Computer networks crc practical

```
#include <iostream>
using namespace std;

string xor1(string a, string b)
{

    // Initialize result
    string result = "";

    int n = b.length();

    // Traverse all bits, if bits are
    // same, then XOR is 0, else 1
    for (int i = 1; i < n; i++) {
        if (a[i] == b[i])
            result += "0";
        else
            result += "1";
    }
    return result;
}

// Performs Modulo-2 division
string mod2div(string dividend, string divisor)
{
    int pick = divisor.length();

    string tmp = dividend.substr(0, pick);

    int n = dividend.length();

    while (pick < n) {
        if (tmp[0] == '1')
```

```
            tmp = xor1(divisor, tmp) + dividend[pick];
        else
            tmp = xor1(std::string(pick, '0'), tmp)
                + dividend[pick];

        pick += 1;
    }

    if (tmp[0] == '1')
        tmp = xor1(divisor, tmp);
    else
        tmp = xor1(std::string(pick, '0'), tmp);

    return tmp;
}


void encodeData(string data, string key)
{
    int l_key = key.length();

    // Appends n-1 zeroes at end of data
    string appended_data
        = (data + std::string(l_key - 1, '0'));

    string remainder = mod2div(appended_data, key);

    // Append remainder in the original data
    string codeword = data + remainder;
    cout << "Remainder : " << remainder << "\n";
    cout << "Encoded Data (Data + Remainder) :" << codeword
        << "\n";
}

void receiver(string data, string key)
{
    string currxor
        = mod2div(data.substr(0, key.size()), key);
    int curr = key.size();
    while (curr != data.size()) {
```

```
        if (currxor.size() != key.size()) {
            currxor.push_back(data[curr++]);
        }
        else {
            currxor = mod2div(currxor, key);
        }
    }
    if (currxor.size() == key.size()) {
        currxor = mod2div(currxor, key);
    }
    if (currxor.find('1') != string::npos) {
        cout << "there is some error in data" << endl;
    }
    else {
        cout << "correct message received" << endl;
    }
}
// Driver code
int main()
{
    string data = "100100";
    string key = "1101";
    cout << "Sender side..." << endl;
    encodeData(data, key);

    cout << "\nReceiver side..." << endl;
    receiver(data+mod2div(data+std::string(key.size() - 1,
'0'),key), key);

    return 0;
}
```

```
Sender side...
Remainder : 001
Encoded Data (Data + Remainder) :100100001

Receiver side...
correct message received
```