Exp 7)Implementation of Hamming Code.

```
def calculate_parity_bits(data_bits, r):
    n = len(data_bits) + r
    parity_positions = [2**i for i in range(r)]
    hamming_code = [0] * (n + 1)
    j = 0
    for i in range(1, n + 1):
        if i in parity_positions:
            continue
        hamming_code[i] = int(data_bits[j])
        j += 1
    for p in parity_positions:
        parity = 0
        for i in range(1, n + 1):
            if i & p:
                parity ^= hamming_code[i]
        hamming_code[p] = parity
    return hamming_code[1:]
def detect_and_correct_error(hamming_code):
    n = len(hamming_code)
    r = 0
    while (2**r) < (n + 1):
        r += 1
    error_position = 0
    for p in [2**i for i in range(r)]:
        parity = 0
        for i in range(1, n + 1):
            if i & p:
                parity ^= hamming_code[i-1]
        if parity:
            error_position += p
    if error_position:
        print(f"Error detected at position: {error_position}")
        hamming_code[error_position-1] ^= 1  # Correct error
        print("Corrected code:", hamming_code)
    else:
        print("No errors detected.")
    return hamming_code
def main():
    data_bits = input("Enter the data bits: ")
    r = 0
    while (2**r) < (len(data_bits) + r + 1):
        r += 1
```

```
    hamming_code = calculate_parity_bits(data_bits, r)
    print("Hamming code with parity bits:", hamming_code)
    error_hamming_code = hamming_code[:]
    error_position = int(input("Enter error position (1-based index,
        0 for no error): "))
    if error_position:
        error_hamming_code[error_position-1] ^= 1
    print("Received code:", error_hamming_code)
    detect_and_correct_error(error_hamming_code)
if __name__ == "__main__":
    main()
```

Output:

```
Enter the data bits: 11001101
Hamming code with parity bits: [1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1]
Enter error position (1-based index, 0 for no error): 6
Received code: [1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1]
Error detected at position: 6
Corrected code: [1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1]
```