

## Softcopy – Practical-2 (Huffman)

### # Code

```
import java.util.*;

public class Code_A2 {
    private static HuffmanNode root;
    private static Map<Character, String> huffmanCodes = new HashMap<>();
    private static Map<Character, Integer> freqMap = new HashMap<>();
    private static String inputText = "";

    static class HuffmanNode implements Comparable<HuffmanNode> {
        char ch;
        int freq;
        HuffmanNode left, right;

        HuffmanNode(char ch, int freq) {
            this.ch = ch;
            this.freq = freq;
        }

        public int compareTo(HuffmanNode other) {
            return this.freq - other.freq;
        }
    }

    private static void generateCodes(HuffmanNode node, String code) {
        if (node == null) return;
        if (node.left == null && node.right == null)
            huffmanCodes.put(node.ch, code);
        generateCodes(node.left, code + "0");
        generateCodes(node.right, code + "1");
    }

    private static void buildHuffmanTree() {
        PriorityQueue<HuffmanNode> pq = new PriorityQueue<>();
        for (Map.Entry<Character, Integer> e : freqMap.entrySet())
            pq.add(new HuffmanNode(e.getKey(), e.getValue()));

        while (pq.size() > 1) {
            HuffmanNode left = pq.poll();
            HuffmanNode right = pq.poll();
            HuffmanNode parent = new HuffmanNode('-', left.freq +
right.freq);
            parent.left = left; parent.right = right;
            pq.add(parent);
        }
        root = pq.peek();
        huffmanCodes.clear();
        generateCodes(root, "");
    }

    private static void displayCodes() {
        if (huffmanCodes.isEmpty()) {
            System.out.println("No Huffman codes generated yet!");
            return;
        }
        System.out.println("\nCharacter\tFrequency\tHuffman Code");
        for (Map.Entry<Character, String> entry :
huffmanCodes.entrySet())
            System.out.println(entry.getKey() + "\t\t" +
freqMap.get(entry.getKey()) + "\t\t" + entry.getValue());
    }
}
```

```

private static void encodeText() {
    if (huffmanCodes.isEmpty()) {
        System.out.println("Please build Huffman Tree first!");
        return;
    }
    StringBuilder encoded = new StringBuilder();
    for (char c : inputText.toCharArray())
        encoded.append(huffmanCodes.get(c));
    System.out.println("\nEncoded Text: " + encoded.toString());
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    while (true) {
        System.out.println("\n=== Huffman Encoding Menu ===");
        System.out.println("1. Enter input string (auto frequency)");
        System.out.println("2. Enter characters with frequencies manually");
        System.out.println("3. Build Huffman Tree & display codes");
        System.out.println("4. Encode the text");
        System.out.println("5. Exit");
        System.out.print("Enter your choice: ");
        int choice = sc.nextInt();
        sc.nextLine();

        switch (choice) {
            case 1:
                System.out.print("Enter the text: ");
                inputText = sc.nextLine();
                freqMap.clear();
                for (char c : inputText.toCharArray())
                    freqMap.put(c, freqMap.getOrDefault(c, 0) + 1);
                break;

            case 2:
                freqMap.clear();
                inputText = "";
                System.out.print("Enter number of characters: ");
                int n = sc.nextInt();
                sc.nextLine();
                for (int i = 0; i < n; i++) {
                    System.out.print("Character " + (i + 1) + ": ");
                    char c = sc.nextLine().charAt(0);
                    System.out.print("Frequency of " + c + ": ");
                    int f = sc.nextInt();
                    sc.nextLine();
                    freqMap.put(c, f);
                    inputText += String.valueOf(c).repeat(f);
                }
                break;

            case 3:
                if (freqMap.isEmpty()) {
                    System.out.println("Please enter input first!");
                } else {
                    buildHuffmanTree();
                    displayCodes();
                }
                break;

            case 4:
                if (inputText.isEmpty())
                    System.out.println("Enter input text first!");
                else
                    encodeText();
                break;

            case 5:
                System.out.println("Exiting program. Goodbye!");
        }
    }
}

```

```

        sc.close();
        return;
    default:
        System.out.println("Invalid choice!");
    }
}

```

## # Output

```

$ java Code_A2

=== Huffman Encoding Menu ===
1. Enter input string (auto frequency)
2. Enter characters with frequencies manually
3. Build Huffman Tree & display codes
4. Encode the text
5. Exit
Enter your choice: 1
Enter the text: meowmeow

=== Huffman Encoding Menu ===
1. Enter input string (auto frequency)
2. Enter characters with frequencies manually
3. Build Huffman Tree & display codes
4. Encode the text
5. Exit
Enter your choice: 3

Character      Frequency    Huffman Code
e              2           00
w              2           11
m              2           10
o              2           01

=== Huffman Encoding Menu ===
1. Enter input string (auto frequency)
2. Enter characters with frequencies manually
3. Build Huffman Tree & display codes
4. Encode the text
5. Exit
Enter your choice: 4

Encoded Text: 1000011110000111

=== Huffman Encoding Menu ===
1. Enter input string (auto frequency)
2. Enter characters with frequencies manually
3. Build Huffman Tree & display codes
4. Encode the text
5. Exit
Enter your choice: 5
Exiting program. Goodbye!

```