CSE 1004

NETWORK AND COMMUNICATION

Assessment – 2

L23+L24 | PLBG17

WINTER SEMESTER 2020–21

by

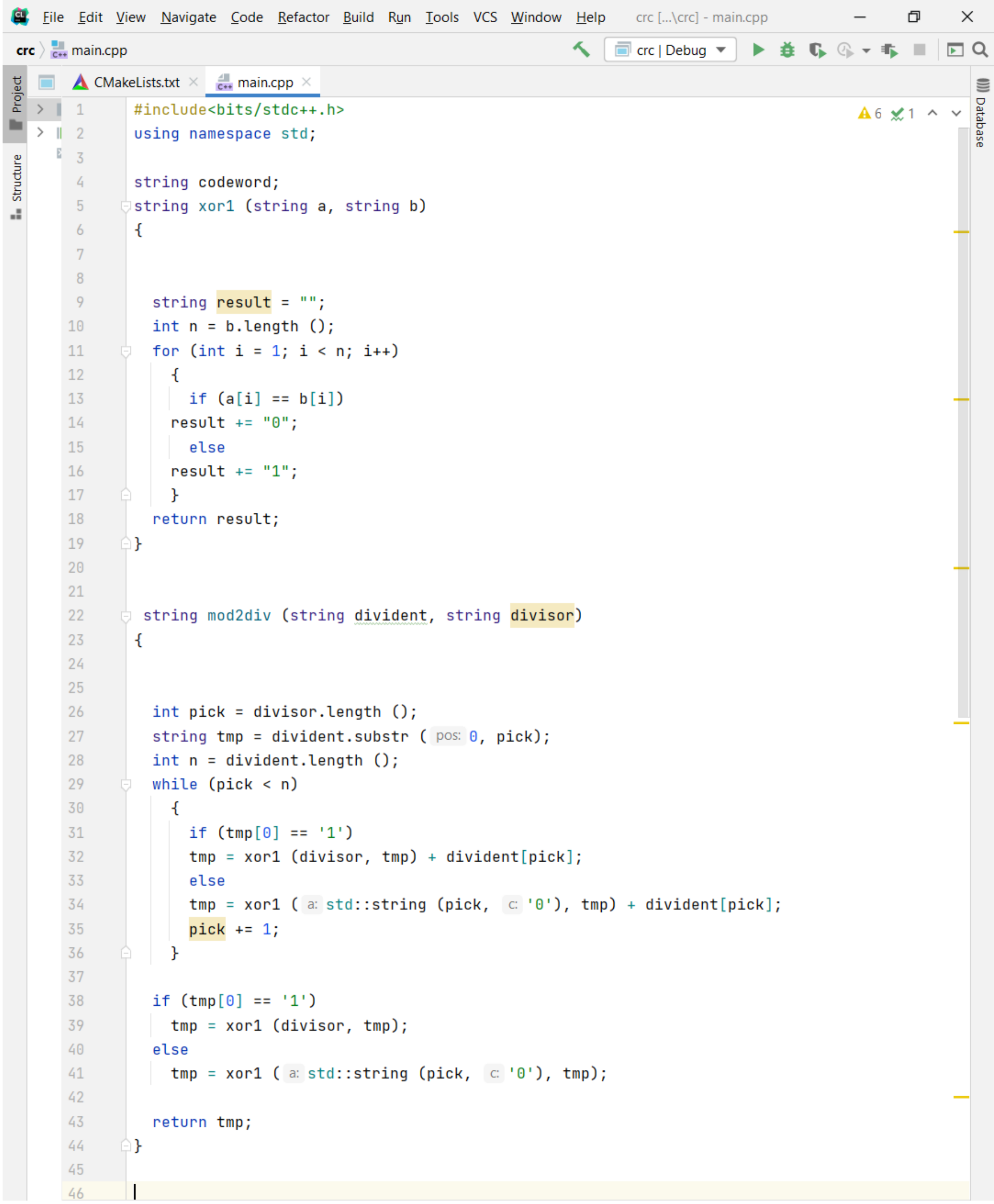
SHARADINDU ADHIKARI
19BCE2105

PART 1

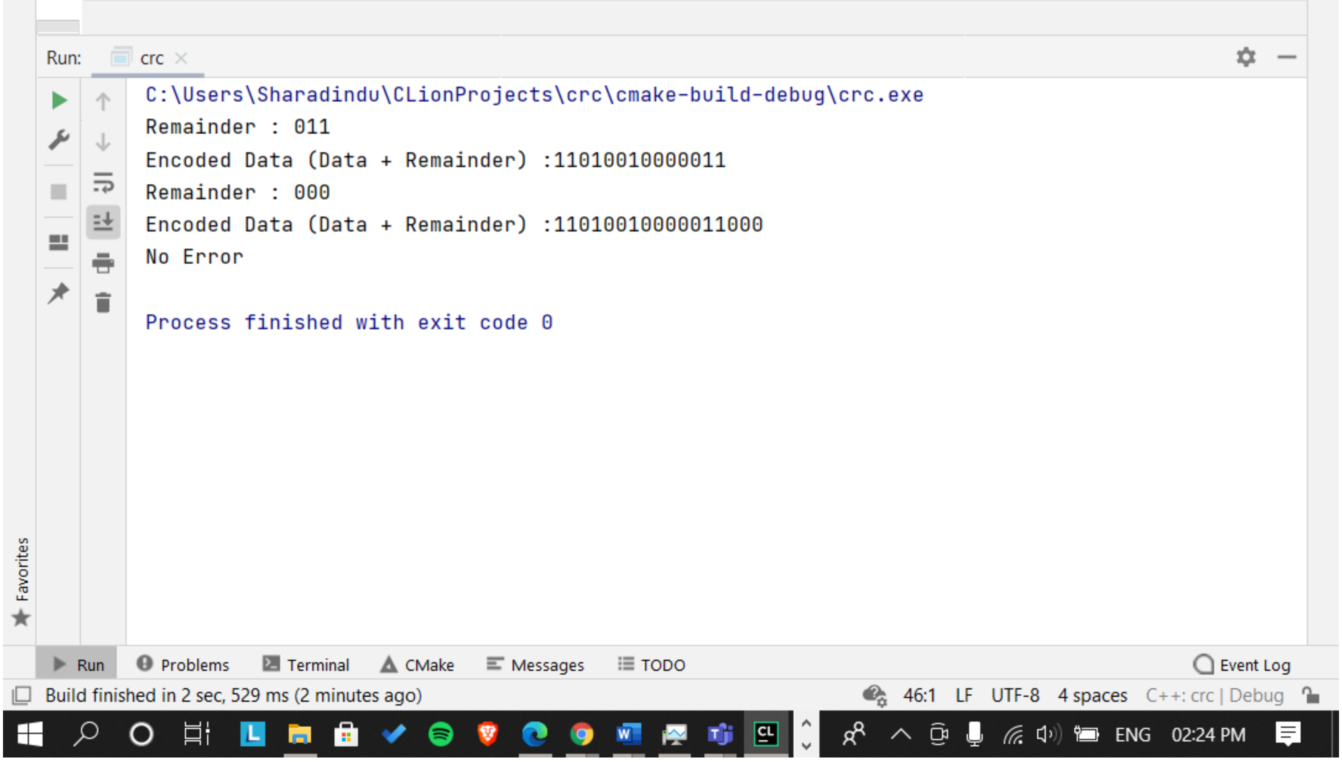
Crc

```
#include<bits/stdc++.h>
using namespace std;

string codeword;
string xor1 (string a, string b)
{


    string result = "";
    int n = b.length ();
    for (int i = 1; i < n; i++)
    {
        if (a[i] == b[i])
            result += "0";
        else
            result += "1";
    }
    return result;
}


string mod2div (string divident, string divisor)
{


    int pick = divisor.length ();
    string tmp = divident.substr (0, pick);
    int n = divident.length ();
    while (pick < n)
    {
        if (tmp[0] == '1')
            tmp = xor1 (divisor, tmp) + divident[pick];
        else
            tmp = xor1 (std::string (pick, '0'), tmp) + divident[pick];
        pick += 1;
    }

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

    return tmp;
}


string 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
    codeword = data + remainder;
    cout << "Remainder : " << remainder << "\n";
    cout << "Encoded Data (Data + Remainder) :" << codeword << "\n";
    return remainder;
}
```

```
int main ()
{
    string data = "11010010000";
    string divisor = "1001";
    string h="000";
    encodeData (data, divisor);
    string remainder = encodeData (codeword,divisor);

    if (remainder==h)//checking if message has error or not
        cout<<"No Error"<<endl;
    else
        cout<<"Error"<<endl;
    return 0;
}
```



```
#include<bits/stdc++.h>
using namespace std;

string codeword;
string xor1 (string a, string b)
{


    string result = "";
    int n = b.length ();
    for (int i = 1; i < n; i++)
        {
            if (a[i] == b[i])
        result += "0";
            else
        result += "1";
        }
    return result;
}


 string mod2div (string divident, string divisor)
{


    int pick = divisor.length ();
    string tmp = divident.substr ( pos: 0, pick);
    int n = divident.length ();
    while (pick < n)
        {
            if (tmp[0] == '1')
            tmp = xor1 (divisor, tmp) + divident[pick];
            else
            tmp = xor1 ( a: std::string (pick,  c: '0'), tmp) + divident[pick];
            pick += 1;
        }

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

    return tmp;
}

```



```
C:\Users\Sharadindu\CLionProjects\crc\cmake-build-debug\crc.exe
Remainder : 011
Encoded Data (Data + Remainder) :11010010000011
Remainder : 000
Encoded Data (Data + Remainder) :11010010000011000
No Error

Process finished with exit code 0
```

Build finished in 2 sec, 529 ms (2 minutes ago)

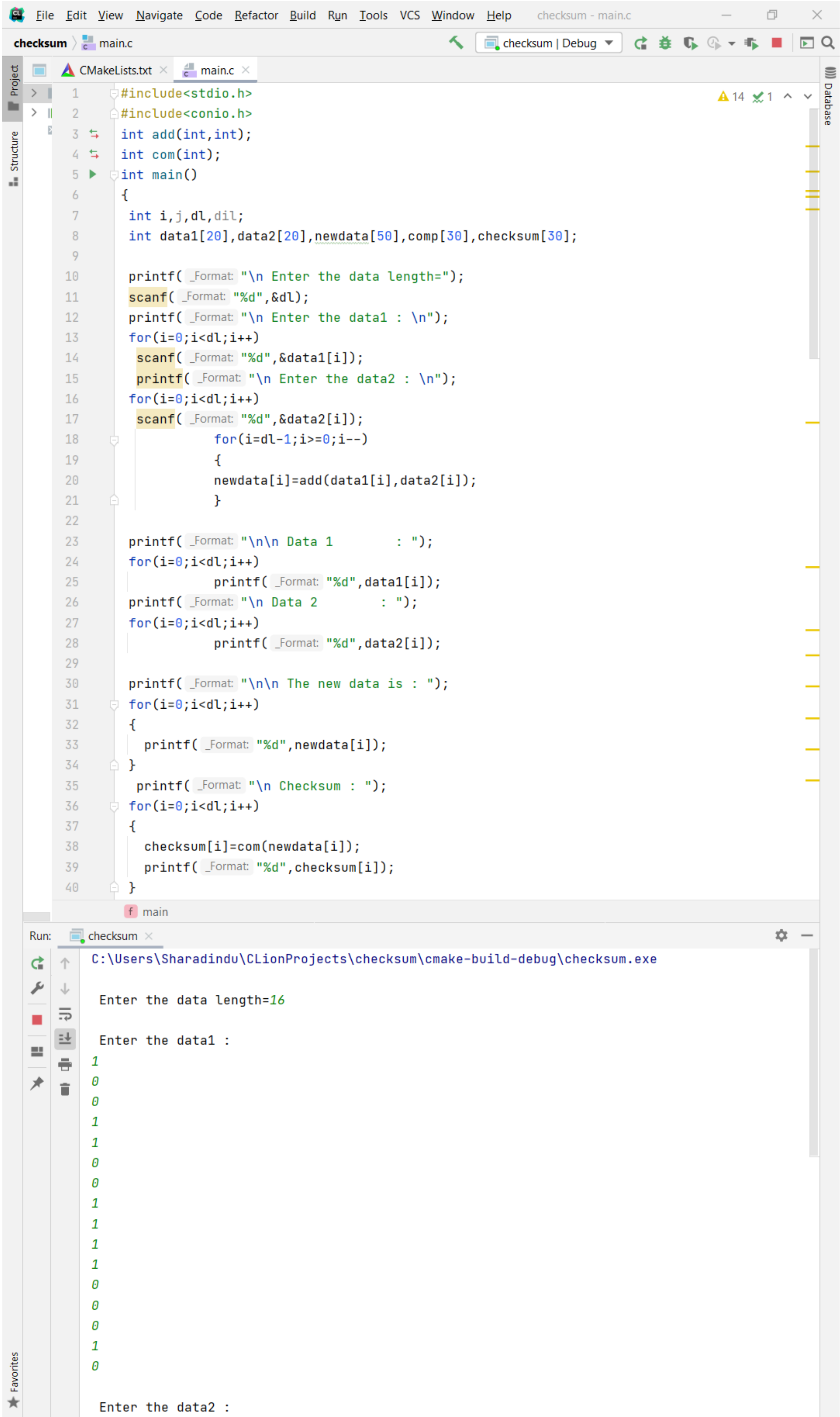
46:1 LF UTF-8 4 spaces C++: crc | Debug

ENG 02:24 PM

PART 2

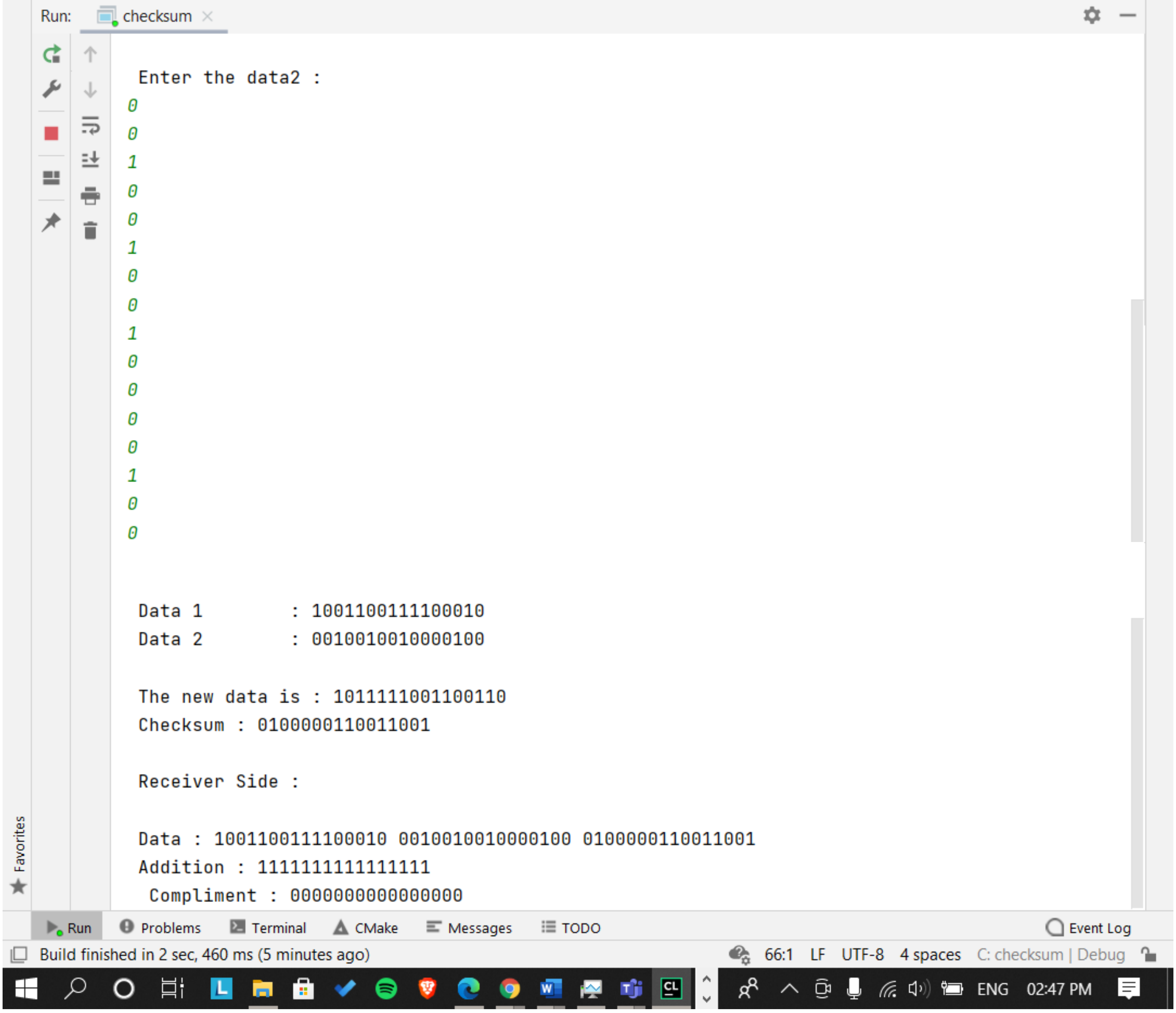
Checksum

```
#include<stdio.h>
#include<conio.h>
int add(int,int);
int com(int);
int main()
{
    int i,j,dl,dil;
    int data1[20],data2[20],newdata[50],comp[30],checksum[30];

    printf("\n Enter the data length=");
    scanf("%d",&dl);
    printf("\n Enter the data1 : \n");
    for(i=0;i<dl;i++)
        scanf("%d",&data1[i]);
    printf("\n Enter the data2 : \n");
    for(i=0;i<dl;i++)
        scanf("%d",&data2[i]);
    for(i=dl-1;i>=0;i--)
    {
        newdata[i]=add(data1[i],data2[i]);
    }

    printf("\n\n Data 1        : ");
    for(i=0;i<dl;i++)
        printf("%d",data1[i]);
    printf("\n Data 2        : ");
    for(i=0;i<dl;i++)
        printf("%d",data2[i]);

    printf("\n\n The new data is : ");
    for(i=0;i<dl;i++)
    {
        printf("%d",newdata[i]);
    }
    printf("\n Checksum : ");
    for(i=0;i<dl;i++)
    {
        checksum[i]=com(newdata[i]);
        printf("%d",checksum[i]);
    }

    printf("\n\n Receiver Side : \n");
    printf("\n Data : ");
    for(i=0;i<dl;i++)
        printf("%d",data1[i]);printf(" ");
    for(i=0;i<dl;i++)
        printf("%d",data2[i]);printf(" ");
    for(i=0;i<dl;i++)
        printf("%d",checksum[i]);

    printf("\n Addition : ");
    for(i=dl-1;i>=0;i--)
    {
        newdata[i]=add(newdata[i],checksum[i]);
    }
    for(i=0;i<dl;i++)
    {
        printf("%d",newdata[i]);
    }
    printf("\n  Compliment : ");
    for(i=0;i<dl;i++)
    {
        comp[i]=com(newdata[i]);
        printf("%d",comp[i]);
    }

    getch();
}

int add(int x, int y)
{
    static int carry=0;
    if(x==1 && y==1 && carry==0)
    {
        carry=1;
        return 0;
    }
    else if(x==1 && y==1 && carry==1)
    {
        carry=1;
        return 1;
    }
    else if(x==1 && y==0 && carry==0)
    {
        carry=0;
        return 1;
    }
    else if(x==1 && y==0 && carry==1)
    {
        carry=1;
        return 0;
    }
    else if(x==0 && y==1 && carry==0)
    {
        carry=0;
        return 1;
    }
    else if(x==0 && y==1 && carry==1)
    {
        carry=1;
        return 0;
    }
    else if(x==0 && y==0 && carry==0)
    {
        carry=0;
        return 0;
    }
    else
    {
        carry=0;
        return 1;
    }
}
int com(int a)
{
    if(a==0)
        return 1;
    else
        return 0;
}
```



```c
#include<stdio.h>
#include<conio.h>
int add(int,int);
int com(int);
int main()
{
 int i,j,dl,dil;
 int data1[20],data2[20],newdata[50],comp[30],checksum[30];

 printf("\n Enter the data length=");
 scanf("%d",&dl);
 printf("\n Enter the data1 : \n");
 for(i=0;i<dl;i++)
  scanf("%d",&data1[i]);
  printf("\n Enter the data2 : \n");
 for(i=0;i<dl;i++)
  scanf("%d",&data2[i]);
            for(i=dl-1;i>=0;i--)
            {
            newdata[i]=add(data1[i],data2[i]);
            }

 printf("\n\n Data 1        : ");
 for(i=0;i<dl;i++)
            printf("%d",data1[i]);
 printf("\n Data 2        : ");
 for(i=0;i<dl;i++)
            printf("%d",data2[i]);

 printf("\n\n The new data is : ");
 for(i=0;i<dl;i++)
 {
   printf("%d",newdata[i]);
 }
  printf("\n Checksum : ");
 for(i=0;i<dl;i++)
 {
   checksum[i]=com(newdata[i]);
   printf("%d",checksum[i]);
 }
```

```
C:\Users\Sharadindu\CLionProjects\checksum\cmake-build-debug\checksum.exe

 Enter the data length=16

 Enter the data1 :
1
0
0
1
1
0
0
1
1
1
1
0
0
0
1
0

 Enter the data2 :
```



```
 Enter the data2 :
0
0
1
0
0
1
0
0
1
0
0
0
0
1
0
0


 Data 1         : 1001100111100010
 Data 2         : 0010010010000100

 The new data is : 1011111001100110
 Checksum : 0100000110011001

 Receiver Side :

 Data : 1001100111100010 0010010010000100 0100000110011001
 Addition : 1111111111111111
  Compliment : 0000000000000000
```

Run
Problems
Terminal
CMake
Messages
TODO
Event Log
Build finished in 2 sec, 460 ms (5 minutes ago)
66:1 LF UTF-8 4 spaces C: checksum | Debug
ENG 02:47 PM

PART 3

Hamming Code

Instructions

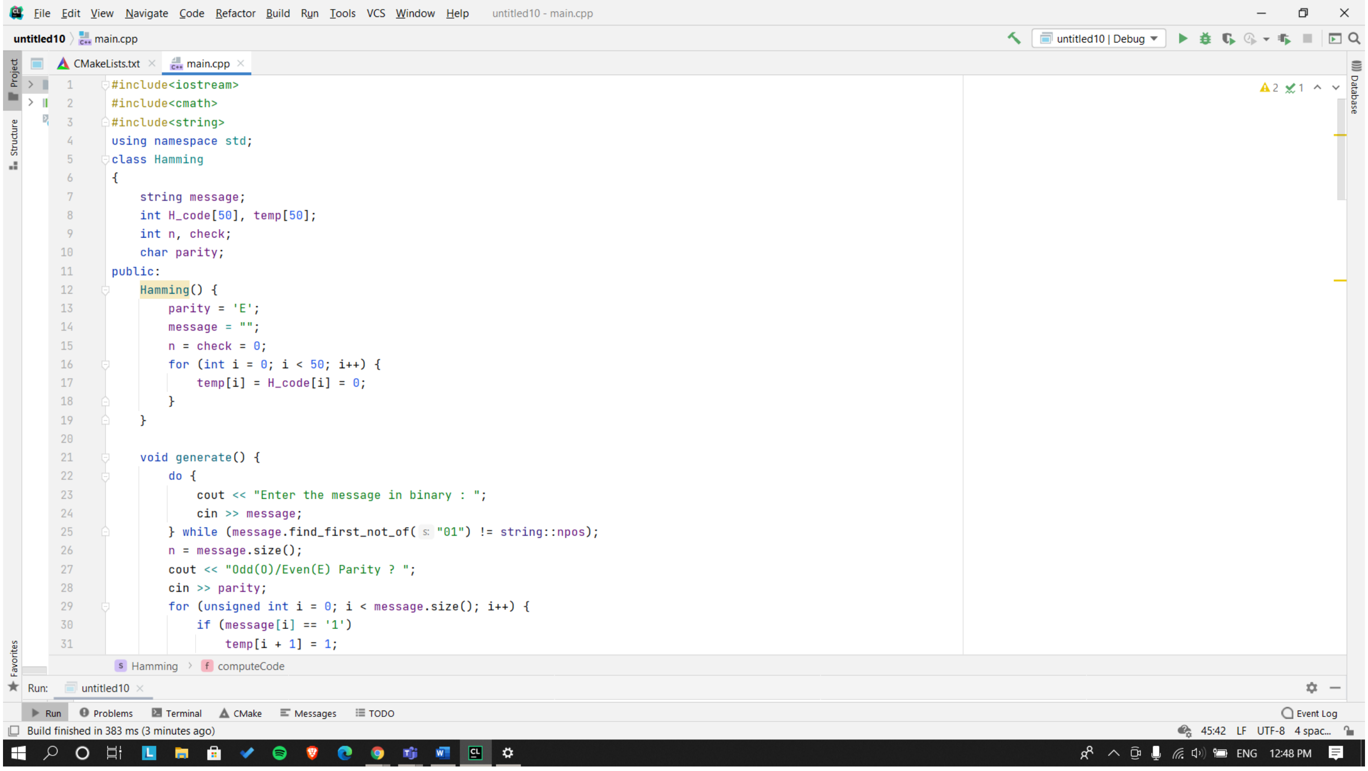
Input: Read the dataword at sender side, Read the code word at receiver side.
Output: No. of redundant bits, codeword at sender side. Error or no error, if it is error, Corrupted bit position at receiver side

Code:

```
#include<iostream>
#include<cmath>
#include<string>
using namespace std;
class Hamming
{
    string message;
    int H_code[50], temp[50];
    int n, check;
    char parity;
public:
    Hamming() {
        parity = 'E';
        message = "";
        n = check = 0;
        for (int i = 0; i < 50; i++) {
            temp[i] = H_code[i] = 0;
        }
    }

    void generate() {
        do {
            cout << "Enter the message in binary : ";
            cin >> message;
        } while (message.find_first_not_of("01") != string::npos);
        n = message.size();
        cout << "Odd(O)/Even(E) Parity ? ";
        cin >> parity;
        for (unsigned int i = 0; i < message.size(); i++) {
            if (message[i] == '1')
                temp[i + 1] = 1;
            else
                temp[i + 1] = 0;
        }
        computeCode();
    }
    void computeCode() {
        check = findr();
        cout << "Number of Redundant Bits : " << check << endl;
        cout << "Number of Bits in Codeword : " << n + check << endl;
        for (int i = (n + check), j = n; i > 0; i--) {
            if ((i & (i - 1)) != 0)
                H_code[i] = temp[j--];
            else
                H_code[i] = setParity(i);
        }
        cout << "Parity Bits - ";
        for (int i = 0; i < check; i++)
            cout << "P" << pow(2, i) << " : " << H_code[(int) pow(2, i)] << "\t";
        cout << endl;
        cout << "H_code :" << endl;
        for (int i = 1; i <= (n + check); i++)
            cout << H_code[i] << " ";
        cout << endl;
    }
    int findr() {
```

```
    for (int i = 1;; i++) {
        if (n + i + 1 <= pow(2, i))
            return i;
    }
}
int setParity(int x) {
    bool flag = true;
    int bit;
    if (x == 1) {
        bit = H_code[x + 2];
        for (int j = x + 3; j <= (n + check); j++) {
            if (j % 2) {
                bit ^= H_code[j];
            }
        }
    } else {
        bit = H_code[x + 1];
        for (int i = x; i <= (n + check); i++) {
            if (flag) {
                if (i == x || i == x + 1)
                    bit = H_code[x + 1];
                else
                    bit ^= H_code[i];
            }
            if ((i + 1) % x == 0)
                flag = !flag;
        }
    }
    if (parity == 'O' || parity == 'o')
        return !bit;
    else
        return bit;
}
void correct() {
    do {
        cout << "Enter the received codeword : ";
        cin >> message;
    } while (message.find_first_not_of("01") != string::npos);
    for (unsigned int i = 0; i < message.size(); i++) {
        if (message[i] == '1')
            H_code[i + 1] = 1;
        else
            H_code[i + 1] = 0;
    }
    detect();
}
void detect() {
    int position = 0;
    cout << "Parity Bits - ";
    for (int i = 0; i < check; i++) {
        bool flag = true;
        int x = pow(2, i);
        int bit = H_code[x];
        if (x == 1) {
            for (int j = x + 1; j <= (n + check); j++) {
                if (j % 2) {
                    bit ^= H_code[j];
                }
            }
        } else {
            for (int k = x + 1; k <= (n + check); k++) {
                if (flag) {
                    bit ^= H_code[k];
                }
                if ((k + 1) % x == 0)
                    flag = !flag;
            }
        }
        cout << "P" << x << ": " << bit << "\t";
```

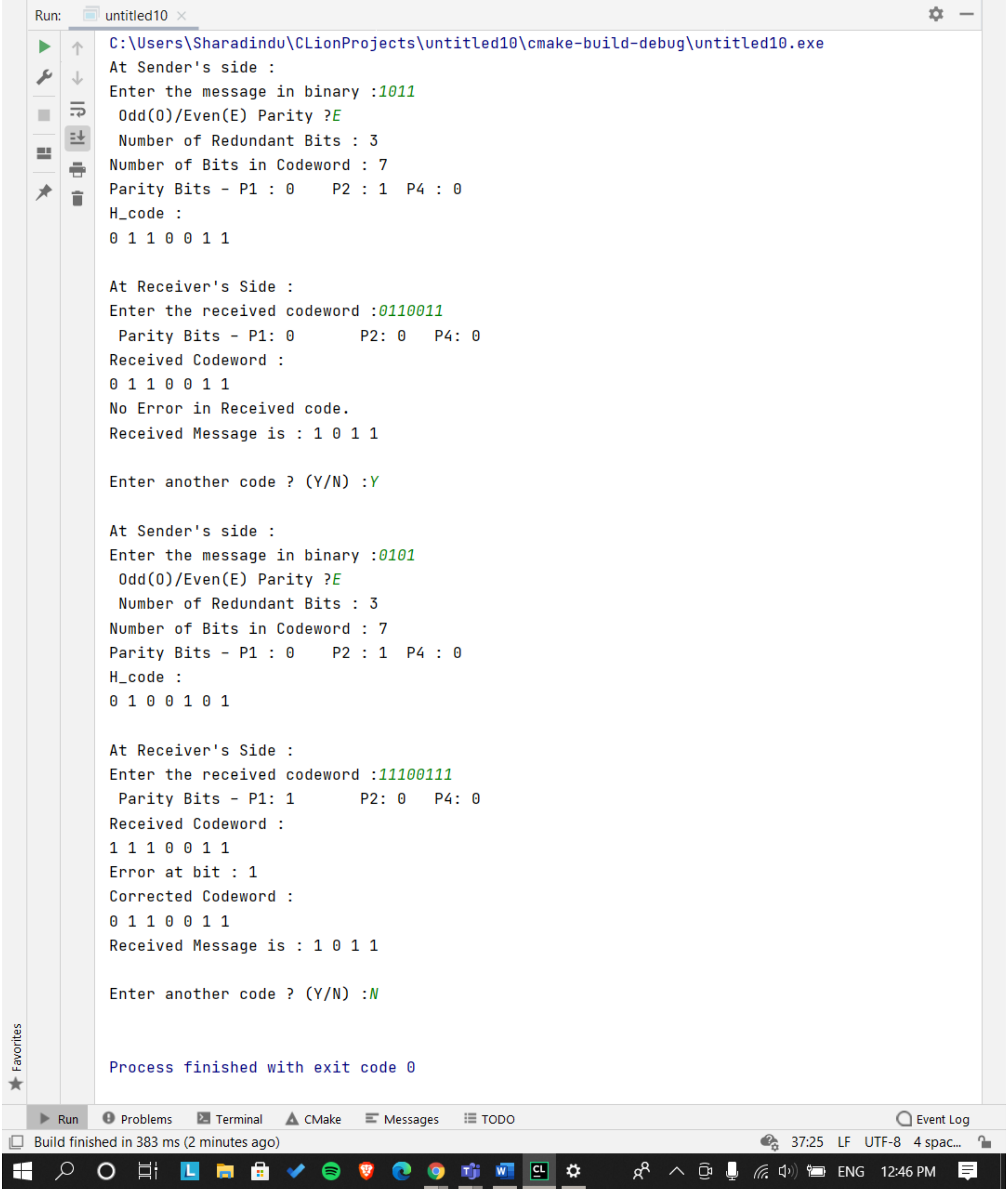
```
            if ((parity == 'E' || parity == 'e') && bit == 1)
                position += x;
            if ((parity == 'O' || parity == 'o') && bit == 0)
                position += x;
        }
        cout << endl << "Received Codeword :" << endl;
        for (int i = 1; i <= (n + check); i++)
            cout << H_code[i] << " ";
        cout << endl;
        if (position != 0) {
            cout << "Error at bit : " << position << endl;
            H_code[position] = !H_code[position];
            cout << "Corrected Codeword : " << endl;
            for (int i = 1; i <= (n + check); i++)
                cout << H_code[i] << " ";
            cout << endl;
        } else
            cout << "No Error in Received code." << endl;
        cout << "Received Message is : ";
        for (int i = 1; i <= (n + check); i++)
            if ((i & (i - 1)) != 0)
                cout << H_code[i] << " ";
        cout << endl;
    }
};
int main() {
    char choice;
    do {
        Hamming a;
        cout << "At Sender's side : " << endl;
        a.generate();
        cout << endl << "At Receiver's Side : " << endl;
        a.correct();
        cout << endl << "Enter another code ? (Y/N) : ";
        cin >> choice;
        cout << endl;
    } while (choice == 'y' || choice == 'Y');
    return 0;
}
```



```
#include<iostream>
#include<cmath>
#include<string>
using namespace std;
class Hamming
{
    string message;
    int H_code[50], temp[50];
    int n, check;
    char parity;
public:
    Hamming() {
        parity = 'E';
        message = "";
        n = check = 0;
        for (int i = 0; i < 50; i++) {
            temp[i] = H_code[i] = 0;
        }
    }

    void generate() {
        do {
            cout << "Enter the message in binary : ";
            cin >> message;
        } while (message.find_first_not_of( s: "01") != string::npos);
        n = message.size();
        cout << "Odd(O)/Even(E) Parity ? ";
        cin >> parity;
        for (unsigned int i = 0; i < message.size(); i++) {
            if (message[i] == '1')
                temp[i + 1] = 1;
```

Build finished in 383 ms (3 minutes ago)

OUTPUT:



```
C:\Users\Sharadindu\CLionProjects\untitled10\cmake-build-debug\untitled10.exe
At Sender's side :
Enter the message in binary :1011
 Odd(O)/Even(E) Parity ?E
 Number of Redundant Bits : 3
Number of Bits in Codeword : 7
Parity Bits - P1 : 0    P2 : 1  P4 : 0
H_code :
0 1 1 0 0 1 1

At Receiver's Side :
Enter the received codeword :0110011
 Parity Bits - P1: 0       P2: 0   P4: 0
Received Codeword :
0 1 1 0 0 1 1
No Error in Received code.
Received Message is : 1 0 1 1

Enter another code ? (Y/N) :Y

At Sender's side :
Enter the message in binary :0101
 Odd(O)/Even(E) Parity ?E
 Number of Redundant Bits : 3
Number of Bits in Codeword : 7
Parity Bits - P1 : 0    P2 : 1  P4 : 0
H_code :
0 1 0 0 1 0 1

At Receiver's Side :
Enter the received codeword :11100111
 Parity Bits - P1: 1       P2: 0   P4: 0
Received Codeword :
1 1 1 0 0 1 1
Error at bit : 1
Corrected Codeword :
0 1 1 0 0 1 1
Received Message is : 1 0 1 1

Enter another code ? (Y/N) :N


Process finished with exit code 0
```

Build finished in 383 ms (2 minutes ago)