Experiment No 7

B.1: Code of performed experiment Program:

```python
#Hamming Code
def parity(data, p_bit):
    i,j=0,0
    c=0
    while(i<len(data)):
        j=j+1
        if data[i]==1:
            c=c+1
        if j==p_bit:
            i=i+p_bit+1
            j=0
        else:
            i=i+1

    if c%2==0:
        return 0
    else:
        return 1

code=list(int(i) for i in input("Enter 4 bit data to be sent:").split())
code.insert(6,0)
code.insert(5,0)
code.insert(3,0)
p1=parity(code,1)
p2=parity(code,2)
p4=parity(code,4)
print("Parities are:",p1,p2,p4)
code[6]=p1
code[5]=p2
code[3]=p4
print("Hamming code sent is:",code)
```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

tjrox0825@Taruns-MacBook-Pro Tj CN Exps %  /usr/bin/env /Library/Frameworks/Python.fra
353 -- "/Users/tjrox0825/Desktop/Sem 5/Computer Networks/Tj CN Exps/exp7_N049.py"
Enter 4 bit data to be sent:1 0 1 1
Parities are: 1 0 0
Hamming code sent is: [1, 0, 1, 0, 1, 0, 1]
tjrox0825@Taruns-MacBook-Pro Tj CN Exps % █
```

B.2: Observations and Learning's:

The data link layer has the responsibility of error control, which is error detection and correction.
Error detecting has many methods like CRC, Simple parity, 2D parity, etc. For error correction,
we use Hamming code. In Hamming code, we generate parities based on the data bits and put
them in positions that are powers of 2. To get the parity bits, we see the bits related to it, eg: for
parity bit 2, the bits related to it are D2, D3, D6, D7 and so on. Basically, bits in groups of 2 at
gap of 2 bits are considered. This gives us the final hamming code to be sent to receiver.
Hamming code for 4-bit data is: $D_7 D_6 D_5 P_4 D_3 P_2 P_1$. It also gives us the exact position of error,
in case of single bit errors.

B.3: Conclusion:

I have learnt the concept of error correction and its method, Hamming code. I have also
successfully implemented Hamming code as a program.

B.4: Questions of Curiosity:

If there is 1-Bit error. How can it be corrected on the receiver end using hamming code?

```python
#QOC
#For 7 bit Hamming code with 4 bit data
def parity(data, p_bit):
    i,j=0,0
    c=0
    end=len(data)-(p_bit)-1
    while(i<=end):
        j=j+1
        if data[i]==1:
            c=c+1
        if j==p_bit:
            i=i+p_bit+1
            j=0
        else:
            i=i+1

    if c%2==0:
        return 0
    else:
        return 1

code=list(int(i) for i in input("Enter Hamming code received:").split())
p1=code[6]
p2=code[5]
p4=code[3]
print("Parity bits in message received:",p1,p2,p4)
calc_p1=parity(code,1)
calc_p2=parity(code,2)
calc_p4=parity(code,4)
print("Calculated parity bits are:",calc_p1,calc_p2,calc_p4)
if calc_p1==p1 and calc_p2==p2 and calc_p4==p4:
    print("No error")
else:
    e=0
    if calc_p4!=p4:
        e=e+4
    if calc_p2!=p2:
        e=e+2
    if calc_p1!=p1:
        e=e+1
    print("Error is at position:",e)
```

<u>Answer</u>
<u>Code</u>

<u>Output:</u>

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL
tjrox0825@Taruns-MacBook-Pro Tj CN Exps %  /usr/bin/env /Library/Frameworks/Python.f
421 -- "/Users/tjrox0825/Desktop/Sem 5/Computer Networks/Tj CN Exps/exp7.2_N049.py"
Enter Hamming code received:1 0 1 0 1 0 1
Parity bits in message received: 1 0 0
Calculated parity bits are: 1 0 0
No error
tjrox0825@Taruns-MacBook-Pro Tj CN Exps %
```

```
 /usr/bin/env /Library/Frameworks/Python.framework/Versions/3.7/bin/python3 /Users/t
tjrox0825@Taruns-MacBook-Pro Tj CN Exps %  /usr/bin/env /Library/Frameworks/Python.f
474 -- "/Users/tjrox0825/Desktop/Sem 5/Computer Networks/Tj CN Exps/exp7.2_N049.py"
Enter Hamming code received:1 1 1 0 1 0 1
Parity bits in message received: 1 0 0
Calculated parity bits are: 1 1 1
Error is at position: 6
tjrox0825@Taruns-MacBook-Pro Tj CN Exps %
```