

PRACTICAL 5

Aim :

Implement sender Parity, LRC,VRC & CRC programs for input1.txt and input2.txt file

Hardware Requirement :

N/A

Software Requirement :

N/A

Knowledge Requirement :**Theory :****Code for VRC :**

```
#include <iostream>
#include <fstream>
using namespace std;
class A
{
    public:
    int a[100],n=0,k;
    void input()
    {
        cout<<"enter no. of bit of string ";
        cin>>k;
        for (int i=0;i<k;i++)
        {
            cin>>a[i];
        }
    }
};
```

```
        if ( a [ i ] == 1 )
        {
            n++;
        }
    }
    cout << endl;
    if ( n % 2 == 0 )
    {
        for ( int i = 0; i < k; i++ )
        {
            cout << a [ i ];
        }
        cout << " | 0 ";
    }
    else {
        for ( int i = 0; i < k; i++ )
        {
            cout << a [ i ];
        }
        cout << " | 1 ";
    }
}

};

void cmd()
{
    A a;
    a.input();
}

int main()
{
    int a;
    cout << "enter 1 if you want to open cmd and enter 2
    if you want to open in file ";
```

```
cin>>a;
switch (a)
{
    case 1:
    {
        cmd();
        break;
    }
case 2:
{
    ofstream files;
    files.open("vrcfile.txt");
    int a[100],n=0,k;
    cout<< "enter no. of bit of string \n";
    cin>>k;
    for(int i=0;i<k;i++)
    {
        cin>>a[i];
        if(a[i]==1)
        {
            n++;
        }
    }
    for(int i=0;i<k;i++)
    {
        files <<a[i];
    }
    cout<<endl;
    files << "\n";
    if(n%2==0)
    {
        for(int i=0;i<k;i++)
        {
            files << a[i];
```

```

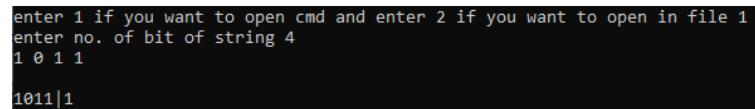
    }
    files <<"|0";
    }
    else
    {
        for (int i=0;i<k;i++)
        {
            files <<a[i];
        }
        files <<"|1";
    }

    files.close();
    std::cin.get();
    break;
}
}
}

```

output

In cmd



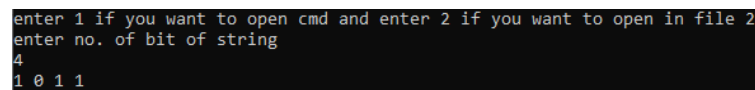
```

enter 1 if you want to open cmd and enter 2 if you want to open in file 1
enter no. of bit of string 4
1 0 1 1
1011|1

```

Figure 1: Output of VRC in cmd

In file



```

enter 1 if you want to open cmd and enter 2 if you want to open in file 2
enter no. of bit of string
4
1 0 1 1

```

Figure 2: Input in cmd of VRC in file

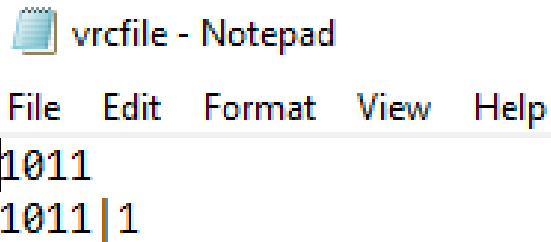


Figure 3: Output of VRC in file

Code for LRC:

```
#include<iostream>
#include<fstream>
using namespace std;
class vrc
{
    public:
    int a[16],n=0,p[4];
    void inputs()
    {
        cout<<"enter the bit string"<<endl;
        for(int i=0;i<16;i++)
        {
            cin>>a[i];
        }
        for(int j=0;j<4;j++)
        {
            for(int i=j;i<16;i+=4)
            {
                if(a[i]==1)
                {
                    n++;
                }
            }
        }
    }
}
```

```
    }
    if (n%2==0)
    {
        p[j]=0;
        n=0;
    }
    else
    {
        p[j]=1;
        n=0;
    }
}
for (int i=0;i<16;i++)
{
    cout<<a[i];
}
cout<<" ";
for (int i=0;i<4;i++)
{
    cout<<p[i];
}
}
};
void cmd()
{
    vrc v;
    v.inputs();
}
int main()
{
    int a;
    cout<<"enter 1 if you want to open cmd and enter 2
    if you want to open in file ";
    cin>>a;
```

```
switch (a)
{
    case 1:
    {
        cmd();
        break;
    }
case 2:
{
    ofstream files;
    files.open("lrcfile.txt");
    int a[16],n=0,p[4];
    cout<<"enter the bit string"<<endl;
    for(int i=0;i<16;i++)
    {
        cin>>a[i];
    }
    for(int j=0;j<4;j++)
    {
        for(int i=j;i<16;i+=4)
        {
            if(a[i]==1)
            {
                n++;
            }
        }
        if(n%2==0)
        {
            p[j]=0;
            n=0;
        }
        else
        {
            p[j]=1;
        }
    }
}
```

```

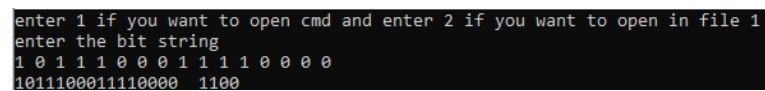
        n=0;
    }
}
for (int i=0;i <16;i++)
{
    files <<a[ i ];
}
files <<" ";
for (int i=0;i <4;i++)
{
    files <<p[ i ];
}
files.close();
std::cin.get();
break;
}

}
}

```

output

In cmd



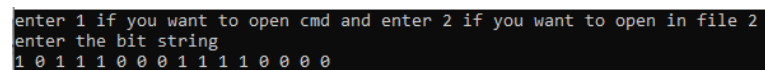
```

enter 1 if you want to open cmd and enter 2 if you want to open in file 1
enter the bit string
1 0 1 1 1 0 0 0 1 1 1 1 0 0 0 0
1011100011110000 1100

```

Figure 4: Output of LRC in cmd

In file



```

enter 1 if you want to open cmd and enter 2 if you want to open in file 2
enter the bit string
1 0 1 1 1 0 0 0 1 1 1 1 0 0 0 0

```

Figure 5: Input in cmd of LRC in file

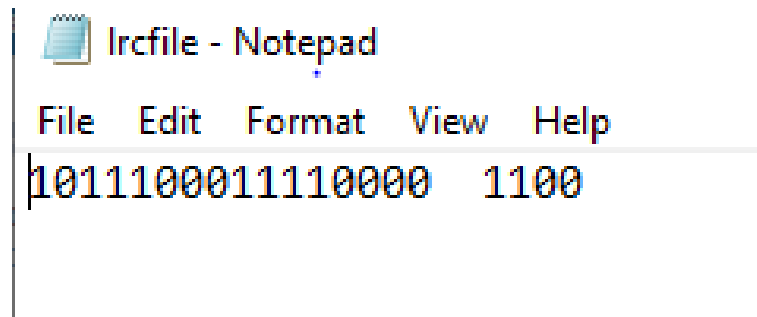


Figure 6: Output of LRC in file

Code for CRC:

```
#include<iostream>
#include<fstream>
using namespace std;
class c
{
public:
    int i,j,n,g,k,a[20],b[20],q[20],s;
    void input()
    {
        cout<<"Transmitter side:";
        cout<<"\nEnter no. of data bits:";
        cin>>n;
        cout<<"Enter data:";
        for(i=0;i<n;i++)
            cin>>b[i];

        cout<<"Enter size of divisor:";
        cin>>g;
        do{
            cout<<"Enter divisor:";
            for(j=0;j<g;j++)
                cin>>a[j];

        }
    }
};
```

```
while (a[0]!=1);
cout<<"\n\tThe divisor matrix:";
for (j=0;j< g;j++)
cout<<a[j];

k=n+(g-1);
cout<<"\n\tThe appended matrix is:";
for (i=0;i< j;++i)
b[n+i]=0;

for (i=0;i< k;++i)

cout<<b[i];

for (i=0;i< n;++i)
q[i]= b[i];

for (i=0;i< n;++i)
{
if (b[i]==0)
{
for (j=i;j< g+i;++j)
b[j] = b[j]^0;
}
else
{
b[i] = b[i]^a[0];
b[i+1]=b[i+1]^a[1];
b[i+2]=b[i+2]^a[2];
b[i+3]=b[i+3]^a[3];
}
}
cout<<"\n\tThe CRC is  ";
for (i=n;i < k;++i)
```

```
cout<<b[ i ];
s=n+k;
for ( i=n; i< s; i++)
q[ i]=b[ i ];
cout<<"\n";
for ( i=0; i< k; i++)
cout<<q[ i ];
    }

};
void cmd()
{
    c c;
    c.input ();
}
int main()
{
    int a;
    cout<<"enter 1 if you want to open cmd and enter 2
    if you want to open in file ";
    cin>>a;
    switch (a)
    {
        case 1:
        {
            cmd();
            break;
        }
    case 2:
    {
        ofstream files;
        files.open("crcfile.txt");
        int i,j,n,g,k,a[20],b[20],q[20],s;
        files<<"Transmitter side:";
```

```
cout<<"\nEnter no. of data bits:";
cin>>n;
cout<<"Enter data:";
for (i=0;i< n;i++)
cin>>b[i];

cout<<"Enter size of divisor:";
cin>>g;
do{
cout<<"Enter divisor:";
for (j=0;j< g;j++)
cin>>a[j];

}
while (a[0]!=1);
files<<"\n\tThe divisor matrix:";
for (j=0;j< g;j++)
files<<a[j];

k=n+(g-1);
files<<"\n\tThe appended matrix is:";
for (i=0;i< j;++i)
b[n+i]=0;

for (i=0;i< k;++i)

files<<b[i];

for (i=0;i< n;++i)
q[i]= b[i];

for (i=0;i< n;++i)
{
if (b[i]==0)
```

```
{
for (j=i ; j< g+i;++j)
b[j] = b[j]^0;
}
else
{
b[i] = b[i]^a[0];
b[i+1]=b[i+1]^a[1];
b[i+2]=b[i+2]^a[2];
b[i+3]=b[i+3]^a[3];
}
}
files <<"\n\tThe CRC is :";
for (i=n; i < k;++i)
files <<b[i];
s=n+k;
for (i=n; i< s; i++)
q[i]=b[i];
files <<"\n";
for (i=0; i< k; i++)
files <<q[i];
        files.close();
    std::cin.get();
    break;
}
}
}
```

output

In cmd

```
enter 1 if you want to open cmd and enter 2 if you want to open in file 1
Transmitter side:
Enter no. of data bits:4
Enter data:1 0 0 1
Enter size of divisor:4
Enter divisor:1 0 1 1

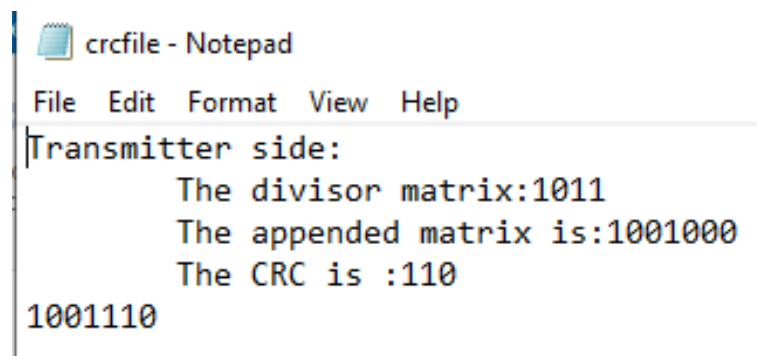
    The divisor matrix:1011
    The appended matrix is:1001000
    The CRC is :110
1001110
```

Figure 7: Output of CRC in cmd

In file

```
enter 1 if you want to open cmd and enter 2 if you want to open in file 2
Enter no. of data bits:4
Enter data:1 0 0 1
Enter size of divisor:4
Enter divisor:1 0 1 1
```

Figure 8: Input in cmd of CRC in file



```
Transmitter side:
    The divisor matrix:1011
    The appended matrix is:1001000
    The CRC is :110
1001110
```

Figure 9: Output of CRC in file

Conclusion :

Hence we have studied that how do the VRC,LRC and CRC works.

References :

- <https://stackoverflow.com/questions/3175105/inserting-code-in-this-latex-document-with-indentation>

- ecomputernotes.com/computernetworkingnotes/communication-networks/cyclic-redundancy-check

Question Answer :

1) Why Error correction is required ?

Data can be corrupted during transmission. For reliable communication error must be detected and corrected. Error detection and correction or error control are techniques that enable reliable delivery of digital data over unreliable communication channels.

Error transmission and detection can be implemented on either on data link layer or transport layer in OSI model.

2) What is parity bit?

A parity bit is a bit that is added to a group of source bits to ensure that the number of set bits (i.e., bits with value 1) in the outcome is even or odd. It is a very simple scheme that can be used to detect single or any other odd number (i.e., three, five, etc.) of errors in the output. An even number of flipped bits will make the parity bit appear correct even though the data is erroneous.