Data Structures
Δεύτερο Παραδοτέο

Γεώργιος-Νεκτάριος Εγγλέζος it21824
Αντώνης Δρόσος it218115
Ανδρέας Σαραντόπουλος it218140

Εισαγωγή

Για την υλοποίηση του δεύτερου παραδοτέου της εργασίας υλοποιήσαμε 3 καινούργιες κλάσεις (Πέρα από τις προηγούμενες στο Πρώτο Παραδοτέο)

1. Second.class (Second.java)
2. HuffmanTree.class(Huffman.java)
3. HuffmanNode.class(HuffmanNode.java)

Αλλαγές που έγιναν στο πρώτο παραδοτέο:

1. Μετονομασία του Text.java σε First.java (Πρώτο Παραδοτέο)
2. Στην μέθοδο που βλέπαμε και μετρούσαμε τους χαρακτήρες είχαμε βάλει να προσπερνάει τα κενά (space). Τώρα το αφαιρέσαμε και τα μετράμε και αυτά.

Second.java

Καλείται από την main και είναι υπεύθυνη για το διάβασμα του frequencies.dat που είχαμε στο πρώτο παραδοτέο και δημιουργεί ένα αρχικό ArrayList με όλα τα frequencies. Τέλος καλεί την συνάρτηση που υπάρχει στο HuffmanTree.java για την δημιουργία του τελικού δέντρου.

HuffmanTree.java

Περιέχει την συνάρτηση που φτιάχνει το τελικό δέντρο.

HuffmanNode.java

Περιέχει το constructor που χρησιμοποιείται ως τύπος για το ArrayList μαζί με τους getters και τους setters και 2 μεθόδους , μια για την ταξινόμηση της λίστας και μια για να ελέγχω αν το node είναι φύλο ή όχι που θα χρησιμοποιήσουμε στο επόμενο παραδοτέο.

Κώδικας

Main.java

Μέσα στην main αφού έχει τελειώσει ο κώδικας του πρώτου Παραδοτέου καλώ την συνάρτηση Second.*secondDeliverable*(); για να αρχίσουμε με το διάβασμα του frequencies.dat.

Second.java

secondDeliverable()

```
public static void secondDeliverable() throws IOException{

makeArrayList();
if (hNodes.isEmpty()){
  System.out.println("There were no frequencies to make the tree");
}else{
  HuffmanNode finalNode = HuffmanTree.buildTree(hNodes); // I get the final
tree

  makeTreeDat(finalNode); //Makes the tree.dat file
}
```

Αρχικά βρισκόμαστε μέσα στην μέθοδο ***secondDeliverable*** η οποία καλεί την ***makeArrayList*** για το διάβασμα του frequencies.dat και την δημιουργία του ArrayList hNodes με τις συχνότητες.

Στην συνέχεια αν η λίστα hNodes (**“private static ArrayList<HuffmanNode> hNodes = new ArrayList<>()”**)δεν είναι άδεια καλούμε την μέθοδο buildTree() της κλάσης HuffmanTree για να πάρουμε το τελικό δέντρο , δίνοντας ως παράμετρο την λίστα με τις συχνότητες hNodes. Αν η λίστα είναι άδεια τελειώνει το πρόγραμμα.

Τέλος καλούμε την συνάρτηση makeTreeDat() δίνοντας το τελικό δέντρο huffman finalNode για να το αποθηκεύσει στο αρχείο tree.dat.

readFrequenciesDat()

```
private static BufferedReader readFrequenciesDat(){
  try {
     BufferedReader inputStream = new BufferedReader(new
FileReader("frequencies.dat"));
     return inputStream;
} catch (FileNotFoundException ex) {
System.out.println("frequencies.dat not found.");
     return null;
  }
}
```

Χρησιμοποιεί έναν BufferedReader για το διάβασμα όλου του αρχείου Frequencies.dat και το επιστρέφει. Σε περίπτωση που δεν το βρει εμφανίζει μήνυμα λάθους.

makeArrayList()

```
private static void makeArrayList() throws IOException {

   String temp2;
   int asciiNumber = 0;
   BufferedReader file = readFrequenciesDat();


   while ((currentLine = file.readLine()) != null) {

      StringTokenizer st = new StringTokenizer(currentLine);
      while (st.hasMoreTokens()) { //To break all the frequencies
         String temp = st.nextToken(",");

         int index = 0;

         StringTokenizer st2 = new StringTokenizer(temp);
         while (st2.hasMoreTokens()) {
            temp2 = st2.nextToken(":"); //To separate the frequency from
the ascii number
            if (index == 0) {
               asciiNumber = Integer.parseInt(temp2);
            }
            if (index == 1) //skip this to get only the frequencies
(frequencies.dat format was asciiNum:Frequency)
            {
               if (Integer.parseInt(temp2) != 0) {
                  hNodes.add(new HuffmanNode(asciiNumber,
Integer.parseInt(temp2), null, null));
               }
            }
            index++;
         }
      }
```

```
    }
}
```

Αρχικά καλεί την μέθοδο ***readFrequenciesDat();*** που εξηγήσαμε από πάνω αποθηκεύοντας το κείμενο που επιστρέφει σε μια μεταβλητή file.

Στην συνέχεια χρησιμοποιήσαμε δύο **StringTokenizer** ένα για το σπάσιμο όλων των συχνοτήτων που είναι της μορφής "**νούμεροAscii:νούμεροΣυχνότητας**" και χωρίζονται με κόμμα (",") και ένα για το σπάσιμο του Ascii και της συχνότητας (":") για την αποθήκευση στο **ArrayList hNodes**. Η hNodes αποθηκεύει στιγμιότυπα της κλάσης HuffmanNode ("***hNodes***.add(new HuffmanNode(asciiNumber, Integer.***parseInt***(temp2), null, null)); ") και από την στιγμή που είναι φύλλα βάζουμε το αριστερό και δεξί παιδί ως null, ενώ τέλος χρειάζεται και να μετατρέψουμε το temp2 και το asciiNumber από string σε int με το **Integer.parseInt()**.

makeTreeDat()

```
private static void makeTreeDat(HuffmanNode tree) throws IOException {
    FileOutputStream fos = new FileOutputStream("tree.dat");
    ObjectOutputStream oos = new ObjectOutputStream(fos);

    oos.writeObject(tree);
    oos.close();
    fos.close();
    System.out.println("tree.dat is ready in the Project Folder!");
}
```

Αφού έχουμε πάρει το τελικό δέντρο, αποθηκεύουμε απευθείας το αντικείμενο του στο αρχείο tree.dat χρησιμοποιώντας Java Object Serialization ακολουθώντας τις οδηγίες του που υπήρχαν στην εκφώνηση.

HuffmanNode

Η κλάση κάνει implement το **Serializable** για να μπορεί να αποθηκευτεί μετά στο αρχείο tree.dat.

HuffmanNode()

```
HuffmanNode(int ch, int freq, HuffmanNode left, HuffmanNode right) {
  this.ch = ch;
  this.freq = freq;
  this.left = left;
  this.right = right;
}
```

Περιέχει έναν contructor με όλες τις μεταβλητές και τα getters και setters τους και τέλος μια μέθοδο **sortArrayList** που παίρνει ως παράμετρο την λίστα με τα nodes και τα ταξινομεί χρησιμοποιώντας lambda expressions με βάση το frequency.

sortArrayList()

```
public static void sortArrayList(ArrayList<HuffmanNode> hNodes) {
  hNodes.sort(Comparator.comparing(o -> o.freq));
}
```

HuffmanTree

buildTree()

```
public static HuffmanNode buildTree(ArrayList<HuffmanNode> hNodes) {

  while (hNodes.size() > 1) {
    HuffmanNode.sortArrayList(hNodes);

    HuffmanNode left = hNodes.get(0);
    hNodes.remove(0);

    HuffmanNode right = hNodes.get(0);
    hNodes.remove(0);

    HuffmanNode parent = new HuffmanNode(-1, left.getFreq() +
right.getFreq(), left, right);
    hNodes.add(parent);

  }
  System.out.println("The tree is ready!");
  return hNodes.get(0);
}
```

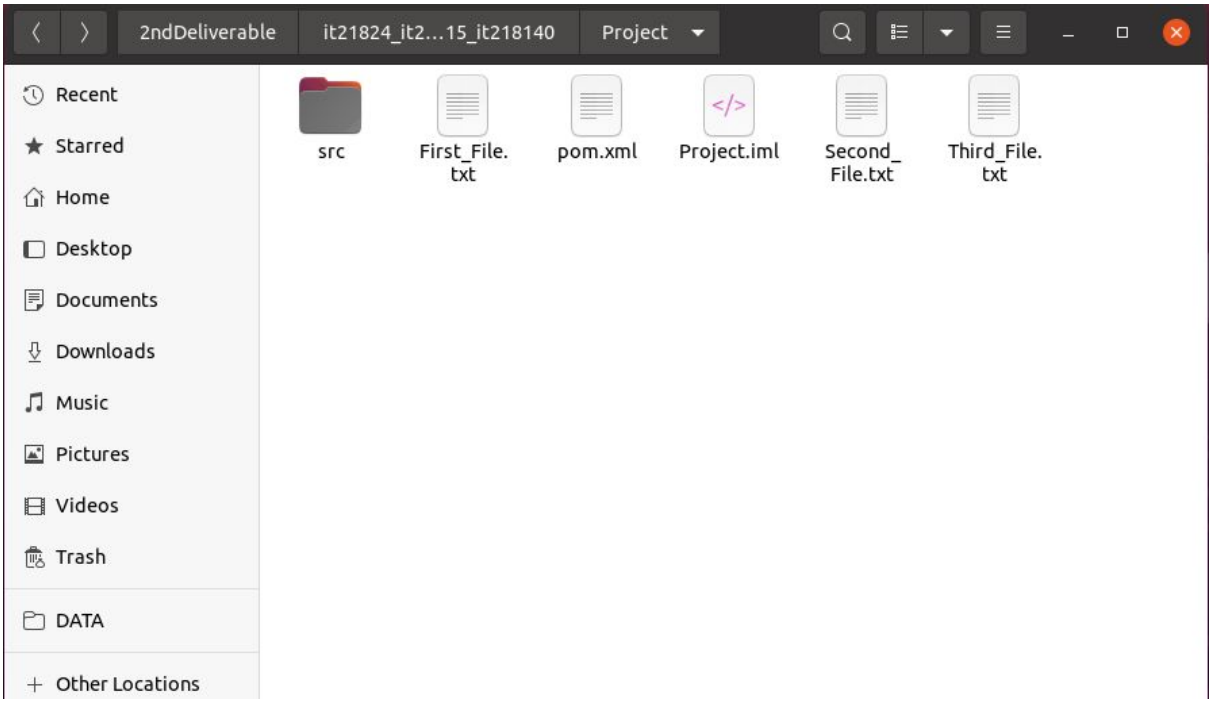
Για την δημιουργία του δέντρου καλούμε την μέθοδο αυτή η οποία τρέχει μια επανάληψη όσο υπάρχουν πάνω από ένα αντικείμενα αποθηκευμένα στο ArrayList hNodes. Σε κάθε επανάληψη ταξινομώ την λίστα και κρατάω σε προσωρινές μεταβλητές τα δύο

μικρότερα nodes που βρίσκονται στην θέση 0 κάθε φορά και τα αφαιρώ από την λίστα (hNodes.remove(index)). Με βάση αυτά φτιάχνω ένα προσωρινό αντικείμενο “parent” της κλάσης HuffmanNode και βάζω στον constructor τα δύο nodes από πριν ως αριστερό και δεξί παιδί. Τέλος αποθηκεύω το προσωρινό node Parent στο arrayList (hNodes.add(index)) και επαναλαμβάνω τα προηγούμενα βήματα.

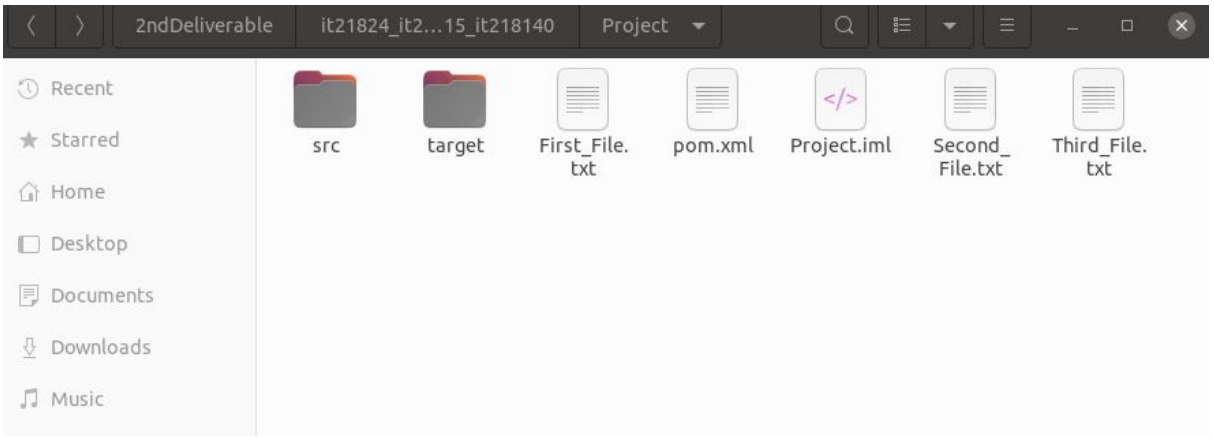
Ενδεικτικά Αποτελέσματα

Περίπτωση 1

Αρχικά στον φάκελο του Project υπάρχουν μόνο τα τρία αρχεία κειμένου από το πρώτο παραδοτέο.



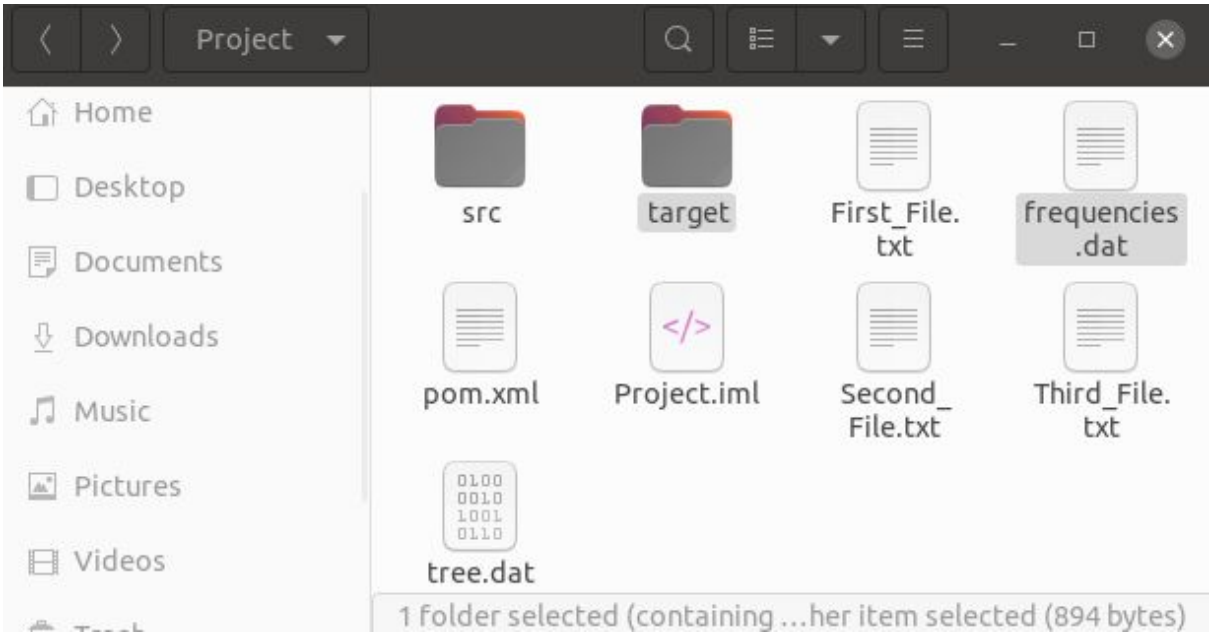
Τρέχουμε την εντολή “**mvn package**” για να φτιαχτεί ο φάκελος target με το αρχείο jar



Τέλος τρέχουμε την εντολή “**java -cp target/Project-1.0-SNAPSHOT.jar org.hua.project.Main**” για να τρέξει το πρόγραμμα που μας εμφανίζει αυτές τις τρεις γραμμές και δημιουργεί τα δύο αρχεία frequencies.dat και tree.dat.

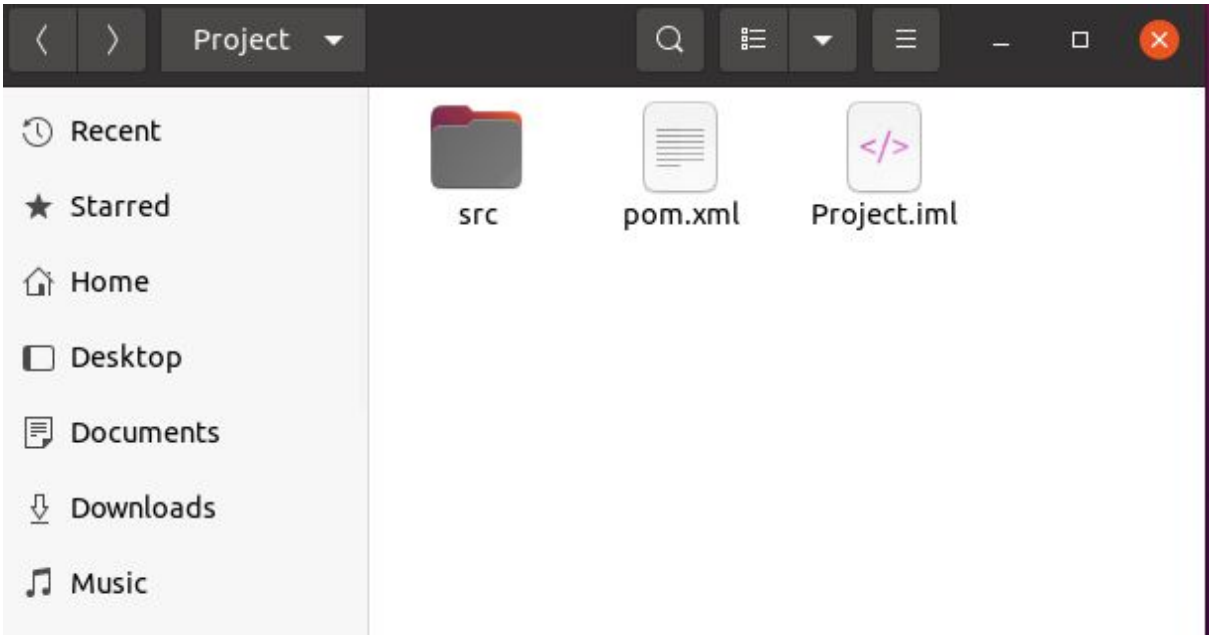
```
geonek@GeoLaptop:~/Downloads/Data-Structures-main/2ndDeliverable/it21824_it218115_i
t218140/Project$ java -cp target/Project-1.0-SNAPSHOT.jar org.hua.project.Main
Frequencies.dat is ready in the project folder!
The tree is ready!
tree.dat is ready in the Project Folder!
```

και έτσι στο τέλος έχουμε αυτά τα αρχεία στο root του φακέλου Project:



Περίπτωση 2

Η μόνη περίπτωση που δεν θα φτιαχτεί το αρχείο tree.dat είναι αν λείπουν όλα τα αρχεία κειμένου από το πρώτο παραδοτέο .Στην περίπτωση αυτή ενώ θα φτιαχτεί ένα αρχείο frequencies.dat ,δεν θα φτιαχτεί το δέντρο γιατί όλες οι συχνότητες θα είναι μηδενικές και θα εμφανίσει κατάλληλο μήνυμα.




```
geonek@GeoLaptop:~/Downloads/Data-Structures-main/2ndDeliverable/it21824_it218115
_it218140/Project$ java -cp target/Project-1.0-SNAPSHOT.jar org.hua.project.Main
file First_File was not found.
file Second_File was not found.
file Third_File was not found.
Frequencies.dat is ready in the project folder!
There were no frequencies to make the tree
```