CN LAB 3

Name: Rachit Nimje
Class: TY-CS-D
Roll no: 12

Batch: 1
PRN: 12210952

Title: Write a program in Java for error detection and correction for 7/8 bits ASCII codes using Hamming Code.

Code:

```
import java.util.Scanner;

public class HammingCode {
   private static final Scanner sc = new Scanner(System.in);

   public static void main(String[] args) {
       String input = getInput();
       int[] data = convertToHammingCode(input);

       encodeHamming(data);
       System.out.println("Encoded Hamming Code: " + arrayToString(data));

       int errorPosition = introduceError(data);

       int detectedErrorPosition = detectAndCorrectError(data);
       displayErrorInfo(detectedErrorPosition, errorPosition, data);

       String decodedAscii = decodeHamming(data);
       System.out.println("Decoded ASCII: " + decodedAscii);
       System.out.println("Original ASCII: " + input);

       sc.close();
   }

   private static String getInput() {
       System.out.print("Enter a 7-bit ASCII code: ");
       String input = sc.nextLine();

       if (input.length() != 7 || !input.matches("[01]+")) {
           System.out.println("Invalid input. Please enter a 7-bit binary
string.");
           System.exit(1);
       }

       return input;
   }
```

```
    private static int[] convertToHammingCode(String input) {
        int[] data = new int[12];
        int j = 0;
        for (int i = 0; i < 12; i++) {
            if (i == 0 || i == 1 || i == 3 || i == 7) {
                data[i] = 0;
            } else {
                if (j < input.length()) {
                    data[i] = input.charAt(j) - '0';
                    j++;
                }
            }
        }
        return data;
    }

    private static void encodeHamming(int[] data) {
        data[0] = data[2] ^ data[4] ^ data[6] ^ data[8] ^ data[10];
        data[1] = data[2] ^ data[5] ^ data[6] ^ data[9] ^ data[10];
        data[3] = data[4] ^ data[5] ^ data[6] ^ data[11];
        data[7] = data[8] ^ data[9] ^ data[10] ^ data[11];
    }

    private static int introduceError(int[] data) {
        System.out.print("Enter position to introduce error (1-12), or 0 for no
error: ");
        int errorPosition = sc.nextInt();

        if (errorPosition > 0 && errorPosition <= 12) {
            data[errorPosition - 1] ^= 1;
            System.out.println("Hamming Code with error: " +
arrayToString(data));
        }
        return errorPosition;
    }

    private static int detectAndCorrectError(int[] data) {
        int c1 = data[0] ^ data[2] ^ data[4] ^ data[6] ^ data[8] ^ data[10];
        int c2 = data[1] ^ data[2] ^ data[5] ^ data[6] ^ data[9] ^ data[10];
        int c3 = data[3] ^ data[4] ^ data[5] ^ data[6] ^ data[11];
        int c4 = data[7] ^ data[8] ^ data[9] ^ data[10] ^ data[11];

        int errorBit = c1 + (c2 << 1) + (c3 << 2) + (c4 << 3);

        if (errorBit > 0) {
            data[errorBit - 1] ^= 1;
        }

        return errorBit;
    }
```

```
    private static void displayErrorInfo(int detectedErrorPosition, int
actualErrorPosition, int[] data) {
        if (detectedErrorPosition == 0) {
            System.out.println("No error detected.");
        } else {
            System.out.println("Error detected at position: " +
detectedErrorPosition);
            System.out.println("Actual error position: " +
actualErrorPosition);
            System.out.println("Corrected Hamming Code: " +
arrayToString(data));
        }
    }

    private static String decodeHamming(int[] data) {
        StringBuilder decoded = new StringBuilder();
        for (int i = 0; i < 12; i++) {
            if (i != 0 && i != 1 && i != 3 && i != 7) {
                decoded.append(data[i]);
            }
        }
        return decoded.toString();
    }

    private static String arrayToString(int[] arr) {
        StringBuilder sb = new StringBuilder();
        for (int bit : arr) {
            sb.append(bit);
        }
        return sb.toString();
    }
}
```

Output:

```
/home/halogen/.jdks/openjdk-22.0.2/bin/java -javaagent:/var/lib/s
Enter a 7-bit ASCII code: 1100001
Encoded Hamming Code: 101110010010
Enter position to introduce error (1-12), or 0 for no error: 11
Hamming Code with error: 101110010000
Error detected at position: 11
Actual error position: 11
Corrected Hamming Code: 101110010010
Decoded ASCII: 11000010
Original ASCII: 1100001

Process finished with exit code 0
```