PROGRAM 2

- ASHISH KUMAR
- 2K18/SE/041

Aim:- Write a C++ program to implement cyclic redundancy check(CRC).

Theory:- CRC or Cyclic Redundancy Check is a method of detecting accidental changes/errors in the communication channel. CRC uses **Generator Polynomial** which is available on both sender and receiver side.

Following are the steps used in CRC for error detection:

- In CRC technique, a string of n 0s is appended to the data unit, and this n number is less than the number of bits in a predetermined number, known as division which is n+1 bits.
- Secondly, the newly extended data is divided by a divisor using a process is known as binary division. The remainder generated from this division is known as CRC remainder.
- Thirdly, the CRC remainder replaces the appended 0s at the end of the original data. This newly generated unit is sent to the receiver.
- The receiver receives the data followed by the CRC remainder. The receiver will treat this whole unit as a single unit, and it is divided by the same divisor that was used to find the CRC remainder.

If the resultant of this division is zero which means that it has no error, and the data is accepted.

If the resultant of this division is not zero which means that the data consists of an error. Therefore, the data is discarded.

CODE:-

```
#include <iostream>

#include <bits/stdc++.h>

using namespace std;

int main()

{
```

```
int i,j,k,l;
int fs;
cout<<"\n Enter Frame size: ";
cin>>fs;

int f[20];
cout<<"\n Enter Frame:";
for(i=0;i<fs;i++)
{
   cin>>f[i];
}

int gs;
cout<<"\n Enter Generator bits size: ";
cin>>gs;

int g[20];
cout<<"\n Enter Generator bits(or key):";
for(i=0;i<gs;i++)
{
   cin>>g[i];
}
```

```
cout<<"\n On Sender Side:";
 //Append 0's
int rs=gs-1;
   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];
}

//Division
for(i=0;i<fs;i++)
{
    j=0;
    k=i;
```

```
    //check whether it is divisible or not
    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];
}
```

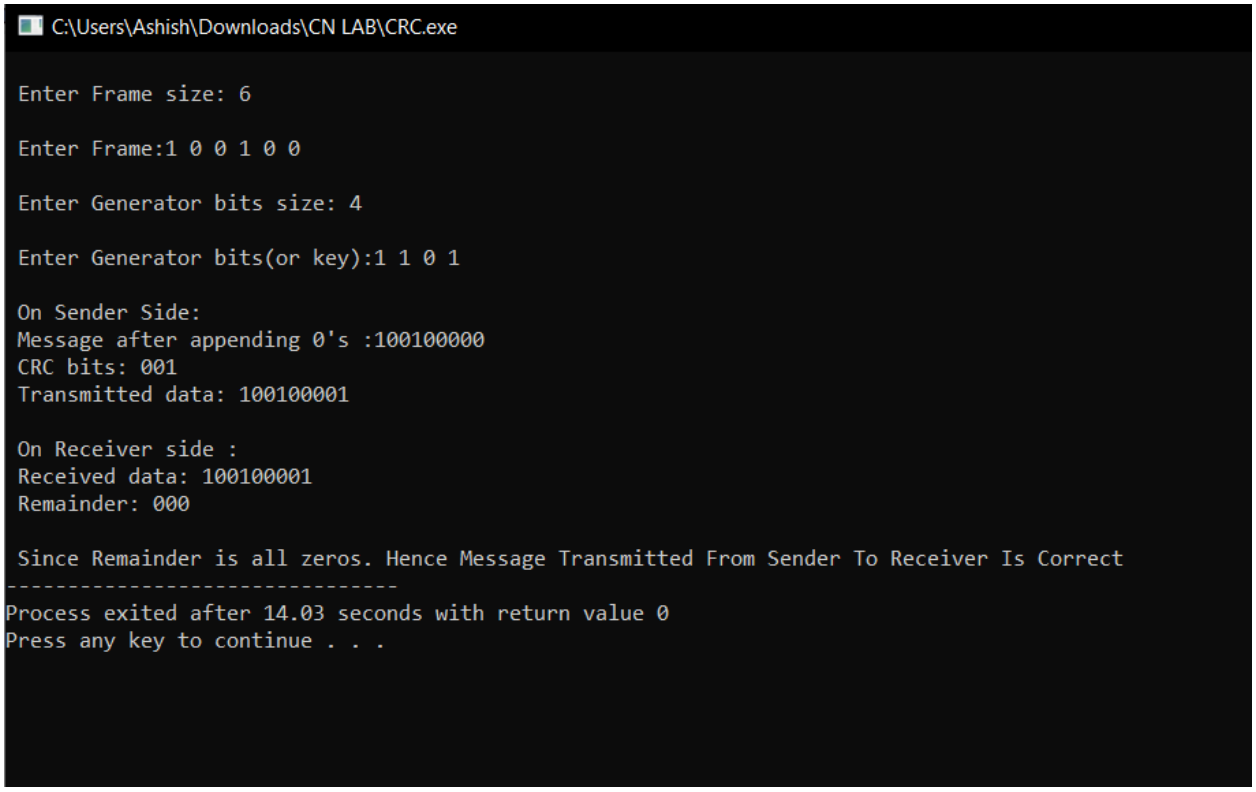
```
cout<<"\n CRC bits: ";
for(i=0;i<rs;i++)
{
   cout<<crc[i];
}

cout<<"\n Transmitted data: ";
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\n On Receiver side : ";
cout<<"\n Received data: ";
for(i=0;i<fs+rs;i++)
{
```

```
    cout<<tf[i];
}

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)
  {
    cout<<"\n\n Since Remainder is all zeros. Hence Message Transmitted From Sender To Receiver Is Correct";
  }
```

```
    else
    {
        cout<<"\n\n Since remainder is not all zeroes. Hence Message Transmitted From Sender To Receiver Contains Error";
    }
    return 0;
}
```

OUTPUT:-



Learning Outcome:- We have successfully implemented cyclic redundancy check(CRC) and we learnt its application in computer network.