МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”

Кафедра ИИТ

ОТЧЁТ
По лабораторной работе №1
«Избыточное кодирование данных в информационных системах. Код Хемминга»

Выполнил:
Студент группы ИИ-22
Полиенко В.Э.
Проверил:
Хацкевич А.С.

2024

Цель работы: приобретение практических навыков кодирования/декодирования двоичных данных при использовании кода Хемминга.

Задание.

1. Закрепить теоретические знания по использованию методов помехоустойчивого кодирования для повышения надежности передачи и хранения в памяти компьютера двоичных данных.
2. Разработать приложение для кодирования/декодирования двоичной информации кодом Хемминга с минимальным кодовым расстоянием 3 или 4.
3. Результаты выполнения лабораторной работы оформить в виде отчета с листингом разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.
4. Ответить на контрольные вопросы

Ход работы

Вариант	M[1]	r
15	636	3

1. Составить код Хемминга (классический алгоритм) (M+r, M), допустить ошибку в одном из разрядов и отыскать её по алгоритму.

2. Составить код Хемминга (расширенный алгоритм) (первые 7 битов M + 3 проверочных, первые 7 битов M), допустить 2 или более ошибок в разрядах и отыскать их по алгоритму

Код программы:

```
import java.util.Arrays;

public class HammingCode {
    private static int r = 0;

    public static int[] encodeHammingCode(int number) {
        String binaryNumber = Integer.toBinaryString(number);
        int dataLength = binaryNumber.length();

        r = 0;
        while (Math.pow(2, r) < (dataLength + r + 1)) {
            r++;
        }
        int codeLength = dataLength + r;

        int XOR = 0;
        int[] code = new int[codeLength];

        for (int i = 0, j = 0; i < codeLength; i++) {
            if (Math.log(i + 1) / Math.log(2) == Math.floor(Math.log(i + 1) /
Math.log(2))) {
                code[i] = 0;
            } else {
                code[i] = Character.getNumericValue(binaryNumber.charAt(j++));
                if (code[i] == 1) {
                    XOR ^= i + 1;
                }
            }
        }

        String str = Integer.toBinaryString(XOR);
        while (str.length() < r) {
            str = "0" + str;
        }

        for (int i = 0; i < r; i++) {
            int position = (int) Math.pow(2, i);
            code[position - 1] = Integer.parseInt(String.valueOf(str.charAt(i)));
        }
```

```
        System.out.print("\nGenerated Hamming Code:\n");
        Arrays.stream(code).forEach(System.out::print);
        System.out.println();
        return code;
    }

    public static void checkHammingCode(int[] code) {
        int XOR = 0;

        for (int i = 0; i < code.length; i++) {
            if (code[i] == 1) {
                XOR ^= i + 1;
            }
        }

        if (XOR != 0) {
            System.out.println("Error detected at position: " + XOR);
            code[XOR - 1] ^= 1;
            System.out.print("Corrected Hamming Code: ");
            Arrays.stream(code).forEach(System.out::print);
            System.out.println("\n");
        } else {
            System.out.println("No errors detected.");
        }
    }

    public static int decodeHammingCode(int[] code) {
        StringBuilder dataBits = new StringBuilder();
        for (int i = 0; i < code.length; i++) {
            if (Math.log(i + 1) / Math.log(2) != Math.floor(Math.log(i + 1) /
Math.log(2))) {
                dataBits.append(code[i]);
            }
        }
        return Integer.parseInt(dataBits.toString(), 2);
    }


}

public class Main{
    public static void main(String[] args){
        int input = 636;
        System.out.println("Entered number: " + input);

        int[] code = HammingCode.encodeHammingCode(input);
        for (int i = 0; i < code.length; i++){
            code[i] ^= 1;
            System.out.printf("Incorrected Hamming Code: ");
            Arrays.stream(code).forEach(System.out::print);
            System.out.printf("\n");
            HammingCode.checkHammingCode(code);
        }

        int decoded = HammingCode.decodeHammingCode(code);
        System.out.println("Decoded number: " + decoded);

        HammingCode.checkHammingCode(code);
    }
}
```

Результат работы:

```
Entered number: 636

Generated Hamming Code:
00100010111100
Incorrected Hamming Code: 10100010111100
Error detected at position: 1
Corrected Hamming Code: 00100010111100

Incorrected Hamming Code: 01100010111100
Error detected at position: 2
Corrected Hamming Code: 00100010111100

Incorrected Hamming Code: 00000010111100
Error detected at position: 3
Corrected Hamming Code: 00100010111100
```

Вывод: приобрёл практические навыки кодирования/декодирования двоичных данных при использовании кода Хемминга.