Assignment no 03

#include <stdio.h>

// Function to calculate parity bits

void generateHammingCode(int data[], int code[]) {

// Place data bits at non-parity positions: 3,5,6,7,9,10,11 (1-based)

code[2] = data[0];

code[4] = data[1];

code[5] = data[2];

code[6] = data[3];

code[8] = data[4];

code[9] = data[5];

code[10] = data[6];

// Calculate parity bits at positions 1, 2, 4, and 8 (0-based: 0,1,3,7)

code[0] = code[2] ^ code[4] ^ code[6] ^ code[8] ^ code[10];

code[1] = code[2] ^ code[5] ^ code[6] ^ code[9] ^ code[10];

code[3] = code[4] ^ code[5] ^ code[6];

code[7] = code[8] ^ code[9] ^ code[10];

}

// Function to detect and correct single-bit errors

int detectAndCorrect(int code[]) {

int p1 = code[0] ^ code[2] ^ code[4] ^ code[6] ^ code[8] ^ code[10];

int p2 = code[1] ^ code[2] ^ code[5] ^ code[6] ^ code[9] ^ code[10];

int p4 = code[3] ^ code[4] ^ code[5] ^ code[6];

int p8 = code[7] ^ code[8] ^ code[9] ^ code[10];

int errorPos = p8 \* 8 + p4 \* 4 + p2 \* 2 + p1 \* 1;

return errorPos;

}

int main() {

int data[7];

int code[11] = {0};

printf("SENDER SIDE:\n");

printf("Enter 7 data bits (space-separated, e.g., 1 0 1 1 0 0 1): ");

for (int i = 0; i < 7; i++) {

scanf("%d", &data[i]);

}

generateHammingCode(data, code);

printf("Generated 11-bit Hamming Code (to send): ");

for (int i = 0; i < 11; i++) {

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

}

printf("\n\nRECEIVER SIDE:\n");

int receivedCode[11];

printf("Enter the 11-bit Hamming code received (space-separated): ");

for (int i = 0; i < 11; i++) {

scanf("%d", &receivedCode[i]);

}

int errorPos = detectAndCorrect(receivedCode);

if (errorPos == 0) {

printf("\nNo error detected in received data.\n");

} else {

printf("\nError detected at position: %d\n", errorPos);

receivedCode[errorPos - 1] ^= 1; // Correct the error

printf("Corrected Code: ");

for (int i = 0; i < 11; i++) {

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

}

printf("\n");

}

return 0;

}

OUTPUT

