ITC Coding Assignment
Name- Anmol Agrawal
Roll _No - 122CS0300

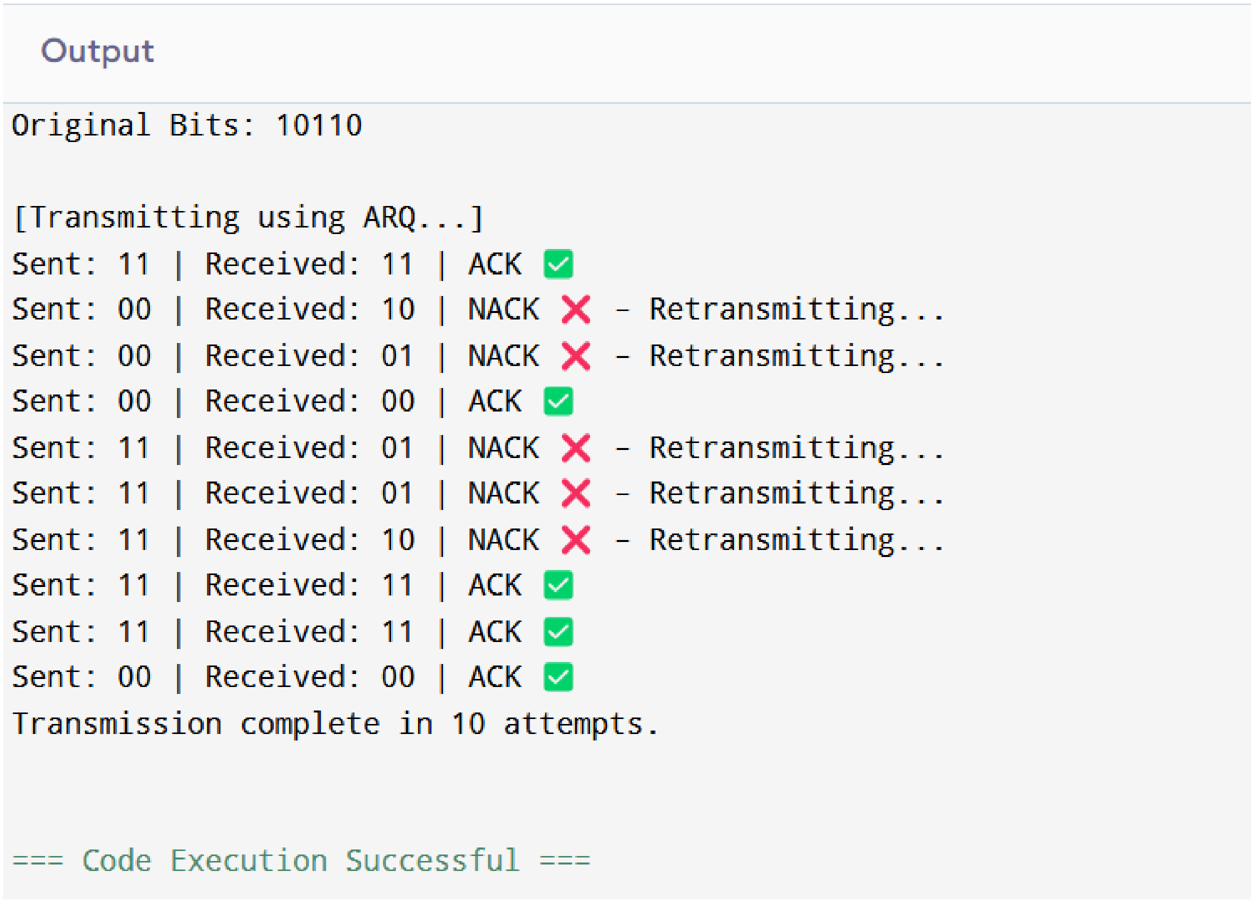
<u>Implementation of ARQ(stop and wait protocol scheme)</u>

```
#include <iostream>
#include <vector>
#include <cstdlib>
#include <ctime>
using namespace std;

// ----- Convolutional Encoder -----

pair<int, int> encodeBits(const vector<int>& shiftRegister) {
    int G1[] = {1, 1, 1}; // 111
    int G2[] = {1, 0, 1}; // 101
    int out1 = 0, out2 = 0;

    for (int i = 0; i < 3; i++) {
        out1 ^= (shiftRegister[i] & G1[i]);
        out2 ^= (shiftRegister[i] & G2[i]);
    }

    return {out1, out2};
}

vector<int> convolutionalEncode(const vector<int>& inputBits) {
    vector<int> outputBits;
    vector<int> shiftRegister(3, 0);
    for (int bit : inputBits) {
        shiftRegister[2] = shiftRegister[1];
        shiftRegister[1] = shiftRegister[0];
        shiftRegister[0] = bit;
        auto [out1, out2] = encodeBits(shiftRegister);
        outputBits.push_back(out1);
        outputBits.push_back(out2);
    }

    return outputBits;
}

// ----- Simulated Channel -----

vector<int> transmitWithNoise(const vector<int>& bits, double
errorProb) {
    vector<int> noisyBits = bits;
    for (int& bit : noisyBits) {
        double randVal = (double) rand() / RAND_MAX;
        if (randVal < errorProb) {
            bit ^= 1;  // flip bit
        }
    }
    return noisyBits;
}

// ----- Basic Decoder (Hard Decision) -----

bool isDataCorrect(const vector<int>& sent, const vector<int>&
received) {
    return sent == received;
```

```
}

// ----- ARQ Mechanism -----

void transmitWithARQ(const vector<int>& data, double errorProb) {
    cout << "\n[Transmitting using ARQ...]\n";
    int attempts = 0;
    for (int bit : data) {
        vector<int> input = {bit};
        vector<int> encoded = convolutionalEncode(input);

        bool ackReceived = false;
        while (!ackReceived) {
            attempts++;
            vector<int> received = transmitWithNoise(encoded, errorProb);

            cout << "Sent: ";
            for (int b : encoded) cout << b;
            cout << " | Received: ";
            for (int b : received) cout << b;

            if (isDataCorrect(encoded, received)) {
                cout << " | ACK ✅\n";
                ackReceived = true;
            } else {
                cout << " | NACK ❌ – Retransmitting...\n";
            }
        }
    }
    cout << "Transmission complete in " << attempts << " attempts.\n";
}
```

~~~cpp
// ----- Main Function -----

int main() {
    srand(time(0)); // Seed for randomness

    vector<int> inputBits = {1, 0, 1, 1, 0};
    double errorProbability = 0.2;  // 20% chance to flip each bit

    cout << "Original Bits: ";
    for (int b : inputBits) cout << b;
    cout << "\n";

    transmitWithARQ(inputBits, errorProbability);

    return 0;
}
```
~~~

<u>OUTPUT:</u>



```
Original Bits: 10110

[Transmitting using ARQ...]
Sent: 11 | Received: 11 | ACK ✅
Sent: 00 | Received: 10 | NACK ❌ - Retransmitting...
Sent: 00 | Received: 01 | NACK ❌ - Retransmitting...
Sent: 00 | Received: 00 | ACK ✅
Sent: 11 | Received: 01 | NACK ❌ - Retransmitting...
Sent: 11 | Received: 01 | NACK ❌ - Retransmitting...
Sent: 11 | Received: 10 | NACK ❌ - Retransmitting...
Sent: 11 | Received: 11 | ACK ✅
Sent: 11 | Received: 11 | ACK ✅
Sent: 00 | Received: 00 | ACK ✅
Transmission complete in 10 attempts.


=== Code Execution Successful ===
```