**RAJALAKSHMI ENGINEERING COLLEGE**

**RAJALAKSHMI NAGAR, THANDALAM – 602 105**

****

|  |
| --- |
| **CS19541**  **COMPUTER NETWORKS LABORATORY** |
| **Laboratory Manual Note Book** |

|  |
| --- |
| **Name : Amritha.A**  **Year/Branch/Section : 3rd year - Computer Science and Design**  **Register No. : 221701007**  **Semester : 5**  **Academic Year : 2024-2025** |

**EXP NO : 4 HAMMING CODE**

**DATE : 07/08/24**

**AIM:**

To write a Python program to implement error detection and correction using Hamming Code.

**SOFTWARE USED:**

Jupyter Notebook

**PROCEDURE:**

Sender Side:

1. Input to sender file should be a text of any length. Program should convert the text to binary.

2. Apply hamming code concept on the binary data and add redundant bits to it.

3. Save this output in a file called channel.

Receiver Side:

1. Receiver program should read the input from Channel file.

2. Apply hamming code on the binary data to check for errors.

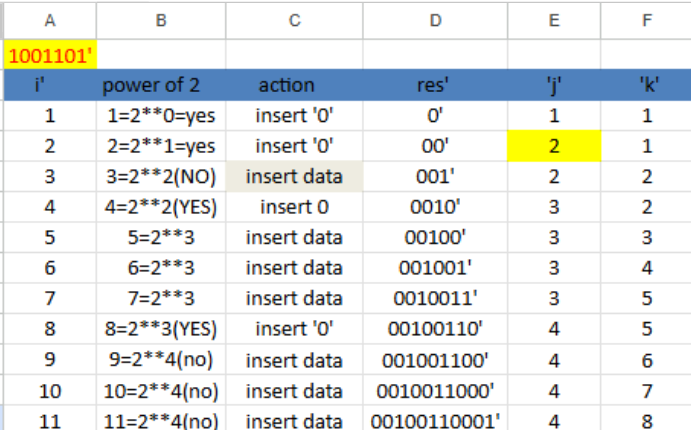
3. If there is an error, display the position of the error.

4. Else remove the redundant bits and convert the binary data to ascii and display the output.

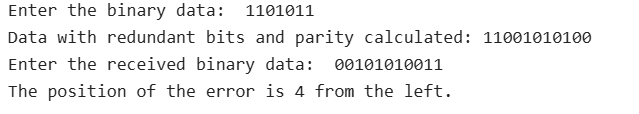
**Program:**

def calcRedundantBits(m):  
    for i in range(m):  
        if 2\*\*i >= m + i + 1:  
            return i  
  
def posRedundantBits(data, r):  
    j = 0  
    k = 0  
    m = len(data)  
    res = ''  
    for i in range(1, m + r + 1):  
        if i == 2\*\*j:  
            res += '0'  
            j += 1  
        else:  
            res += data[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 ^= 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(nr):  
        val = 0  
        for j in range(1, n + 1):  
            if j & (2\*\*i) == (2\*\*i):  
                val ^= int(arr[-1 \* j])  
        res += val \* (10\*\*i)  
    return int(str(res), 2)  
  
data = input("Enter the binary data: ")  
m = len(data)  
  
r = calcRedundantBits(m)  
  
arr = posRedundantBits(data, r)  
  
arr = calcParityBits(arr, r)  
print("Data with redundant bits and parity calculated: " + arr)  
  
received\_data = input("Enter the received binary data: ")  
  
correction = detectError(received\_data, r)  
  
if correction == 0:  
    print("There is no error in the received message.")  
else:  
    print("The position of the error is", len(received\_data) - correction + 1, "from the left.")

**Tabulation:**



**Output:**



**RESULT**:

Thus, the Python program to implement error detection and correction using Hamming Code has been executed successfully.