Import java.io.\*;

Import java.util.\*;

Class HuffmanNode {

Int data;

Char c;

HuffmanNode left, right;

}

Class MyComparator implements Comparator<HuffmanNode> {

Public int compare(HuffmanNode x, HuffmanNode y) {

Return x.data – y.data;

}

}

Public class HuffmanCompression {

Public static void printCode(HuffmanNode root, String s) {

If (root.left == null && root.right == null && Character.isLetter(root.c)) {

System.out.println(root.c + “:” + s);

Return;

}

printCode(root.left, s + “0”);

printCode(root.right, s + “1”);

}

Public static void main(String[] args) throws IOException {

String testString = “This is a test string for Huffman coding”;

Char[] charArray = testString.toCharArray();

Map<Character, Integer> freqMap = new HashMap<>();

For (char c : charArray) {

freqMap.put(c, freqMap.getOrDefault(c, 0) + 1);

}

PriorityQueue<HuffmanNode> pq = new PriorityQueue<>(freqMap.size(), new MyComparator());

For (Map.Entry<Character, Integer> entry : freqMap.entrySet()) {

HuffmanNode hn = new HuffmanNode();

Hn.c = entry.getKey();

Hn.data = entry.getValue();

Hn.left = null;

Hn.right = null;

Pq.add(hn);

}

HuffmanNode root = null;

While (pq.size() > 1) {

HuffmanNode x = pq.poll();

HuffmanNode y = pq.poll();

HuffmanNode f = new HuffmanNode();

f.data = x.data + y.data;

f.c = ‘-‘;

f.left = x;

f.right = y;

root = f;

pq.add(f);

}

printCode(root, “”);

// Compression

String encodedString = “”;

For (char c : charArray) {

encodedString += getCode(root, c);

}

System.out.println(“Encoded String: “ + encodedString);

// Decompression

StringBuilder decodedString = new StringBuilder();

HuffmanNode temp = root;

For (int I = 0; I < encodedString.length(); i++) {

Char code = encodedString.charAt(i);

If (code == ‘0’) {

Temp = temp.left;

} else {

Temp = temp.right;

}

If (temp.left == null && temp.right == null) {

decodedString.append(temp.c);

temp = root;

}

}

System.out.println(“Decoded String: “ + decodedString);

}

Public static String getCode(HuffmanNode root, char c) {

If (root == null) return “”;

If (root.c == c) return “”;

String leftCode = getCode(root.left, c);

If (leftCode != null) return “0” + leftCode;

String rightCode = getCode(root.right, c);

If (rightCode != null) return “1” + rightCode;

Return null;

}

}