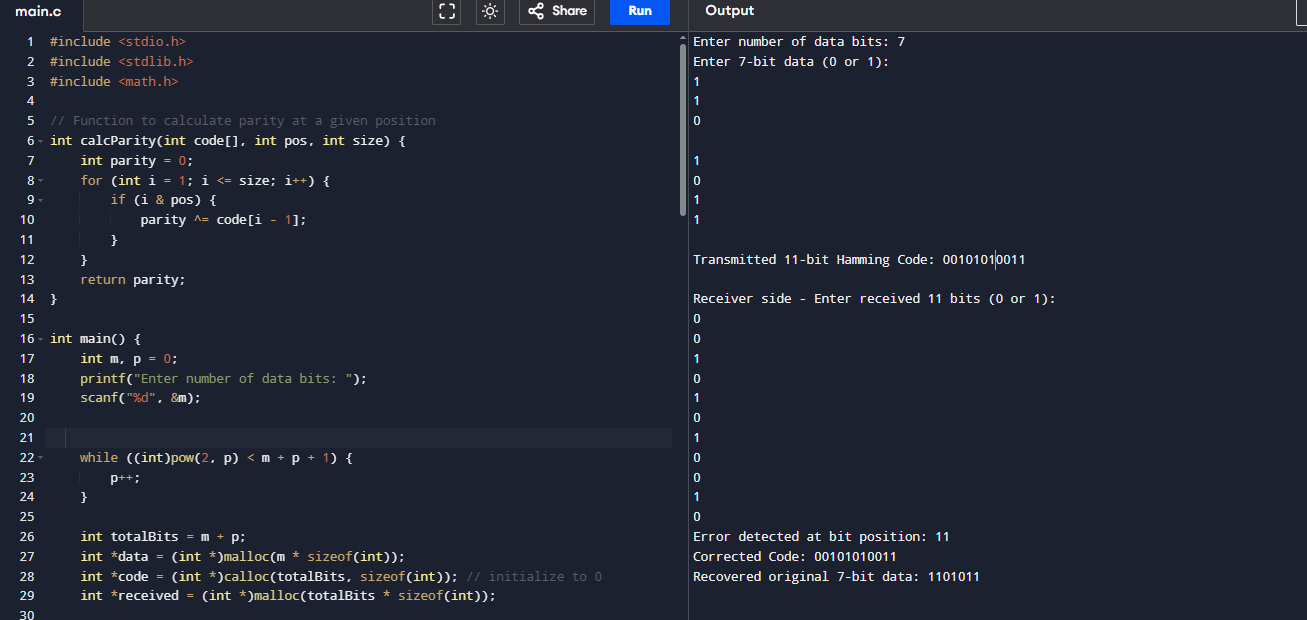
Title : hamming code



#include <stdio.h>

// Function to calculate parity for a given position

int calcParity(int code[], int pos, int size) {

int parity = 0;

for (int i = 1; i <= size; i++) {

if (i & pos) {

parity ^= code[i - 1];

}

}

return parity;

}

int main() {

int data[7], code[11] = {0}, i, j = 0;

printf("Enter 7-bit data (only 0 or 1):\n");

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

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

if (data[i] != 0 && data[i] != 1) {

printf("Invalid input. Only 0 or 1 allowed.\n");

return 1;

}

}

// Insert data bits into 11-bit array (skip parity positions: 1,2,4,8)

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

if (i == 0 || i == 1 || i == 3 || i == 7) {

continue; // parity positions

}

code[i] = data[j++];

}

// Calculate parity bits and insert

code[0] = calcParity(code, 1, 11); // P1

code[1] = calcParity(code, 2, 11); // P2

code[3] = calcParity(code, 4, 11); // P4

code[7] = calcParity(code, 8, 11); // P8

printf("\nTransmitted 11-bit Hamming Code: ");

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

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

}

printf("\n");

// Simulate receiver side

int received[11];

printf("\nReceiver side - Enter received 11 bits (with or without error):\n");

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

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

if (received[i] != 0 && received[i] != 1) {

printf("Invalid input.\n");

return 1;

}

}

// Detect error

int p1 = calcParity(received, 1, 11);

int p2 = calcParity(received, 2, 11);

int p4 = calcParity(received, 4, 11);

int p8 = calcParity(received, 8, 11);

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

if (errorPos == 0) {

printf("No error detected.\n");

} else {

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

received[errorPos - 1] ^= 1; // Correct the bit

printf("Corrected code: ");

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

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

}

printf("\n");

}

// Extract original 7-bit data from corrected code

int original[7];

j = 0;

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

if (i == 0 || i == 1 || i == 3 || i == 7) continue;

original[j++] = received[i];

}

printf("Recovered original 7-bit data: ");

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

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

}

printf("\n");

return 0;