**CYCLE 2:**

**Experiment 13:**

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

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

Code:

#include <stdio.h>

#include <string.h>

// CRC-CCITT polynomial: x^16 + x^12 + x^5 + 1 (0x1021)

//#define CRC\_POLY 0x1021

// Function to perform bitwise XOR on binary strings

void binaryXOR(char \*result, const char \*a, const char \*b) {

for (int i = 0; i < 16; i++) {

result[i] = (a[i] == b[i]) ? '0' : '1';

}

result[16] = '\0';

}

// Function to calculate CRC-CCITT checksum

void calculateCRC(const char \*data, int length, char \*checksum) {

char crc[17];

for (int i = 0; i < 16; i++) {

crc[i] = '0';

}

crc[16] = '\0';

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

for (int j = 0; j < 8; j++) {

char msb = crc[0];

for (int k = 0; k < 16; k++) {

crc[k] = crc[k + 1];

}

crc[15] = '0';

if (msb == '1') {

char temp[17];

binaryXOR(temp, crc, "10001000000100001"); // CRC\_POLY in binary

strcpy(crc, temp);

}

}

crc[15] = (data[i] == '1') ? '1' : '0';

}

strcpy(checksum, crc);

}

int main() {

char data[100]; // Replace with your actual data

printf("Enter data in binary: ");

scanf("%s", data);

int dataLength = strlen(data);

char checksum[17];

calculateCRC(data, dataLength, checksum);

printf("Calculated CRC: %s\n", checksum);

// Simulating error by changing a bit

// data[2] ^= 0x01; // Uncomment this line to introduce an error

// Verify the received data

char receivedChecksum[17];

printf("Enter received CRC: ");

scanf("%s", receivedChecksum);

if (strcmp(receivedChecksum, checksum) == 0) {

printf("Data is error-free.\n");

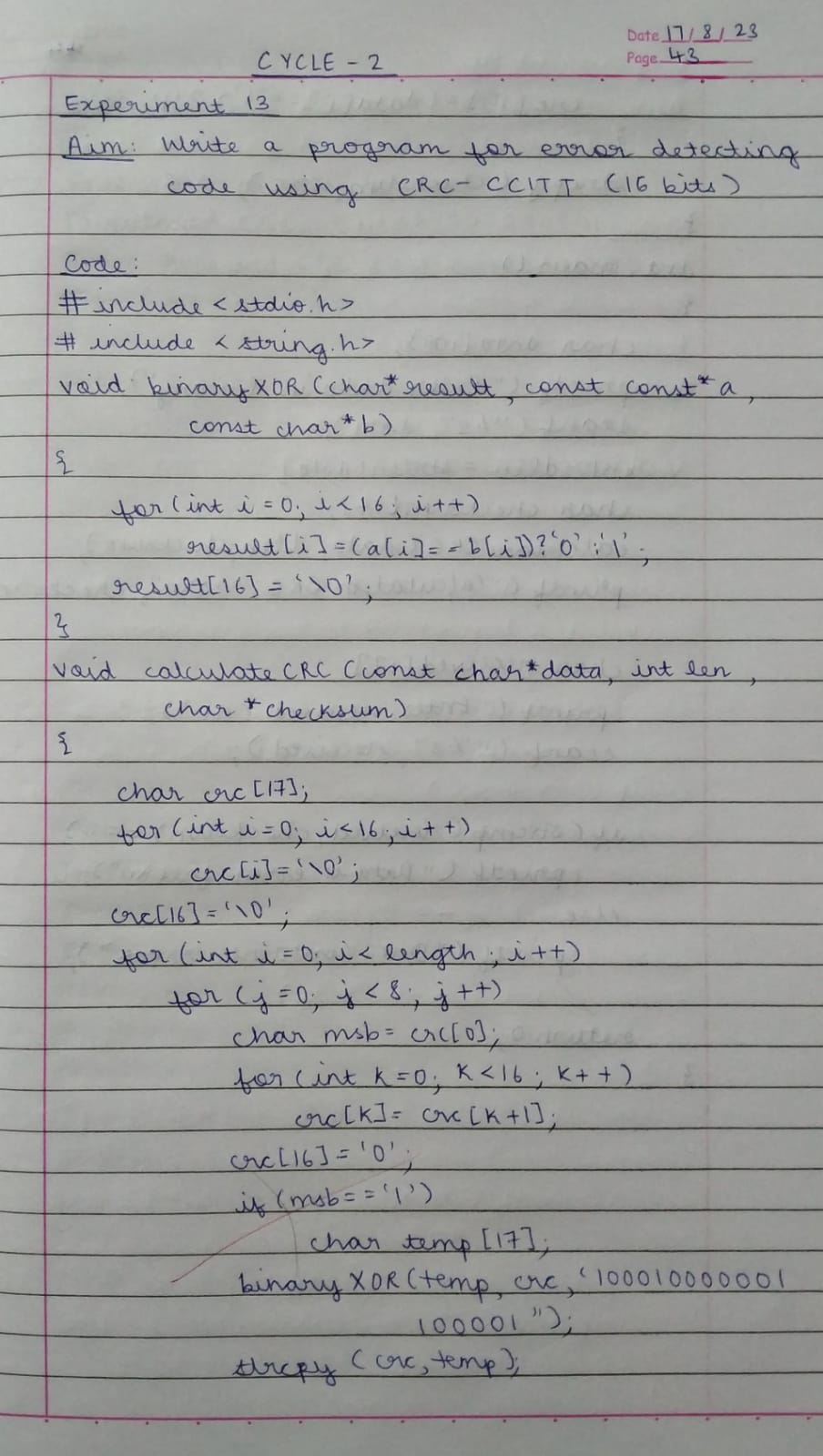
} else {

printf("Data contains errors.\n");

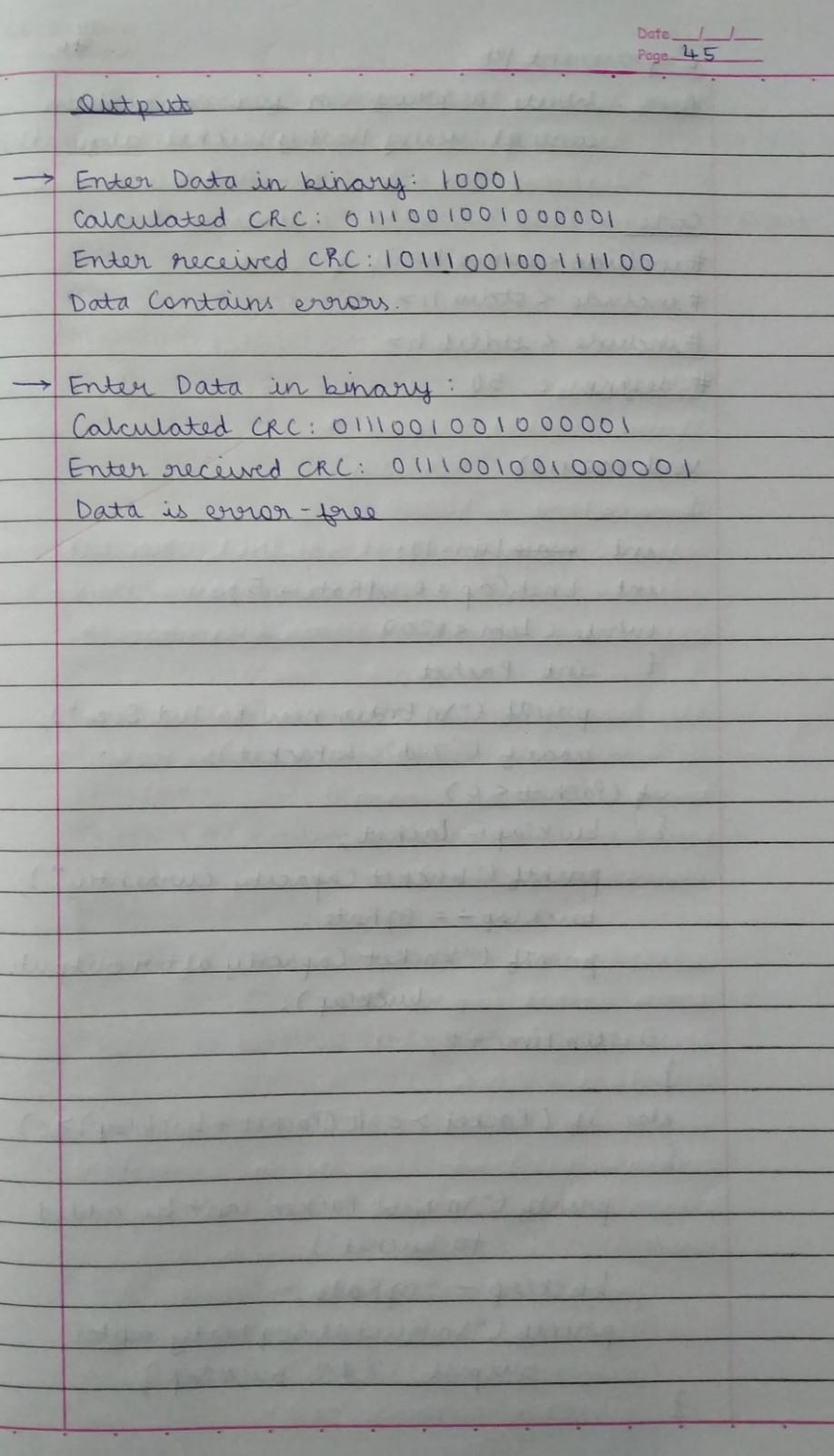
}

return 0;

Observation:







Output:

