```
import java.util.Comparator;
import java.util.HashMap;
import java.util.Map;
import java.util.PriorityQueue;
import java.util.Scanner;

class Node {
    Character ch;
    Integer freq;
    Node left = null;
    Node right = null;

    Node(Character ch, Integer freq) {
        this.ch = ch;
        this.freq = freq;
    }

    public Node(Character ch, Integer freq, Node left, Node right) {
        this.ch = ch;
        this.freq = freq;
        this.left = left;
        this.right = right;
    }
}

public class HuffmanCode {
    public static void createHuffmanTree(String text) {
        if (text == null || text.length() == 0) {
            return;
        }
        Map<Character, Integer> freq = new HashMap<>();
        for (char c : text.toCharArray()) {
            freq.put(c, freq.getOrDefault(c, 0) + 1);
        }
        PriorityQueue<Node> pq = new PriorityQueue<>(Comparator.comparingInt(l -> l.freq));
        for (var entry : freq.entrySet()) {
            pq.add(new Node(entry.getKey(), entry.getValue()));
        }
        while (pq.size() != 1) {
            Node left = pq.poll();
            Node right = pq.poll();
            int sum = left.freq + right.freq;
            pq.add(new Node(null, sum, left, right));
        }
        Node root = pq.peek();
        Map<Character, String> huffmanCode = new HashMap<>();
        encodeData(root, "", huffmanCode);
        System.out.println("Huffman Codes of the characters are: " + huffmanCode);
        System.out.println("The initial string is: " + text);
        StringBuilder sb = new StringBuilder();
        for (char c : text.toCharArray()) {
            sb.append(huffmanCode.get(c));
        }
        System.out.println("The encoded string is: " + sb);
        System.out.print("The decoded string is: ");
        if (isLeaf(root)) {
            while (root.freq-- > 0) {
```

```
            System.out.print(root.ch);
        }
    } else {
        int index = -1;
        while (index < sb.length() - 1) {
            index = decodeData(root, index, sb);
        }
    }
}

public static void encodeData(Node root, String str, Map<Character, String> huffmanCode) {
    if (root == null) {
        return;
    }
    if (isLeaf(root)) {
        huffmanCode.put(root.ch, str.length() > 0 ? str : "1");
    }
    encodeData(root.left, str + '0', huffmanCode);
    encodeData(root.right, str + '1', huffmanCode);
}

public static int decodeData(Node root, int index, StringBuilder sb) {
    if (root == null) {
        return index;
    }
    if (isLeaf(root)) {
        System.out.print(root.ch);
        return index;
    }
    index++;
    root = (sb.charAt(index) == '0') ? root.left : root.right;
    index = decodeData(root, index, sb);
    return index;
}

public static boolean isLeaf(Node root) {
    return root.left == null && root.right == null;
}

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter the string to perform Huffman coding: ");
    String text = scanner.nextLine();
    scanner.close();
    createHuffmanTree(text);
}
}
```