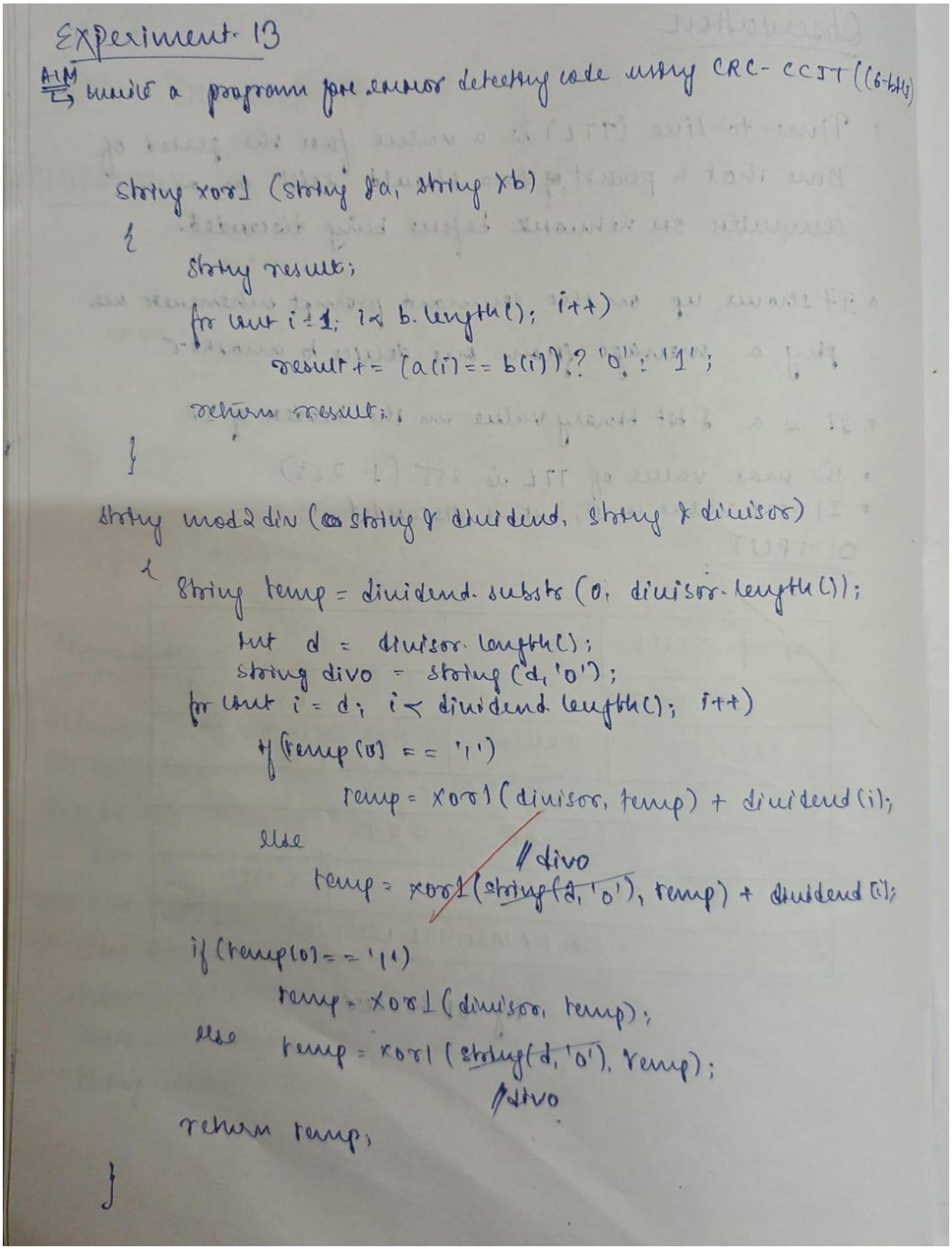
Experiment 13

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



```
Experiment 13
AIM: write a program for error detecting code using CRC-CCITT (16-bits)

string xor1 (string a, string b)
{
    string result;
    for (int i=1; i< b.length(); i++)
        result += (a[i]==b[i])? '0': '1';
    return result;
}

string mod2div (string dividend, string divisor)
{
    string temp = dividend.substr(0, divisor.length());
    int d = divisor.length();
    string divo = string(d, '0');
    for (int i = d; i < dividend.length(); i++)
        if (temp[0] == '1')
            temp = xor1(divisor, temp) + dividend[i];
        else
            temp = xor1(string(d, '0'), temp) + dividend[i];   // divo

    if (temp[0]=='1')
        temp = xor1(divisor, temp);
    else
        temp = xor1(string(d, '0'), temp);                     // divo
    return temp;
}
```

```
String encodeData (string& data, string & key)
{
    string appended = data + string (key.size()-1, '0');
    string remainder = mod2div (appended, key);
    cout << "remainder : " << remainder << endl;
    string codeword = data + remainder;
    cout << codeword;
    return codeword;
}

void receiver(string& codeword, string & key)
{
    string received;
    cin >> received;
    string checksum = mod2div(received.substr(0, key.size()), key);
    int curr = key.size();
    while (curr != received.size())
    {
        if (checksum.size != key.size())
            checksum.push_back(received[curr]);
            curr++;
        else
            checksum = mod2div(checksum, key);
    }
    if (checksum.size() == key.size())
        checksum = mod2div (checksum, key);
    if (checksum.find('1') != string::npos) // string::npos
        cout << "error"
    else cout << "No error" << endl;
}
```

```
int main()
{
    string data, key;
    cout << "enter data"
     cin>> data;
    cout << "enter key"
     cin >> key;

    cout << "sender's side" << endl;
    string codeword = encodeData (data, key);

    cout << "receiver's end" << endl;
    receiver (codeword, key);

    return 0;
}
```

OUTPUT

enter data: 10101100 11

enter key: 1011

Sender's side:

codeword: 10101100 11010

Receiver's end:

10101100 11010

No error

PROGRAM:

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

string xor1(const string &a, const string &b) {
    string result;
    for (size_t i = 1; i < b.length(); ++i)
        result += (a[i] == b[i]) ? '0' : '1';
    return result;
}

string mod2div(const string &dividend, const string &divisor) {
    string tmp = dividend.substr(0, divisor.length());
    size_t pick = divisor.length();

    for (size_t i = pick; i < dividend.length(); ++i) {
        if (tmp[0] == '1')
            tmp = xor1(divisor, tmp) + dividend[i];
        else
            tmp = xor1(string(pick, '0'), tmp) + dividend[i];
    }

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

    return tmp;
}

string encodeData(const string &data, const string &key) {
    string appended_data = data + string(key.size() - 1, '0');
    string remainder = mod2div(appended_data, key);
    cout << "Remainder: " << remainder << "\n";
    string codeword = data + remainder;
    cout << "Encoded Data (Data + Remainder): " << codeword << "\n";
    return codeword;
}

void receiver(const string &codeword, const string &key) {
    string received_codeword;
    cout << "Enter received codeword: ";
    cin >> received_codeword;

    string currxor = mod2div(received_codeword.substr(0, key.size()), key);
    size_t curr = key.size();

    while (curr != received_codeword.size()) {
        if (currxor.size() != key.size())
            currxor.push_back(received_codeword[curr++]);
        else
            currxor = mod2div(currxor, key);
    }
```

```
    if (currxor.size() == key.size())
        currxor = mod2div(currxor, key);

    cout << ((currxor.find('1') != string::npos) ? "There is an error in data" : "Correct
message received") << endl;
}

int main() {
    string data, key;
    cout << "Enter data: ";
    cin >> data;
    cout << "Enter key: ";
    cin >> key;

    cout << "Sender side..." << endl;
    string codeword = encodeData(data, key);

    // cout << "\nReceiver side..." << endl;
    // receiver(codeword, key);
    cout<<"enter recieved..";
    string recieved;
    cin>>recieved;

    string res = mod2div(recieved, key);
    int flag = 1;
    for(auto x : res) {
        if(x=='1') {
            cout<<"error";
            flag = 0;
        }
    }
    if(flag) {
        cout<<"correct";
    }


    return 0;
}
```

OUTPUT :

```
PS C:\Users\anosh\OneDrive\Desktop\CPP in VS> cd "c:\Users\anosh
Enter data: 10101011001101
Enter key: 1011
Sender side...
Remainder: 100
Encoded Data (Data + Remainder): 10101011001101100
enter recieved..10101011001101100
correct
PS C:\Users\anosh\OneDrive\Desktop\CPP in VS>
```