Κλάση TreeNode :

Καταρχάς κάνουμε import το 'java.io.Serializable' ώστε να έχουμε την δυνατότητα να κανουμε Serialization
για την αναπαράσταση του Huffman δέντρου μας

Κάνουμε implements την Comparable με μεταβλητή τον κόμβο ώστε να είναι συγκρίσιμος. Η κλάση έχει μεταβλητές ένα int frequency, δηλαδή την συχνοτητά, ενα int c, όπου χρησιμοποιήτε ως ο χαρακτήρας για την κάθε συχνότητα. Ακόμα έχει TreeNode left και right, δηλαδή το αριστερό και δεξί παιδί του κόμβου μας (1). Στην συνέχεια δημιουργούμε 'setters' και 'getters' για τα πεδία της κλάσης (2). Έπειτα φτιάχνουμε τον constructor της κλάσης μας(3). Τέλος χρησιμοποιουμε την μέθοδο 'compareTo' μέσω της 'Comparable' για να μπορούμε να συγκρίνουμε τους κόμβους. Συγκεκριμένα μέσω ενός 'if...else if' κάνουμε τις συγκρίσεις για τις συχνότητες των κόμβων και ανάλογα ποιά συχνότητα απο αυτες που έχουμε δώσει είναι μεγαλύτερη μας επιστρέφει και ανάλογο νούμερο (4).

Interface MinHeap και κλάση ArrayMinHeap :

Την δυαδική σωρό μας, την πήραμε σχετικά αυτούσια από το εργαστήριο που την υλοποιήσαμε, ωστόσο κάναμε τις απαραίτητες αλλαγές για το πρόγραμμα μας. Στο 'interface MinHeap' δηλώνονται οι συναρτήσεις που υλοποιούνται στην κλάση 'ArrayMinHeap', και γίνεται extends η Comparable με παράμετρο το TreeNode, όπως και στην 'ArrayMinHeap'. Ωστόσο στην ArrayMinHeap γίνεται implements το interface 'MinHeap' με παράμετρο το TreeNode και το 'interface Serializable. Ουσιαστίκα έχουμε θέσει ως element το TreeNode, αφου ασχολούμαστε αποκλειστίκα με τους κόμβους (5).

Κλαση Huffman :

Στην αρχή της κλάσης κάνουμε τα import, 'java.io.File' καθώς ασχολούμαστε με αρχεία, java.io.IOException ώστε να μας εμφανίζει exception στην οθόνη σε περίπτωση λάθους στον κωδικά μας, java.io.Scanner αφού χρησιμοποιούμε την Scanner για να μπορέσουμε να διαβάσουμε το αρχείο frequency.dat, 'java.io.ObjectOutputStream' και 'java.io.FileOutputStream' για να μπορέσουμε να γράψουμε σαν object το root του δέντρου μας στο αρχείο tree.dat και τελος το 'java.io.Serializable' ώστε να έχουμε την δυνατότητα να κανουμε Serialization.

Στην αρχή της κλάσης κάνουμε implements το interface Serializable. Στην main αρχικά κάνουμε ενα 'throw IOException'. Μετά με το FileOutputStream και το ObjectOutputStream φτιάχνουμε το αρχείο tree.dat και το ορίζουμε ενα objectstream προς το tree.dat. Επιπλέον με την Scanner παίρνουμε τις συχνότητες από το αρχείο frequency.dat (6).

Στην συνέχεια βάζουμε final το arrayLenght ως 128 αφού δεν θέλουμε να αλλάζει μέγεθος. Φτιάχνουμε έναν πίνακα frequencyArray, δυναμικά με μέγεθος 128, όσοι και οι χαρακτήρες του Ascii table που μας ενδιαφέρουν. Μετά με ένα while με συνθήκη 'scanner.hasNextInt()', δηλαδή όσο στο αρχείο υπάρχει και άλλος int, παίρνουμε όλα τα στοιχεία του frequency.dat και τα βάζουμε στον πίνακα frequencyArray (7).

Ακόμα φτίαχνουμε την δυαδίκη σωρό και την ονομάζω heap με element το treeNode. Ορίζω το root ως null και μετά με ενα for γεμίζω την heap με

TreeNodes που παίρνουν ως παραμέτρους τον πινακα frequencyArray που έχει τις συχνότητες, ενα int i που το μετατρέπουμε σε char για να έχουμε μία συχνότητα σε συγκεκριμένο χαρακτήρα, και το δέξι και αριστερό παιδι που τα βάζουμε null. Η heap γεμίζει με την 'heap.insert()' που είναι ορισμένη στο ArrayMinHeap (8).

Με ένα while με συνθήκη 'heap.size() > 1', δηλαδή όσο η heap έχει περισσότερους κόμβους από έναν, παίρνει τους δύο μικρότερους κόμβους της heap με την 'heap.findMin', τα βάζει σε δύο κόμβους x και y και έπειτα τα διαγράφει με την 'heap.deleteMin()' Στην συνέχεια φτιάχνει νέο κόμβο, node με συχνότητα το άθροισμα των συχνοτήτων των x και y, χωρίς χαρακτήρα αφού δεν είναι φύλλο, και αριστερό παιδί το x και δεξί το y. Και μετά βάζει τον κόμβο node που φτιάξαμε ξανά μέσα στην heap με την 'heap.insert()' (9).

Τέλος με το 'ObjectOutputStream.writeObject' πηγαίνει στο tree.dat και τυπώνει το root και με το 'ObjectOutputStream.close()' κλείνει το objectstream που είχαμε φτιάξει (10).

Η αναπαράσταση του root στο tree.dat δεν είναι δυνατόν να την καταλάβουμε ωστόσο την ελέγξαμε με την 'readObject' και νομίζουμε οτι δουλέυει σωστά.

Link που συμβουλευτήκαμε : https://www.geeksforgeeks.org/huffman-coding-greedy-algo-3/

1)
```
public class TreeNode implements Comparable<TreeNode>, Serializable {
  private int frequency;
  private int c;
  private TreeNode left;
  private TreeNode right;
```

2)
```
public int getFrequency() {
    return frequency;
}

public void setFrequency(int frequency) {
    this.frequency = frequency;
}

public int getC() {
    return c;
}

public void setC(int c) {
    this.c = c;
}

public TreeNode getLeft() {
    return left;
}

public void setLeft(TreeNode left) {
    this.left = left;
}
```

```
    public TreeNode getRight() {
        return right;
    }

    public void setRight(TreeNode right) {
        this.right = right;
    }
```

3)
```
public TreeNode(int frequency, int c, TreeNode left, TreeNode right)
{
         this.frequency = frequency;
         this.c = c;
         this.left = left;
         this.right = right;
     }
```

4)
```
   public int compareTo(TreeNode treeNode) {
        if (this.frequency > treeNode.frequency) {
            return 1;
        } else if (this.frequency == treeNode.frequency) {
            return 0;
        }

        return -1;
    }
```

5)
```
public interface MinHeap<TreeNode extends Comparable<TreeNode>> {

    void insert(TreeNode k);

    TreeNode findMin();

    TreeNode deleteMin();

    boolean isEmpty();

    int size();

    void clear();
    }
```

```
   public class ArrayMinHeap<TreeNode extends Comparable<TreeNode>>
implements MinHeap<TreeNode>, Serializable {
```

6)
```
public class Huffman implements Serializable {
    public static void main(String[] args) throws IOException {

        FileOutputStream outputStream = new FileOutputStream("tree.dat");
        ObjectOutputStream objectOutputStream = new
ObjectOutputStream(outputStream);
```

```
        Scanner scanner = new Scanner(new
File("src/main/java/com/ergasia02/maven/java/frequency.dat"));


7)  final int arrayLength = 128;
        int[] frequencyArray = new int[128];


        while (scanner.hasNextInt()) {
            frequencyArray[i++] = scanner.nextInt();
        }



8)   MinHeap<TreeNode> heap = new ArrayMinHeap<TreeNode>();

     TreeNode root = null;
     for (i = 0; i < arrayLength; i++) {
         TreeNode k = new TreeNode(frequencyArray[i], (char) i, null,
null);
         heap.insert(k);
     }


9)    while (heap.size() > 1) {
            TreeNode x = heap.findMin();
            heap.deleteMin();

            TreeNode y = heap.findMin();
            heap.deleteMin();

            TreeNode node = new TreeNode(x.getFrequency() +
y.getFrequency(), '-', x, y);
            root = node;

            heap.insert(node);
        }



10)   objectOutputStream.writeObject(root);
        objectOutputStream.close();
```

Τα ονοματά μας :
Παναγιώτης Κωλέτσης
Παναγιώτης Πετεινάρης
Παναγιώτης Χάρος