Ex: 6.
21/8/24.

PRACTICAL - 6.

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.

PROGRAM:

```
def text_to_binary(text):
    return ''.join(format(ord(char), '08b') for char in text)

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(nr):
        val = 0
        for j in range(1, n+1):
            if j & (2**i) == (2**i):
                val = val ^ int(arr[-1 * j])
        res = res + val * (10**i).
    return int(str(res), 2)
```

```
def correct_error(received_data, error_position):
    if error_position == 0:
        return received_data
    data_list = list(received_data)
    error_index = len(received_data) - error_position
    corrected_bit = '0' if data_list[error_index] == '1' else '1'
    data_list[error_index] = corrected_bit
    return ''.join(data_list)

def main():
    word = input("Enter a word to be transmitted: ")
    binary_data = text_to_binary(word)
    print(f"Binary representation of '{word}' is {binary_data}")
    m = len(binary_data)
    r = calcRedundantBits(m)
    arr = posRedundantBits(binary_data, r)
    arr = calcParityBits(arr, r)
    print("Data transferred is " + arr)
    received_data = input("Enter the received data (with possible errors): ")
```

```
if not all (bit in '01' for bit in received_
                                    data):
    print ("Invalid input ! Received data
        should be a binary string.")
    return
correction = detectError(received_data, r).
if correction == 0:
    print("There is no error in the received")
else:
    corrected_data = correctError(received_data,
                                  correction)
    print("The position of error is", len(received_
        data) - correction + 1, "from the left")
    print("corrected data is " + corrected_data)

if __name__ == "__main__":
    main()
```

OUTPUT :-

Enter a word to be transmitted : heli

Binary representation of 'heli' is 01101000 01100101 01101100 01101001.

Data transferred is 01101101000 11001010111011110001100100 1110.

Enter received data (with possible errors):

01101010001110010101101111000110010011110.

The position of error is 10 from the left.

corrected data is 01101010001100101011011100011001001110.

Result :-

The hamming code is successfully executed & output is verified.

13/9/22