7/8/25

Experiment 5

AIM : Write a program to implement error detection and correction using Hamming code concept.

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 transmitted from the sender to the receiver.

Sender Program :

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

```
def hamming_code(data):
    def insert_bits(data):
        m = len(data)
        r = 0
        while (2**r) < (m + r + 1):
            r += 1
        n = m + r
        result = ['0'] * n
        j = 0
        for i in range(1, n+1):
            if i as (i-1) == 0:
                continue
            result[-i] = data[-(j+1)]
            j += 1
            if j == m:
                break
        return result, n, r
```

```
def calc_parity (pdata, r):
    n = len(pdata)
    result = pdata [:]
    for i in range (r):
        parity_pos = (2** i)
        parity_val = 0
        for k in range (1, n+1):
            if k in parity_pos:
                parity_val ^= int(result[-
        result [-parity_pos] = str (parity_val
    return result
pbits, n, r = insert_bits (data)
code = calc_parity (pbits, r)
return ''.join(code)

input = input ('Enter binary data :')
print ('Hamming code =', hamming_code (input
```

Receiver Program

- Apply hamming code on the binary data to check for errors.
- If there is any error, display the position of the error.

```
def hamming_check (hamming code):
    n = len (hamming code)
    r = 0
    while (2** r) < n+1 :
        r += 1
    syn = 0
    parity = []
```

```
    for i in range (r):
        parity_pos = 2**i
        parity_val = 0
        for k in range (1, n+1):
            if k as parity_pos :
                parity_val ^ = int (hammingcode
                                    [-k])
        parity.append (parity_val)
        syn |= (parity_val << i)
    synbits = ''.join (str (x) for x in
                       reversed (parity))
    return synbits, syn
code = input ("Enter received hamming
                code.")
res, error = hamming_check (code)
print ('Error bits :', res)
if error == 0 :
    print ('No error detected')
else :
    print ('Error detected at bit
            position :', error)
```

OUTPUT :

```
Enter binary data : 1001101
Hamming code = 10011100101

Enter received Hamming code : 10010100101
Error syndrome bits : 0111
Error detected at bit position : 7
```

Enter received Hamming code : 100 111 00101

Error syndrome bits : 0000

No error detected

Result : Sender and receiver program for hamming code concept was executed and get the output.

3/9/24