

dai16020

## Γεώργιος Φασίδης

Για την υλοποίηση της πρώτης εργασίας σχετικά με τα ντετερμινιστικά αυτόματα έγινε χρήση της Java και του Eclipse IDE. Η υλοποίηση περιέχει μόνο την λύση ντετερμινιστικών αυτομάτων και όχι των μη-ντετερμινιστικών με ε-μεταβάσεις.

Αρχικά ο πηγαίος κώδικας μου αποτελείται από 2 Java class αρχεία. Το πρώτο με την ονομασία Main.java περιέχει όλες τις κύριες λειτουργίες του προγράμματος μου ενώ η δεύτερη κλάση Transition.java λειτουργεί βοηθητικά πέραν του constructor που περιέχει προκειμένου να έχω τις μεταβιβάσεις «πιο συμμαζεμένες» σε μια δικιά τους δομή.

Στην Main class αρχικά υπάρχουν οι περισσότερες δηλώσεις μεταβλητών καθώς και δομών που θα χρησιμοποιηθούν, παρακάτω υπάρχουν μερικές ακόμα δηλώσεις, τόσο για την άντληση των πληροφοριών από τα text αρχεία όσο και στην λύση του αυτομάτου. Γίνεται χρήση κυρίως μεταβλητών Integer, String, Character καθώς και δομών όπως ArrayList και Array. Έχει χρησιμοποιηθεί κατάλληλη ονομασία για να είναι εύκολη η κατανόηση των μεταβλητών από οποιονδήποτε.

Στην συνέχεια υπάρχει ο κώδικας ο οποίος ζητάει από τον χρήστη το path του αρχείου που υπάρχει η περιγραφή του αυτομάτου προκειμένου να γίνει η ανάγνωση του και να αντληθούν όλα τα δεδομένα που χρειάζεται το πρόγραμμα για να λειτουργήσει. (ΠΡΟΣΟΧΗ η δομή του αρχείου είναι ίδια με την δομή που υπάρχει στις οδηγίες της άσκησης) [Για τον έλεγχο του προγράμματος παρέχονται 5 αρχεία αυτομάτων στον φάκελο της εργασίας με όνομα DFA Files και οι λέξεις στον φάκελο words files που αντιστοιχούν στα αυτόματα]

Για την ανάγνωση χρησιμοποιείται η BufferedReader και συνάρτηση readLine() η οποία επιστρέφει String και γι' αυτό υπάρχει η μετατροπή τους σε Integer με την

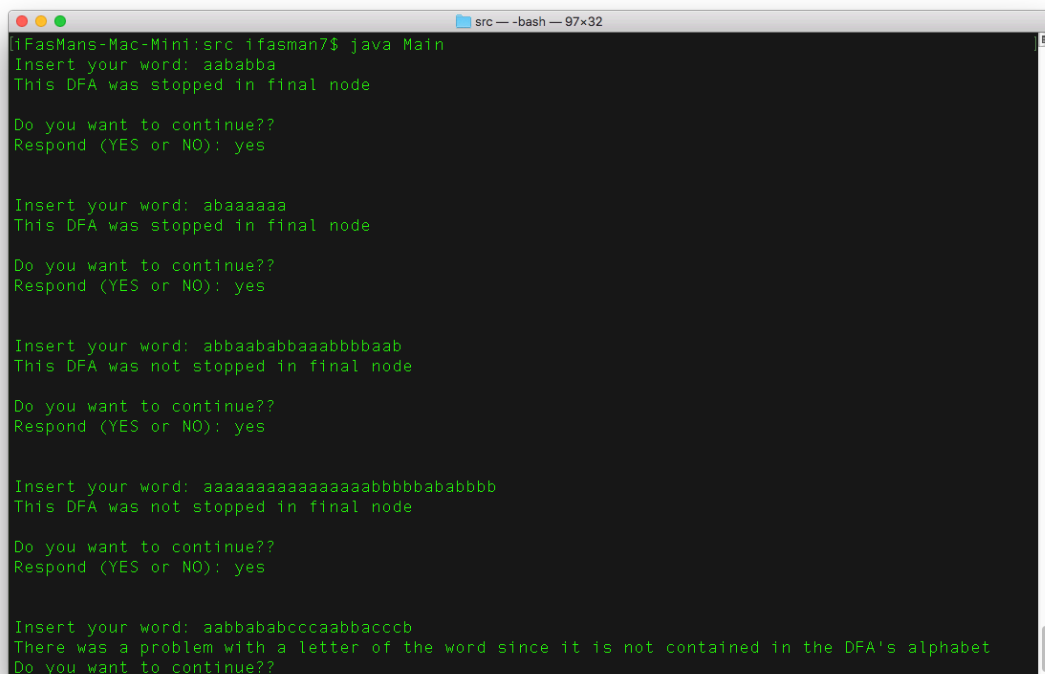
εντολή `Integer.valueOf` . Στις περιπτώσεις που υπάρχουν κενά ανάμεσα στους χαρακτήρες (κυρίως νούμερα αλλά και γράμματα) γίνεται η χρήση της συνάρτησης `split(" ")` μετά την κλήση της `readLine` προκειμένου να εξαλειφθούν όλα τα κενά. Αφού γίνει αυτό προκειμένου να είναι πιο εύκολη η ανάγνωση των χαρακτήρων χωρίς να υπάρξει κάποιο θέμα τα αποθηκεύω σε έναν πίνακα τύπου `String`, δηλαδή το κάθε γράμμα που ήταν χωρισμένο με κενά πάει σε μία θέση του πίνακα. Στις `for` παρακάτω γίνεται αποθήκευση στην αντίστοιχη λίστα των δεδομένων.

Παρακάτω ξεκινάει το `while loop` για την εισαγωγή των λέξεων από τον χρήστη μέσω πληκτρολογίου και τον έλεγχο τους από το πρόγραμμα. Αφού δοθεί η λέξη το πρόγραμμα θέτει σαν τρέχοντα κόμβο τον αρχικό και έπειτα σπάει την λέξη σε έναν πίνακα χαρακτήρων για να είναι πιο εύκολη η ανάγνωση της λέξης από το αυτό. Έπειτα για τον κάθε χαρακτήρα γίνεται έλεγχος για το αν αυτός υπάρχει στο αλφάβητο, δημιουργεί το πρόγραμμα αυτόματα το αλφάβητο κατά την ανάγνωση των μεταβιβάσεων από το αρχείο κειμένου, αν υπάρχει βρίσκει τον συνδυασμό που αντιστοιχεί στον τωρινό κόμβο μαζί με τον χαρακτήρα και επιστρέφει τον επόμενο κόμβο που πρέπει να μεταβεί, αν δεν υπάρχει μετάβαση από κάποιον κόμβο σε άλλον με το συγκεκριμένο χαρακτήρα τότε επιστρέφει έναν αρνητικό αριθμο (-500) και τερματίζει ο έλεγχος της λέξης, με `break`, και γίνεται `true` μια μεταβλητή `noNextNode` που λειτουργεί σαν `flag`. Σε περίπτωση που δεν υπάρχει στο αλφάβητο υπάρχει μια `Boolean` μεταβλητή που λειτουργεί σαν `flag` η οποία γίνεται `true` και έπειτα γίνεται `break` το `for` διότι έχει απορριφθεί η λέξη. Από την στιγμή που τελειώσει είτε το `for` είτε γίνει `break` λόγο της δεύτερης και τρίτης περίπτωσης υπάρχει παρακάτω ένα τμήμα κώδικα `if... else if... else if` το οποίο εμφανίζει κατάλληλο μήνυμα στην οθόνη ανάλογα με το αν έχει τερματίσει η λέξη σε τελικό κόμβο ή όχι καθώς και το αν έχει προκύψει `error` με κάποιον χαρακτήρα που δεν υπάρχει στο αλφάβητο ή δεν υπάρχει μετάβαση για κάποιον χαρακτήρα από κάποιον κόμβο (σ.σ οι μεταβλητές `flag` απο πριν).

Στο τέλος της `Main` υπάρχει το τμήμα κώδικα που ρωτάει τον χρήστη αν θέλει να συνεχίσει με κάποια άλλη λέξη.

Στην κλάση Transition υπάρχει ο constructor για τις private μεταβλητές που ορίζουν την μετάβαση από έναν κόμβο σε έναν άλλον, το αλφάβητο καθώς και οι βασικές get μέθοδοι των μεταβλητών. Την κλάση αυτή ολοκληρώνουν 3 static συναρτήσεις όπου η πρώτη, addLetterInAlphabet δημιουργεί το αλφάβητο του αυτομάτου. Η δεύτερη με ονομασία letterInAlphabet, ελέγχει αν το γράμμα υπάρχει στο αλφάβητο (γίνεται κλήση της συνάρτησης από την Main όπως αναφέρθηκε και πιο πριν) και η τρίτη, findNextNode, επιστρέφει τον επόμενο κόμβο στην μετάβαση.

Η παρακάτω εικόνα είναι από έκδοχή του προγράμματος που δεν ζητάει το path του αρχείου για προσωπική μου ευκολία στις δοκιμές.



```
src — -bash — 97x32
iFasMans-Mac-Mini:src ifasman7$ java Main
Insert your word: aababba
This DFA was stopped in final node

Do you want to continue??
Respond (YES or NO): yes

Insert your word: abaaaaaa
This DFA was stopped in final node

Do you want to continue??
Respond (YES or NO): yes

Insert your word: abbaababbaaabbbaab
This DFA was not stopped in final node

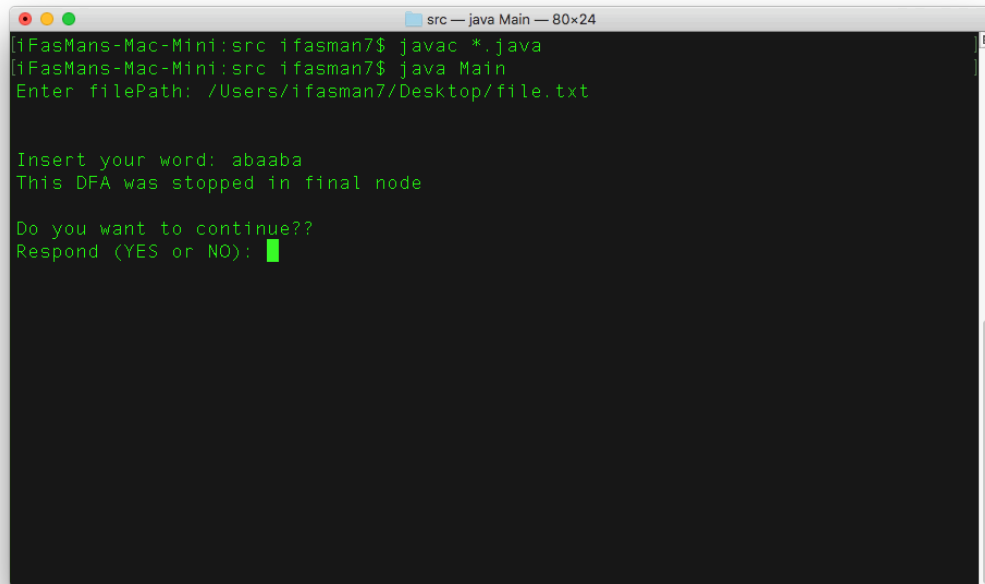
Do you want to continue??
Respond (YES or NO): yes

Insert your word: aaaaaaaaaaaaaabbbbababbbb
This DFA was not stopped in final node

Do you want to continue??
Respond (YES or NO): yes

Insert your word: aabbababcccaabbaccb
There was a problem with a letter of the word since it is not contained in the DFA's alphabet
Do you want to continue??
```

Σε αυτό το screenshot φαίνεται το τελικό jar αρχείο / εκτελέσιμο όπου ζητάει το path του αρχείου προκειμένου να λειτουργήσει .

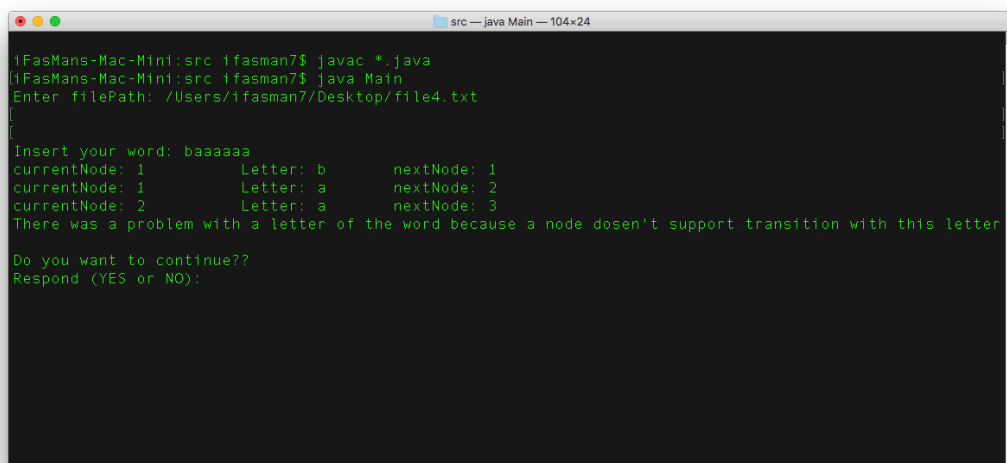


```
src — java Main — 80x24
iFasMans-Mac-Mini:src ifasman7$ javac *.java
iFasMans-Mac-Mini:src ifasman7$ java Main
Enter filePath: /Users/ifa7man7/Desktop/file.txt

Insert your word: abaaba
This DFA was stopped in final node

Do you want to continue??
Respond (YES or NO):
```

Σε αυτό το screenshot φαίνεται και η περίπτωση όπου ο κόμβος δεν υποστηρίζει την μετάβαση για κάποιον χαρακτήρα (DFA Files, αρχείο file4.txt) (τα print στην συγκεκριμένη εικόνα υπάρχουν για δικιά μου βοήθεια κατά τον έλεγχο. Δεν υπάρχουν στο τελικό εκτελέσιμο)

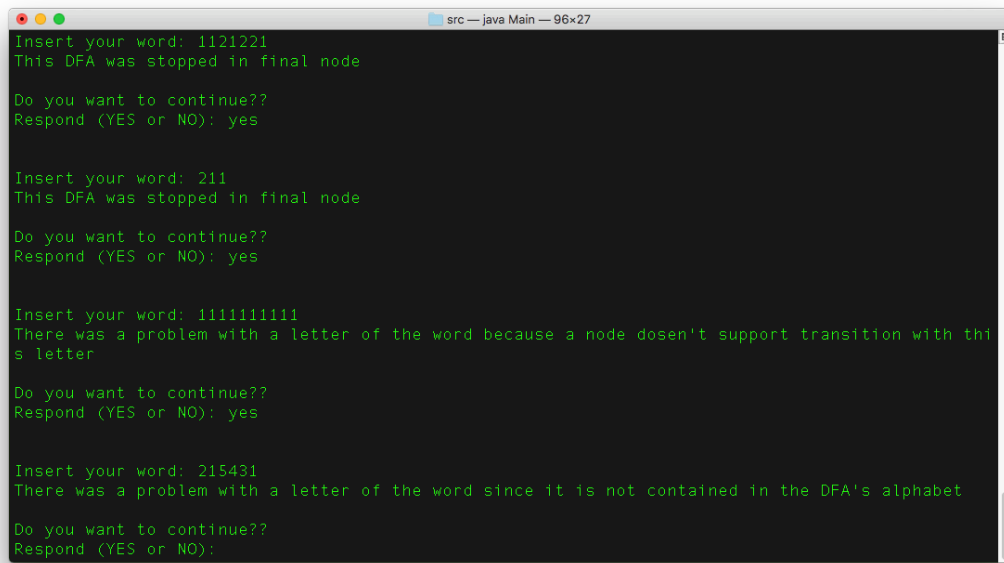


```
src — java Main — 104x24
iFasMans-Mac-Mini:src ifasman7$ javac *.java
iFasMans-Mac-Mini:src ifasman7$ java Main
Enter filePath: /Users/ifa7man7/Desktop/file4.txt

Insert your word: baaaaaa
currentNode: 1 Letter: b nextNode: 1
currentNode: 1 Letter: a nextNode: 2
currentNode: 2 Letter: a nextNode: 3
There was a problem with a letter of the word because a node dosen't support transition with this letter

Do you want to continue??
Respond (YES or NO):
```

Το πρόγραμμα μπορεί να διαβάσει οποιοδήποτε αυτόματο



```
src — java Main — 96x27
Insert your word: 1121221
This DFA was stopped in final node

Do you want to continue??
Respond (YES or NO): yes

Insert your word: 211
This DFA was stopped in final node

Do you want to continue??
Respond (YES or NO): yes

Insert your word: 111111111
There was a problem with a letter of the word because a node dosen't support transition with this letter

Do you want to continue??
Respond (YES or NO): yes

Insert your word: 215431
There was a problem with a letter of the word since it is not contained in the DFA's alphabet

Do you want to continue??
Respond (YES or NO):
```