```
import java.util.*;

// Node class for Huffman Tree
class Node {
    char ch;
    int freq;
    Node left, right;

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

    Node(int freq, Node left, Node right) {
        this.freq = freq;
        this.left = left;
        this.right = right;
    }
}

public class HuffmanEncoding {

    // Function to build Huffman Tree
    private static Node buildHuffmanTree(Map<Character, Integer> freqMap) {
        PriorityQueue<Node> pq = new PriorityQueue<>(Comparator.comparingInt(n ->
n.freq));

        for (Map.Entry<Character, Integer> entry : freqMap.entrySet()) {
            pq.add(new Node(entry.getKey(), entry.getValue()));
        }

        while (pq.size() > 1) {
            Node left = pq.poll();
            Node right = pq.poll();
            Node newNode = new Node(left.freq + right.freq, left, right);
            pq.add(newNode);
        }
        return pq.poll();
    }

    // Recursive function to generate codes from Huffman Tree
    private static void generateCodes(Node root, String code, Map<Character,
String> huffmanCode) {
        if (root == null) return;

        if (root.left == null && root.right == null) {
            huffmanCode.put(root.ch, code);
        }

        generateCodes(root.left, code + "0", huffmanCode);
        generateCodes(root.right, code + "1", huffmanCode);
    }

    // Main encoding function
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string to encode using Huffman Encoding:");
        String input = scanner.nextLine();
```

```
        // Step 1: Calculate frequency of each character
        Map<Character, Integer> freqMap = new HashMap<>();
        for (char ch : input.toCharArray()) {
            if (ch != ' ') { // Ignoring spaces
                freqMap.put(ch, freqMap.getOrDefault(ch, 0) + 1);
            }
        }

        // Step 2: Build Huffman Tree
        Node root = buildHuffmanTree(freqMap);

        // Step 3: Generate Huffman Codes
        Map<Character, String> huffmanCode = new HashMap<>();
        generateCodes(root, "", huffmanCode);

        // Displaying the Huffman codes for each character
        System.out.println("Huffman Codes: " + huffmanCode);

        // Step 4: Encode the input string
        StringBuilder encodedString = new StringBuilder();
        for (char ch : input.toCharArray()) {
            if (ch != ' ') {
                encodedString.append(huffmanCode.get(ch));
            }
        }

        // Output
        System.out.println("Encoded String: " + encodedString.toString());
    }
}
```