## DAA-Practical - 2

### CODE

```
import java.util.*;

class HuffmanNode {
    char data;
    int freq;
    HuffmanNode left, right;

    HuffmanNode(char data, int freq) {
        this.data = data;
        this.freq = freq;
        this.left = null;
        this.right = null;
    }
}

class Compare implements Comparator<HuffmanNode> {
    public int compare(HuffmanNode a, HuffmanNode b) {
        return a.freq - b.freq;
    }
}

public class HuffmanCoding {

    public static void generateCodes(HuffmanNode root, String code, Map<Character, String>
huffmanCode) {
        if (root == null)
            return;
        if (root.left == null && root.right == null) {
            huffmanCode.put(root.data, code);
        }
        generateCodes(root.left, code + "0", huffmanCode);
        generateCodes(root.right, code + "1", huffmanCode);
    }
    public static void buildHuffmanTree(String text) {
        Map<Character, Integer> freq = new HashMap<>();
        for (char ch : text.toCharArray()) {
            freq.put(ch, freq.getOrDefault(ch, 0) + 1);
        }

        PriorityQueue<HuffmanNode> pq = new PriorityQueue<>(new Compare());
        for (Map.Entry<Character, Integer> entry : freq.entrySet()) {
            pq.add(new HuffmanNode(entry.getKey(), entry.getValue()));
        }
        while (pq.size() > 1) {
            HuffmanNode left = pq.poll();
            HuffmanNode right = pq.poll();
            HuffmanNode sum = new HuffmanNode('\0', left.freq + right.freq);
            sum.left = left;
            sum.right = right;

            pq.add(sum);
        }
        HuffmanNode root = pq.peek();

        Map<Character, String> huffmanCode = new HashMap<>();
        generateCodes(root, "", huffmanCode);
        System.out.println("Huffman Codes:");
        for (Map.Entry<Character, String> entry : huffmanCode.entrySet()) {
            System.out.println(entry.getKey() + ": " + entry.getValue());
        }
        StringBuilder encodedString = new StringBuilder();
        for (char ch : text.toCharArray()) {
            encodedString.append(huffmanCode.get(ch));
        }

        System.out.println("\nEncoded string:\n" + encodedString);
    }
    public static void main(String[] args) {
        String text = "huffman encoding example";
        buildHuffmanTree(text);
    }
}
```

**Output**

```
Huffman Codes:
```

 : 1101
a: 000
c: 0010
d: 0011
e: 011
f: 1100
g: 10111
h: 10101
i: 0101
l: 10110
m: 1110
n: 100
o: 11110
p: 11111
u: 10100
x: 0100

Encoded string:
1010110100110011001110000100110101110000101111000110101100101111101011010000
011101111110110011