Δομές Δεδομένων

Τρίτο Παραδοτέο

Γεώργιος-Νεκτάριος Εγγλέζος it21824
Αντώνης Δρόσος it218115
Ανδρέας Σαραντόπουλος it218140

Εισαγωγή

Για την υλοποίηση του Τρίτου Παραδοτέου της εργασίας υλοποιήσαμε Δύο καινούργιες κλάσεις:

1. Την **Third** που βρίσκεται στο Third.java
2. Την **Code** που βρίσκεται μέσα στο HuffmanTree.java

Αλλαγές σε προηγούμενα παραδοτέα:

Η μόνη αλλαγή ήταν η προσθήκη κώδικα στο HuffmanTree.java που θα αναφερθούμε αναλυτικά παρακάτω.

Third.java

Καλείται από την main και είναι υπεύθυνη για το διάβασμα του tree.dat που υλοποιήσαμε στο δεύτερο παραδοτέο και διαβάζει κατευθείαν το Object του δέντρου χρησιμοποιώντας Java Object Serialization που μας είχε δοθεί στην εκφώνηση του δεύτερου παραδοτέου. Καλεί την συνάρτηση που υπάρχει στο HuffmanTree.java για την δημιουργία των κωδικών για κάθε ascii και τέλος τα αποθηκεύει σε ένα αρχείο codes.dat

Code

Χρησιμοποιείται μόνο ως τύπος από ένα arrayList **ArrayList<code>** για την αποθήκευση όλων των κωδικών με τους χαρακτήρες που αντιπροσωπεύουν. Περιέχει μόνο έναν Constructor , Getters και Setters.

Κώδικας

Main.java

Μέσα στην main αφού έχει τελειώσει ο κώδικας του πρώτου και δεύτερου Παραδοτέου καλώ την συνάρτηση **Third.*thirdDeliverable*()**; μέσα στην οποία ρυθμίζεται όλο το τρίτο παραδοτέο.

Third.java

thirdDeliverable()

```
public static void thirdDeliverable() throws IOException {
   hTree=readTreeDat();
   ArrayList<HuffmanTree.code> CodeList=HuffmanTree.getTheCodes(hTree);
   makeCodesDatFile(CodeList);
}
```

Αρχικά καλεί την μέθοδο readTreeDat() που γυρίζει το δέντρο από το αρχείο tree.dat στην μεταβλητή **static HuffmanNode *hTree*;**
Στην συνέχεια για να πάρει τους κώδικες για κάθε ascii καλεί την μέθοδο getTheCodes() που βρίσκεται στην κλάση HuffmanTree που παίρνει ως παράμετρο το δέντρο .
Τέλος για την αποθήκευση των κωδικών καλεί την μέθοδο *makeCodesDatFile*() που παίρνει ως παράμετρο την λίστα codeList.

readTreeDat()

```
private static HuffmanNode readTreeDat(){
   try {
      FileInputStream fis = new FileInputStream("tree.dat");
      ObjectInputStream ois = new ObjectInputStream(fis);


      HuffmanNode tree = (HuffmanNode) ois.readObject();

      ois.close();
      fis.close();
      return tree;
   } catch (FileNotFoundException e) {
      System.out.println("File not found");
   } catch (IOException e) {
      System.out.println("Error initializing stream");
   } catch (ClassNotFoundException e) {
      e.printStackTrace();
   }
   return null;
}
```

Χρησιμοποιούμε τον κώδικα που μας δόθηκε στο προηγούμενο παραδοτέο για το διάβασμα ενός HuffmanNode από το αρχείο tree.dat κάνοντας και τους κατάλληλους ελέγχους με try-catch και το επιστρέφουμε.

makeCodesDatFile()

```
private static void makeCodesDatFile(ArrayList<HuffmanTree.code> arr)
throws IOException {
  PrintWriter out = new PrintWriter(new BufferedWriter(new FileWriter("codes.dat")));
  for (int i=0;i<arr.size();i++)
  {
     out.println(arr.get(i).getC()+":"+arr.get(i).getCode());   }
  out.close();
  System.out.println("codes.dat is ready in the project folder!");
}
```

Αφού έχουμε πάρει την λίστα με τους κωδικούς, χρησιμοποιούμε ένα BufferedWriter για να την αποθηκεύσουμε στο αρχείο codes.dat που δημιουργείται στο root του φακέλου.
Χρησιμοποιούμε το out.println() για να εκτυπώνει όλους τους κώδικες σε διαφορετικές γραμμές και τέλος κλείνουμε τον bufferedWriter με το out.close().

HuffmanNode

isLeaf()

```
public Boolean isLeaf(){return this.getRight()==null && this.getLeft()==null;}
```

Μια συνάρτηση για να ελέγχουμε αν ένα node είναι φύλο ή όχι βλέποντας αν έχει παιδιά. Την είχαμε φτιάξει στο προηγούμενο παραδοτέο αλλά επειδή την χρησιμοποιήσαμε σε αυτό το παραδοτέο την αναφέρουμε.

HuffmanTree

getTheCodes()

```
public static ArrayList<code> getTheCodes(HuffmanNode tree){
  ArrayDeque<String> codeTillNow = new ArrayDeque<>();
  ArrayList<code> codeList=new ArrayList<>();
  makeCodes(codeList,tree,codeTillNow,null);
  return codeList;
}
```

Παίρνει ως παράμετρο το δέντρο.Φτιάχνει ένα ArrayDeque για να αποθηκεύουμε τον κώδικα (0,1) σε μορφή String και φτιάχνει και ένα ArrayList τύπου code για την αποθήκευση των κωδικών. Τέλος καλεί την μέθοδο makeCodes() δίνοντας ως παραμέτρους την λίστα codeList, το δέντρο, την ουρά codeTillNow και ένα null που θα εξηγήσουμε από κάτω τι είναι.

makeCodes()

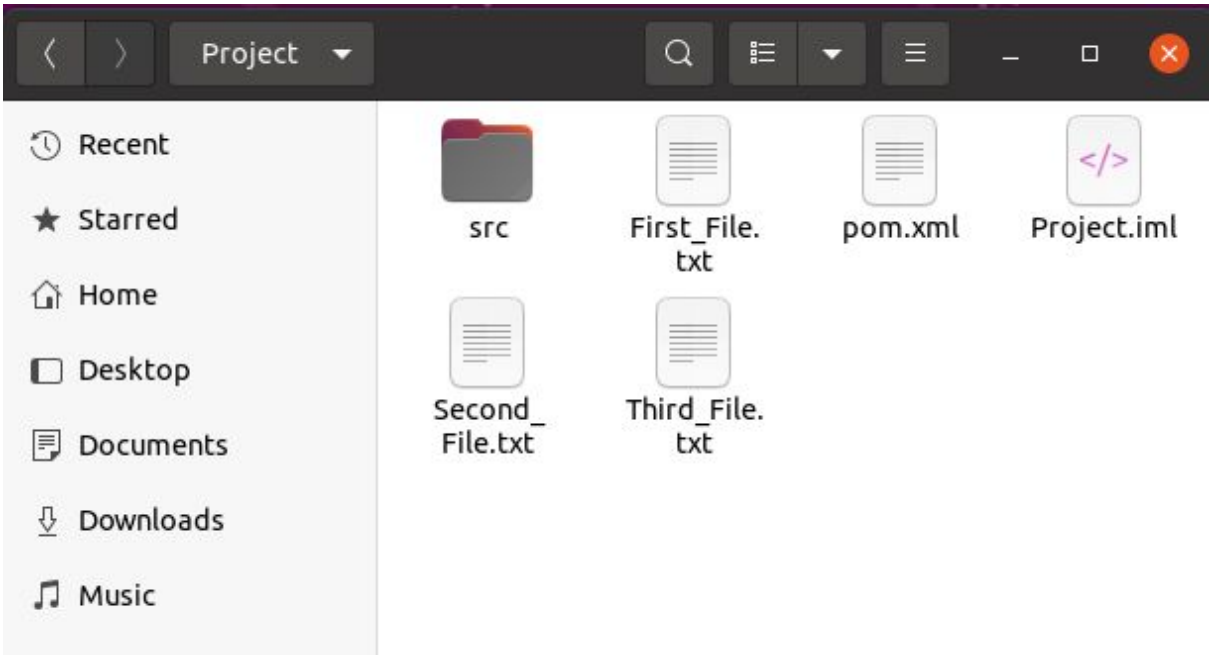
```
public static void makeCodes(ArrayList<code> CodeList,HuffmanNode
node,ArrayDeque<String> CodeTillNow,String bit) {
  if(bit!=null){
     CodeTillNow.push(bit);
  }
  if (!node.isLeaf()){
     makeCodes(CodeList,node.getLeft(),CodeTillNow,"0");
     CodeTillNow.removeFirst();
     makeCodes(CodeList,node.getRight(),CodeTillNow,"1");
     CodeTillNow.removeFirst();
  }else{
     String tmp="";

     for(Iterator itr=CodeTillNow.descendingIterator(); itr.hasNext();)
     {
        tmp=tmp+itr.next().toString();
     }
     CodeList.add(new code((char)node.getAscii(),tmp));
  }
}
```

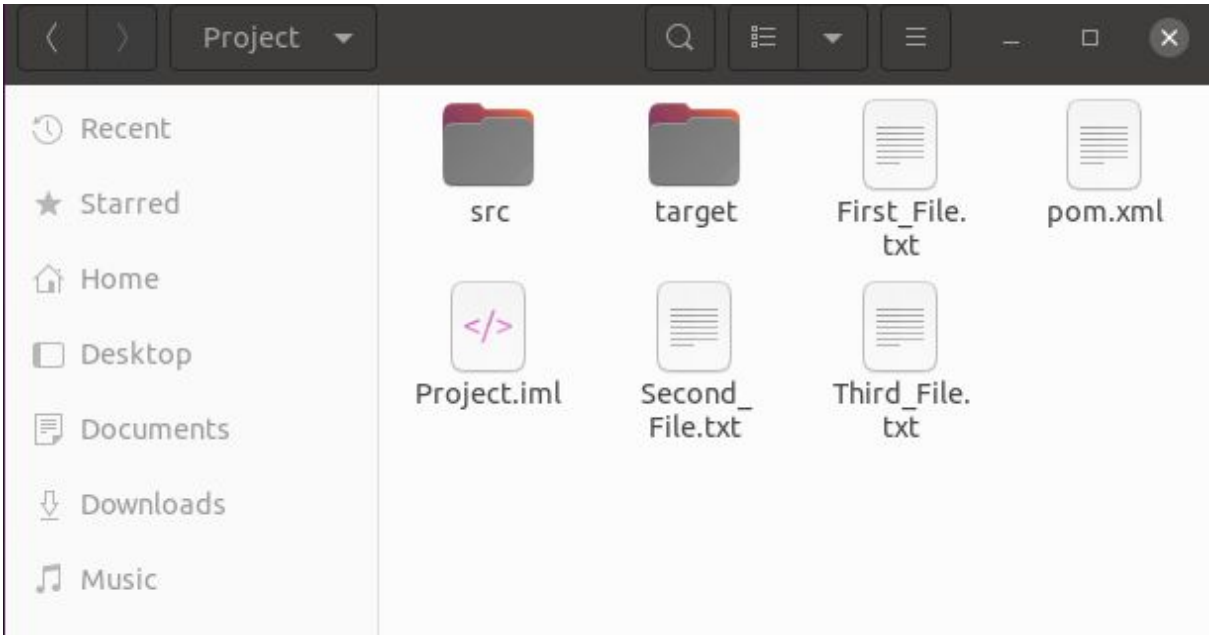
Για την δημιουργία του κωδικού υπάρχουν δύο περιπτώσεις.Η πρώτη περίπτωση είναι αν το node δεν είναι φύλο στην οποία περίπτωση καλούμε ξανά την συνάρτηση δύο φορές αλλά δίνουμε το αριστερό και δεξί node και ένα χαρακτήρα 0 ή 1 αντίστοιχα που προστίθενται στην ουρά με τον κωδικό. Την πρώτη φορά που κλήθηκε η μέθοδος από την **getTheCodes()** αντί για 0 και 1 δίνουμε null ώστε με έναν έλεγχο να μην προσθέσουμε τίποτα στην ουρά με τον κώδικα. Στην περίπτωση που το node είναι φύλο τρέχουμε όλη την ουρά με έναν iterator και προσθέτουμε το νούμερο κάθε φορά σε ένα string tmp.Όταν πάρουμε τον κωδικό τον αποθηκεύουμε στην λίστα codeList μαζί με το γράμμα που αντιπροσωπεύει. Τέλος γυρίζει στην προηγούμενη κλήση της μεθόδου όπου και αφαιρεί το bit που προσθέσαμε στο queue τελευταία φορά που έτρεξε η μέθοδος .

Ενδεικτικό Αποτέλεσμα

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



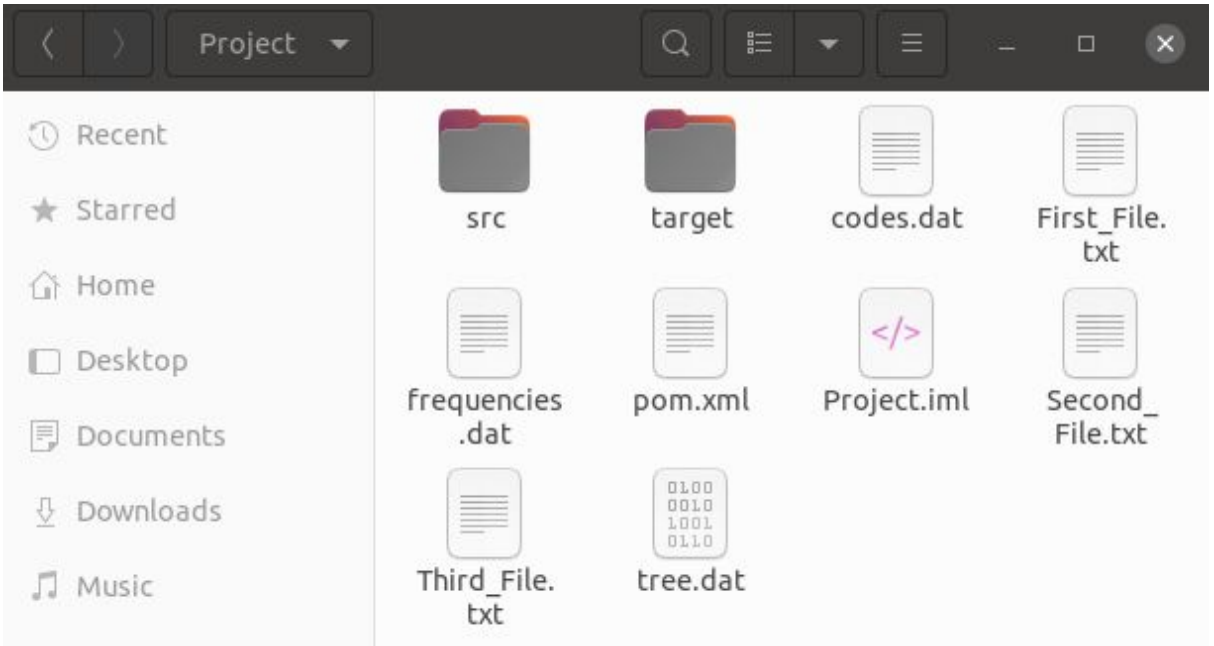
Τρέχουμε την εντολή “**mvn package**” για να φτιαχτεί ο φάκελος target με το αρχείο jar



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 τέσσερις γραμμές και δημιουργεί τα τρία αρχεία frequencies.dat , tree.dat και codes.dat.

```
geonek@GeoLaptop:~/Downloads/Data-Structures-main/3rd Deliverable/it21824_it2181
15_it218140/Project$ java -cp target/Project-1.0-SNAPSHOT.jar org.hua.project.Ma
in
Frequencies.dat is ready in the project folder!
The tree is ready!
tree.dat is ready in the Project Folder!
codes.dat is ready in the project folder!
```

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



Τέλος τρέχουμε την εντολή “**cat codes.dat**” για να δούμε τι αποθηκεύτηκε στ

```
_it218140/Project$ cat codes.dat
e:000
B:001000000
!:001000001
S:001000010
K:001000011000
3:0010000110010
2:0010000110011
(:001000011010
):001000011011
D:0010000111
;:00100010
Y:00100011000
&:0010001100010000000
%:0010001100010000001
$:0010001100100000 1
#:0010001100010000100
@:0010001100010000101
[:0010001100100011
]:0010001100100100
':0010001100100101
":0010001100100011
9:0010001100100 1
V:0010001100101
Q:001000110011
L:0010001101
-:001000111
b:001001
k:0010100
I:00101010
z:0010101100
?:0010101101
P:0010101110
j:0010101111
p:001011
s:0011
h:0100
i:0101
n:0110
o:0111
A:100000000
6:10000000100000
Z:1000000010000100
X:1000000010000101
/:100000001000011
*:1000000010001
1:100000001001
O:10000000101
E:1000000011
T:100000010
q:1000000110
R:10000001110
0:1000000111100
7:10000001111010
4:10000001111011
```