = poly\_len; i < j; i++) {

printf("%d", rem[i]);

if (rem[i] != 0) {

error = 1; // Error detected

}

}

printf("\n");

if (!error)

printf("DATA SENT SUCCESSFULLY\n");

else

printf("ERROR DETECTED IN TRANSMISSION\n");

return 0;

}

**Output:**

\*\*\* CYCLIC REDUNDANCY CHECK-12 \*\*\*

Enter the polynomial in binary (e.g., 110101): 110100

Enter the generator polynomial (e.g., 1101): 1011

FRAME after appending 0's: 110100000

The REMAINDER is: 100

Transmitting FRAME...

Transmitted FRAME is: 110100100

Checking for errors...

Received FRAME: 110100100

The remainder is: 000

DATA SENT SUCCESSFULLY

1. What is CRC?

A cyclic redundancy check (CRC) is an [error-detecting code](https://en.wikipedia.org/wiki/Error_detection_and_correction) commonly used in digital [networks](https://en.wikipedia.org/wiki/Telecommunications_network) and storage devices to detect accidental changes to raw data. Blocks of data entering these systems get a short check value attached, based on the remainder of a [polynomial division](https://en.wikipedia.org/wiki/Polynomial_long_division) of their contents. On retrieval, the calculation is repeated and, in the event the check values do not match, corrective action can be taken against data corruption. CRCs can be used for [error correction](https://en.wikipedia.org/wiki/Error_correcting_code) .

1. What is the use of CRC?

The use of cyclic codes, which encode messages by adding a fixed-length check value, for the purpose of error detection in communication networks.

1. Name the CRC standards for generator polynomial?

CRC-12: **x12+x11+x3+x2+x+1**

CRC- 16: **X16+x15+x2+1**

CRC-CCITT**: x16+x12+x5+1**

1. How do you convert generator polynomial into binary form?

The generator polynomial is converted into the binary form by considering coefficients of polynomial

EX: x12+x11+x3+x2+x1+1 = 1100000001111

1. Define checksum?

A checksum is a simple type of redundancy check that is used to detect errors in data

1. How do you perform binary division operation in CRC?

