CRC Program:

```
#include <iostream>
using namespace std;
int main()
{
   int i,j,k,l;
   int fs;
   cout<<"\n Enter Size of data: ";
   cin>>fs;
   int f[20];
   cout<<" Enter data:";
   for(i=0;i<fs;i++)
      cin>>f[i];
   int gs;
   cout<<"\n Enter key size: ";
   cin>>gs;
   int g[20];
   cout<<"\n Enter key:";
   for(i=0;i<gs;i++)
      cin>>g[i];
   cout<<"\n Sender Side:";
   cout<<"\n data: ";
   for(i=0;i<fs;i++)
      cout<<f[i];
   cout<<"\n key :";
   for(i=0;i<gs;i++)
      cout<<g[i];
   int rs=gs-1;
   cout<<"\n Number of 0's to be appended: "<<rs;
   for (i=fs;i<fs+rs;i++)
      f[i]=0;
```

```
int temp[20];
for(i=0;i<20;i++)
   temp[i]=f[i];
cout<<"\n Message after appending 0's :";
for(i=0; i<fs+rs;i++)
   cout<<temp[i];
for(i=0;i<fs;i++)
{
   j=0;
   k=i;
   if (temp[k]>=g[j])
   {
      for(j=0,k=i;j<gs;j++,k++)
      {
         if((temp[k]==1 && g[j]==1) || (temp[k]==0 && g[j]==0))
         {
            temp[k]=0;
         }
         else
         {
            temp[k]=1;
         }
      }
   }
}
int crc[15];
for(i=0,j=fs;i<rs;i++,j++)
   crc[i]=temp[j];
cout<<"\n CRC bits: ";
for(i=0;i<rs;i++)
   cout<<crc[i];
```

```
cout<<"\n Transmitted Frame: ";
int tf[15];
for(i=0;i<fs;i++)
    tf[i]=f[i];
for(i=fs,j=0;i<fs+rs;i++,j++)
    tf[i]=crc[j];
for(i=0;i<fs+rs;i++)
    cout<<tf[i];
cout<<"\n Receiver side : ";
cout<<"\n Received Frame: ";
for(i=0;i<fs+rs;i++)
    cout<<tf[i];
cout << "\nEnter 0 to continue else enter position number where you want to change the frame" << endl ;
int p ;
cin >> p ;
if ( p != 0 ){
    if ( tf[p] == 0 )
        tf[p] = 1 ;
    else
        tf[p] = 0 ;
}
for(i=0;i<fs+rs;i++)
    temp[i]=tf[i];
for(i=0;i<fs+rs;i++)
{
    j=0;
    k=i;
    if (temp[k]>=g[j])
    {
        for(j=0,k=i;j<gs;j++,k++)
```

```
         {
            if((temp[k]==1 && g[j]==1) || (temp[k]==0 && g[j]==0))
            {
               temp[k]=0;
            }
            else
            {
               temp[k]=1;
            }
         }
      }
   }

   cout<<"\n Remainder: ";
   int rrem[15];
   for (i=fs,j=0;i<fs+rs;i++,j++)
      rrem[j]= temp[i];
   for(i=0;i<rs;i++)
      cout<<rrem[i];
   int flag=0;
   for(i=0;i<rs;i++)
   {
      if(rrem[i]!=0)
      {
         flag=1;
      }
   }
   if(flag==0 && p != 0 )
      cout<<"\n Since Remainder Is 0 Hence Message Transmitted From Sender To Receiver Is
Correct";
   else
```

```
        cout<<"\n Since Remainder Is Not 0 Hence Message Transmitted From Sender To Receiver
Contains Error";
return 0;
}
```

****

Output:

```
Enter Size of data: 8
Enter data:1 0 0 0 1 0 1 1


Enter key size: 3


Enter key:1 0 1


Sender Side:
data: 10001011
key :101
Number of 0's to be appended: 2
Message after appending 0's :1000101100
CRC bits: 11
Transmitted Frame: 1000101111
Receiver side :
Received Frame: 1000101111
Enter 0 to continue else enter position number where you want to change the frame
3


 Remainder: 00
 Since Remainder Is 0 Hence Message Transmitted From Sender To Receiver Is Correct
```

****

Hamming Code Program:

```
#include <iostream>
#include<math.h>
using namespace std;

int main() {
        int data[11],rec[11],parity[4];

        cout << "Hamming Code" << endl;
        cout<<"\nSender's Side: \n";
        cout<<"Enter 7 Bit data to send separated by space:";
        for(int i=10;i>=0;i--)
        {
                if(i==0||i==1||i==3||i==7)
                        continue;
                cin>>data[i];
        }
        //Parity Bit Calculation
        data[0]=data[2]^data[4]^data[6]^data[8]^data[10];
        data[1]=data[2]^data[5]^data[6]^data[9]^data[10];
        data[3]=data[4]^data[5]^data[6];
        data[7]=data[8]^data[9]^data[10];

        cout<<"\nInput Data" ;
        for(int i=10;i>=0;i--)
                {
                        if(i==0||i==1||i==3||i==7)
                                continue;
                        cout<<data[i];
                }
        cout<<"\nEncoded Data: ";
```

```
for(int i=10;i>=0;i--)
            cout<<data[i];


cout<<"\n\nReceiver's Side: ";
cout<<"\nEnter 0 to continue without inverting or enter position to invert a bit:";
int k  ;
cin >> k ;
if ( k != 0 ){
   if ( data[k-1] == 1 )
      data[k-1] = 0 ;
   else
      data[k-1] = 1 ;
}
cout << "\nRecieved data is : " ;
for(int i=10;i>=0;i--){
            cout<<data[i];
            rec[i] = data[i] ;
}


parity[0]=rec[0]^rec[2]^rec[4]^rec[6]^rec[8]^rec[10];
parity[1]=rec[1]^rec[2]^rec[5]^rec[6]^rec[9]^rec[10];
parity[2]=rec[3]^rec[4]^rec[5]^rec[6];
parity[3]=rec[7]^rec[8]^rec[9]^rec[10];


if(parity[0]==0&&parity[1]==0&&parity[2]==0&&parity[3]==0)
{
            cout<<"Data is Correct.No error.";
}
else
{
            cout<<"\nError in the code.\nError Position(Binary): ";
```

```
        for(int i=3;i>=0;i--)
            cout<<parity[i];
        int pos=0;
        for(int i=0;i<4;i++)
        {
            pos=pos+(parity[i]*pow(2,i));     //Binary TO Decimal COnversion
        }
        cout<<"\nError Position(Decimal): "<<pos;
    }



    return 0;
}
```

****

Output:

```
Hamming Code


Sender's Side:

Enter 7 Bit data to send separated by space:1 0 0 1 0 1 1


Input Data1001011

Encoded Data: 10011010110


Receiver's Side:

Enter 0 to continue without inverting or enter position to invert a bit:4


Recieved data is : 10011011110

Error in the code.

Error Position(Binary): 0100

Error Position(Decimal): 4
```

****

Dhruvil Shah   047

F18111051   TE [redacted] Comp 1

Assignment 2

Q1 What is CRC ? Explain CRC generator and checker with example.

Ans. A cyclic redundancy check (CRC) is an error detecting code commonly used in digital networks and storage devices to detect accidental changes to raw data. Blocks of data entering these systems get short check value attached, based on the reminder of a polynomial division of their contents.

Eg Frame : 100100   Generator : 1101

Frame after appending 3 0's : 100 100 000

```
            1111 101
      ______________
1101 | 100 100 000
       1101↓
        1000
        1101↓
         1010
         1101↓
          1110
          1101↓
           110↓
           000
           1100
           1101
           0001 → Remainder
```

Transmitted frame : 100100 001

Suppose we change the 5th bit

Received frame : 100101001

```
              111100
1101 ) 1001010001
       1101↓
        1000
        1101↓
         1011
         1101↓
          1100
          1101↓
          00010
           0000
           101  => 5  (ie error bit)
```

Q2 What is hamming code ? Generate hamming code for 7|8 bit data word.

Ans Hamming code is error detecting code capable of detecting upto 2 bit errors and correcting 1 bit error.

eg Frame : 1001101

r = no. of redundant bits

$2^r \geq m + r + 1$

$2^r \geq 7 + r + 1$

$2^r \geq 8 + r$

$r = 4$

Frame : 100 R_4 110 R_3 1 R_2 R_1

$R_1 = 1 \oplus 0 \oplus 1 \oplus 0 \oplus 1 = 1$

$R_2 = 1 \oplus 1 \oplus 1 \oplus 0 \oplus 1 = 0$

$R_3 = 0 \oplus 1 \oplus 1 = 0$
$R_4 = 0 \oplus \oplus 0 \oplus 1 = 1$

$\therefore$ Transmitted frame : 10011100101

Q3 Explain checksum in detail.

Ans In error detection by checksum, data is divided into fixed sized frames or segments.

Sender's End:

Sender adds the segments using 1's compliment to get the sum. The compliment of the sum is the checksum and it is sent along with the data frame.

Receiver's End:

Adds incoming segments along with checksum using 1's compliment arithmetic to get the sum and compliment it.

If result is zero then there is no error.