Ex. No: 6
21/8/24

Humming Code

Page No :
Date :

Aim: Write a program to implement error detection and correction using Hamming code concept. Make a test run to input data stream and verify error correction feature.

Error Correction at Data Link Layer:

Hamming code is a set of error-correction codes that can be used to detect and correct the errors that can occur when the data is transmitte from the sender to the receiver. It is a technique developed by R.W Hamming for error correction.

Create sender program with below feat

i) Input to Sender file should be a text of any length. Program should convert the text to binary.

ii) Apply hamming code concept on the binary data & add redundant bit.

iii) Save this output in a file called Channel.

Create a receiver program with feature

i) Receiver program should read the input from channel file.

ii) Apply hamming code on binary data check for error.

iii) If there is an error, display the position of the error.

iv) Else remove the redundant bit and convert the binary data to ascii and display the output.

Page No :
Date :

Code:

```
def calcRedundantBits(m):
    for i in range(m):
        if (2**i >= m+i+1):
            return i

def PosRedundantBits(data, r):
    j = 0
    k = 1
    m = len(data)
    res = ''
    for i in range(1, m+r+1):
        if(i == 2**j):
            res = res + '0'
            j += 1
        else:
            res = res + data[-1*k]
            k += 1
    return res[::-1]

def calcParityBits(arr, r):
    n = len(arr)
    for i in range(r):
        val = 0
        for j in range(1, n+1):
            if(j & (2**i) == (2**i)):
                val = val ^ int(arr[-1*j])
```

```
    arr = arr[:n-(2**i)] + str(val) + arr[n-(2**i)+1:]
return arr

def detectError(arr, nr):
    n = len(arr)
    res = 0
    for i in range(0,r):
        val = 0
        for j in range(1, n+1):
            if(j & (2**i) == (2**j)):
                val = val ^ int(arr[-1*j])
        res = res + val*(10**i)
    return int(str(res),2)
data = '1011001'
m = len(data)
r = calcRedundantBits(m)
arr = PosRedundantBits(data, r)
arr = calcParityBits(arr, r)
print("Data transferred is", arr)
arr = '11101001110'
print("Error Data is" + arr)
correction = detectError(arr, r)
if(correction == 0):
    print("There is no error in the
          received message")
else:
    print("The position of error is", len
          (arr) - correction - 1, 'from
                                  the left')
```

Page No :
Date :

Output:

Data transferred is 10101001110
Error Data is 11101001110
The position of error is 2 from the left.

Result:

Thus, the output is verified successfully.

2/9/24