```
import java.util.PriorityQueue;
import java.util.Scanner;

class Node implements Comparable<Node> {
    char character;
    int frequency;
    Node left, right;

    public Node(char character, int frequency) {
        this.character = character;
        this.frequency = frequency;
        left = right = null;
    }

    @Override
    public int compareTo(Node other) {
        return Integer.compare(this.frequency, other.frequency);
    }
}

public class HuffmanEncoding {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter a string:");
        String input = scanner.next();

        System.out.println("Enter the frequency of each character (space-
        separated):");
        int[] frequencies = new int[input.length()];
        for (int i = 0; i < input.length(); i++) {
            frequencies[i] = scanner.nextInt();
        }

        System.out.println("Choose an option:");
        System.out.println("1. Huffman Encoding");
        System.out.println("2. Exit");
        int choice = scanner.nextInt();

        if (choice == 1) {
            huffmanEncoding(input, frequencies);
        }
    }

    public static void huffmanEncoding(String input, int[] frequencies) {
        PriorityQueue<Node> priorityQueue = new PriorityQueue<>();
        for (int i = 0; i < input.length(); i++) {
            priorityQueue.add(new Node(input.charAt(i), frequencies[i]));
        }

        while (priorityQueue.size() > 1) {
            Node left = priorityQueue.poll();
            Node right = priorityQueue.poll();
            Node newNode = new Node('\0', left.frequency + right.frequency);
            newNode.left = left;
            newNode.right = right;
            priorityQueue.add(newNode);
        }

        Node root = priorityQueue.poll();
        StringBuilder[] codes = new StringBuilder[input.length()];
        assignCodes(root, "", codes);

        System.out.println("Character\tFrequency\tCode \n
        ================================================");
        for (int i = 0; i < input.length(); i++) {
            System.out.println(input.charAt(i) + "\t" + frequencies[i] + "\t" +
```

```
          codes[i]);
        }

        int originalSize = input.length() * 8;
        int compressedSize = 0;
        for (int i = 0; i < input.length(); i++) {
            compressedSize += codes[i].length();
        }

        System.out.println("\nOriginal size (bits): " + originalSize);
        System.out.println("Compressed size (bits): " + compressedSize);

        StringBuilder encodedMessage = new StringBuilder();
        for (int i = 0; i < input.length(); i++) {
            encodedMessage.append(codes[i]);
        }

        System.out.println("Encoded message: " + encodedMessage);
    }

    public static void assignCodes(Node node, String code, StringBuilder[]
        codes) {
        if (node == null) {
            return;
        }

        if (node.character != '\0') {
            codes[node.character - 'a'] = new StringBuilder(code);
        }

        assignCodes(node.left, code + "0", codes);
        assignCodes(node.right, code + "1", codes);
    }
}
```

```
Enter a string:
abcdef
Enter the frequency of each character (space-separated):
5 9 12 13 16 45
Choose an option:
1. Huffman Encoding
2. Exit
1
Character       Frequency       Code
====================================
a               5               1100
b               9               1101
c               12              100
d               13              101
e               16              111
f               45              0

Original size (bits): 48
Compressed size (bits): 18
Encoded message: 110011011001011110
```