ΜΕΡΟΣ Α-3
Συμπλήρωση πρότυπου κώδικα FLEX

Μέλη Ομάδας :

Τμήμα Β2

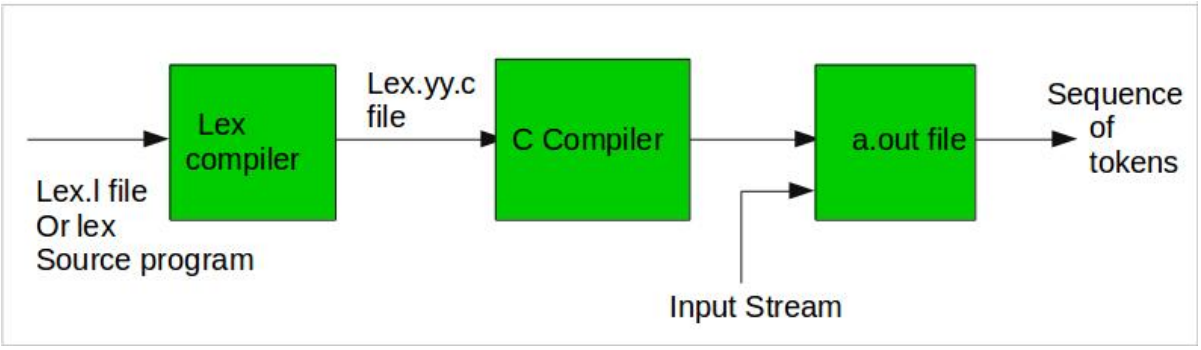
Ομάδα 6

1. ΗΛΙΑΣ ΔΗΜΑΣ (71347267)
2. ΝΙΚΟΛΑΟΣ ΠΑΠΠΑΣ (47422)
3. ΡΟΜΑΝΙΟΥΚ ΒΙΚΤΩΡ (713242017024)
4. ΤΖΩΡΤΖΗΣ ΑΓΓΕΛΟΣ (18390094)

Περιεχόμενα

Τι είναι το FLEX;

Flex is a tool for generating scanners. A scanner is a program which recognizes lexical patterns in text. The flex program reads the given input files, or its standard input if no file names are given, for a description of a scanner to generate.



Some Flex Pattern Matches

[0-9]	all the digits between 0 and 9
[0+9]	either 0, + or 9
[-09]	either -, 0 or 9
a{2, 4}	either aa, aaa or aaaa
a*	0 or more occurrences of a
.	any character except newline
w(x | y)z	wxz or wyz

Αρχείο εισόδου

```
"\"
"""
 "      "
  " \\ \n \""
"Hello"
; INTS
              +5
-10
011
45
68
-43
+98
++45
--45
+78
+23
90



123
87
12
65
+8
-76
---456
```

; FLOATS

45.6

23.6

0.5

+43

3e34

0.45

--3.14

3.14

+0.004

54.5e-11

0.9

++32.4

34.7

87.0e

0e0

e10

45.67

; VARS

?ewfwe

?tyrtje

?567536hsfrtgaer

?45643

?7irtyt

?75iuyie

?^74435t64%%%$&

?2345wgdgfshgJBUWYEFBUY

?4534g+__345435

?|}GERGERGw

?345254$$$$

?varws21

?|#@$%%

?~!XS

?123&&#YTRHW

?235gdgeqrtgerq

?41234ffwfd

?ABC

?443

?()()()()()

; STRINGS

"Hello"

"cOMPILERS 432424 \N"

"test\n\""

"Str ing"

""

" "

"\\\\\\\""

"name"

"erga s i s \t"

"String" + "Concat"

"afasdfasdfasfasf"

"afdasfasf\\n"

"""""""""""""

; NAMES

variable-1

disdiastatos-pinakas

tatic\facts

Ammount

TAVLI12

v?567536_hsfrtgaer

werftw

f c eg

X14123-5refve

JKniub

ggfd2435346

3456ERGTDBFGJ?>?ytu

VCBRTEERTY%$#$%#RE

234-asd_VSDFV?ytry

342534234fsdsdfsd

5345egfd

df

rf

KJIBUY

Κανονικές εκφράσεις FLEX

DELIMITER	[\t]+
INTEGERS	[\s \t]*(0|[+-]?[1-9]+[0-9]*)[\s \t]*
FLOATS	[\s \t]*[-+]?(0|[1-9]+[0-9]*)([.][0-9]+([eE][+-](0 |[1-9]+[0-9]*))?|[eE] [+-]?(0|[1-9]+[0-9]*))[\s \t]*
NAMES	[\s \t]*[a-zA-Z]+((-|_)*[0-9]*[a-zA-Z]*)*[\s \t]*
VARIABLES	[\s \t]*\?([0-9]*[a-zA-Z]*)+[\s \t]*
STRINGS	[\s \t]*\"(\\[\\\"n]|[^\"\\\n])*\"[\s \t]*
COMMENTS	;.*\n
BREAKS	[\s]+
UNKNOWN	^.*

Αρχείο εξόδου

Line=1 Token=UN Value=@"\" @

Line=2 Token=UN Value=@ """@

Οι πρώτες δύο γραμμές είναι unknown, όπως βλέπουμε δεν μπορεί να είναι κάποια λεκτική μονάδα το συγκεκριμένο παράδειγμα καθώς έχουμε ορίσει πώς στα strings οι χαρακτήρες " και \ χρειάζονται \ από μπροστά για να είναι δεκτοί.

Line=3 Token=STRINGS Value=@ " "@

Line=4 Token=STRINGS Value=@ " \\ \n \""@

Line=5 Token=STRINGS Value=@"Hello"@

Εδω βλέπουμε καθαρά οτι για είσοδο έχουμε 3 strings. Το FLEX επιστρέφει σωστό αποτέλεσμα.

Line=7 Token=INTS Value=@ +5 @

Line=8 Token=INTS Value=@-10@

2 προσημασμένοι ακέραιοι. Παρόλο που υπάρχουν κενά πριν και μετά το FLEX αντιλαμβάνεται οτι αυτά είναι delimiter και τα περνάει ως INTS

Line=9 Token=UN Value=@011@

Η γραμματική μας δεν επιτρέπει να έχουμε τέτοιου είδους αριθμό (02232, 01, 0325 κ.α), οπότε είναι unknown

Line=10 Token=INTS Value=@45@

Line=11 Token=INTS Value=@68@

Line=12 Token=INTS Value=@-43@

Line=13 Token=INTS Value=@+98@

Βλέπουμε καθαρά οτι για είσοδο έχουμε ακέραιους αριθμούς με πρόσημο και χωρίς. Το αποτέλεσμα είναι ορθό

Line=14 Token=UN Value=@++45@

Line=15 Token=UN Value=@--45@

Επιτρέπεται να έχουμε μόνο ένα πρόσημο, μια και μόνο φορά. Οπότε το αποτέλεσμα είναι unknown

Line=16 Token=INTS Value=@+78@

Line=17 Token=INTS Value=@+23@

Line=18 Token=INTS Value=@90@

Line=19 Token=INTS Value=@123@

Line=20 Token=INTS Value=@87@

Line=21 Token=INTS Value=@12@

Line=22 Token=INTS Value=@65@

Line=23 Token=INTS Value=@+8@

Line=24 Token=INTS Value=@-76@

Παραπάνω βλέπουμε και άλλα παραδείγματα με σωστό input για integers

Line=25 Token=UN Value=@---456@

Παρόμοιο παράδειγμα με πολλαπλό πρόσημο. Είναι unknown

Line=28 Token=FLOATS Value=@45.6@

Line=29 Token=FLOATS Value=@23.6@

Line=30 Token=FLOATS Value=@0.5@

Παράδειγμα για floats, χωρίς πρόσημο. Βλεπουμε οτι το FLEX τα διαβάζει σωστα

Line=31 Token=INTS Value=@+43@

Line=32 Token=FLOATS Value=@3e34@

Float που υπάρχει e. Το FLEX επίσης το αντιλαμβάνεται και μας επιστρέφει σωστή τιμή

Line=33 Token=FLOATS Value=@0.45@

Line=34 Token=UN Value=@--3.14@

Εσφαλμένη εισοδο: Πολλαπλά πρόσημα.

Line=35 Token=FLOATS Value=@3.14@

Line=36 Token=FLOATS Value=@+0.004@

Line=37 Token=FLOATS Value=@54.5e-11@

Line=38 Token=FLOATS Value=@0.9@

Floats με πρόσημο και e. Επιτρεπτές εισόδοι σύμφωνα με την γραμματική μας. Λαμβάνουμε αποτέλεσμα Float

Line=39 Token=UN Value=@++32.4@

Line=40 Token=FLOATS Value=@34.7@

Line=41 Token=UN Value=@87.0e@

Unknown επειδή μετά το e δεν ακολουθεί κάποιος αριθμός ώστε να πραγματοποιηθεί η δύναμη. Αποτέλεσμα unknown

Line=42 Token=FLOATS Value=@0e0@

Line=43 Token=NAMES Value=@e10@

Σε αυτήν την περίπτωση το αποτέλεσμα μας είναι NAMES, επειδή η γραμματική λεει οτι τα ονόματα ορισμών και στοιχείων γεγονότων ξεκινούν από λατινικό χαρακτήρα, οπότε στην περίπτωσή μας που το input είναι το e10 το διαβάζει ως NAMES

Line=44 Token=FLOATS Value=@45.67@

Line=47 Token=VARS Value=@?ewfwe@

Line=48 Token=VARS Value=@?tyrtje@

Line=49 Token=VARS Value=@?567536hsfrtgaer@

Line=50 Token=VARS Value=@?45643@

Line=51 Token=VARS Value=@?7irtyt@

Line=52 Token=VARS Value=@?75iuyie@

Βλέπουμε ότι οι παραπάνω είσοδοι ξεκινούν από ? και περιέχουν τους επιτρεπτούς χαρακτήρες . Αυτό σημαίνει ότι είναι ονόματα μεταβλητών. Το FLEX το αντιλαμβάνεται σωστά και επιστρέφει VARS

Line=53 Token=UN Value=@?^74435t64%%%$&@

Οι μεταβλητές περιέχουν οποιοδήποτε γράμμα ή αριθμό αλλά και μη αποδεκτούς χαρακτήρες, γι αυτό βλέπουμε Unknown

Line=54 Token=VARS Value=@?2345wgdgfshgJBUWYEFBUY@

Line=55 Token=UN Value=@?4534g+__345435@

Line=56 Token=UN Value=@?|}GERGERGw@

Line=57 Token=UN Value=@?345254$$$$@

Line=58 Token=VARS Value=@?varws21@

Line=59 Token=UN Value=@?|#@$%%@

Line=60 Token=UN Value=@?~!XS@

Line=61 Token=UN Value=@?123&&#YTRHW@

Οι μεταβλητές περιέχουν μη αποδεκτούς χαρακτήρες, γι αυτό βλέπουμε Unknown

Line=62 Token=VARS Value=@?235gdgeqrtgerq@

Line=63 Token=VARS Value=@?41234ffwfd@

Line=64 Token=VARS Value=@?ABC@

Line=65 Token=VARS Value=@?443@

Line=66 Token=UN Value=@?()()()()()@

Line=69 Token=STRINGS Value=@"Hello"@

Line=70 Token=UN Value=@"cOMPILERS 432424 \N"@

Δεν είναι δεκτό το string από πάνω καθώς ο χαρακτήρας διαφυγής είναι \n και όχι \N

Απο την γραμματική ξέρουμε ότι τα strings ξεκινάνε με " και τελειώνουν με " και μέσα μπορούν να περιέχουν οτιδήποτε με κάποιες εξαιρέσεις. Από κάτω βλέπουμε ότι το FLEX το αντιλαμβάνεται σωστά και επιστρέφει οτι η εισοδός μας είναι τύπου STRINGS

Line=71 Token=STRINGS Value=@"test\n\""@

Line=72 Token=STRINGS Value=@"Str ing"@

Line=73 Token=STRINGS Value=@""@

Line=74 Token=STRINGS Value=@" "@

Line=75 Token=STRINGS Value=@"\\\\\\\""@

Line=76 Token=STRINGS Value=@"name"@

Line=77 Token=UN Value=@"erga s i s \t"@

Η γραμματική μας επιτρέπει να χρησιμοποιήσουμε μόνο \n , \\ και \". Οπότε το output είναι Unknown στο παραπάνω παράδειγμα.

Line=78 Token=UN Value=@"String" + "Concat"@

Δεν έχουμε ορίσει πως και αν θα γίνεται το string concatenation, οπότε αποτέλεσμα Unknown

Line=79 Token=STRINGS Value=@"afasdfasdfasfasf"@

Line=80 Token=STRINGS Value=@"afdasfasf\\n"@

Ιδιαίτερη περίπτωση καθώς βλέπουμε \\n μέσα στο string. Αλλα διαβάζονται πρώτα οι 2 κάθετοι που είναι αποδεκτό και μετά το n που είναι επίσης αποδεκτό οπότε δεν προκύπτει κάποιο πρόβλημα.

Line=81 Token=UN Value=@"""""""""""""@

Όπως είπαμε τα string ξεκινάνε με " και τελειώνουν με ". Αν θέλουμε να χρησιμοποιήσουμε το " μέσα πρέπει να το γράψουμε έτσι \". Άρα αποτέλεσμα unknown

Line=83 Token=NAMES Value=@variable-1@

Line=84 Token=NAMES Value=@disdiastatos-pinakas@

Σε αυτήν την περίπτωση το αποτέλεσμα μας είναι NAMES, επειδή η γραμματική λεει οτι τα ονόματα ορισμών και στοιχείων γεγονότων ξεκινούν από λατινικό χαρακτήρα

Line=85 Token=UN Value=@tatic\facts@

Line=86 Token=UN Value=@Ammount\@

Line=87 Token=NAMES Value=@TAVLI12@

Line=90 Token=UN Value=@v?567536_hsfrtgaer@

Τα NAMES μπορούν να περιέχουν μονο - _ απο σύμβολα. Για αυτο στις παραπάνω περιπτώσεις έχουμε Unknown

Line=92 Token=NAMES Value=@werftw@

Line=93 Token=UN Value=@f c eg@

Line=94 Token=NAMES Value=@X14123-5refve@

Line=95 Token=NAMES Value=@ JKniub@

Line=96 Token=UN Value=@ @

Line=97 Token=NAMES Value=@ggfd2435346@

Line=98 Token=UN Value=@3456ERGTDBFGJ?>?ytu@

Line=99 Token=UN Value=@VCBRTEERTY%$#$%#RE@

Line=100 Token=UN Value=@234-asd_VSDFV?ytry@

Line=101 Token=UN Value=@342534234fsdsdfsd@

Line=102 Token=UN Value=@5345egfd@

Line=103 Token=NAMES Value=@df@

Line=104 Token=NAMES Value=@rf@

Line=105 Token=NAMES Value=@KJIBUY@

Προβλήματα που προέκυψαν και πώς αντιμετωπίστηκαν:

Αρχικά ένα πρόβλημα που έπρεπε να επιλυθεί για όλες τις κανονικές εκφράσεις ήταν να αγνοούνται τα white spaces. Για την επίλυση πρίν και μετά από κάθε κανονική έκφραση τοποθετήσαμε αυτόν εδώ το κομμάτι [\s \t]* πρίν και μετά από τις κανονικές μας εκφράσεις που επιτρέπει στο πρόγραμμα να διαβάζει ένα token που θα έχει white spaces πρίν και μετά από αυτό. Π.χ.: [KENO][KENO]5[TAB][KENO][TAB] θα διαβάζεται ώς ο αριθμός 5 χωρίς κάποιο σφάλμα. Επίσης αντιμετωπίσαμε πρόβλημα στην κανονική έκφραση για τα ονόματα ορισμών και στοιχείων γεγονότων, καθώς είχαμε βάλει [A-z] αντί για [A-Z] και μας προέκυπταν λάθος αποτελέσματα στην έξοδο αφού είχαμε βάλει λάθος εύρος τιμών. Το πρόβλημα αυτό ευτυχώς εντοπίστηκε και αντιμετωπίστηκε εύκολα. Τέλος είχε προκύψει και ένα θέμα με τους αριθμούς κινητής υποδιαστολής καθώς διάβαζε τους αριθμούς που ξεκινούσαν με μηδέν και ήταν ακέραιοι ώς FLOATS (π.χ.: 003, 05, 0012). Λύθηκε το πρόβλημα αντικαθιστώντας την παλιά κανονική έκφραση με την καινούργια που υπάρχει σε αυτό το έγγραφο και την βρήκαμε μέσω δοκιμών και χρησιμοποιώντας το regexpal.

Συμπέρασμα:

Έπειτα από αρκετές προσπάθειες και εμπόδια που αντιμετωπίσαμε στην συγκεκριμένη άσκηση, καταφέραμε να την επιλύσουμε. Αρχικά συμπληρώσαμε τα <<FILL ME>> κενά που μας είχαν ανατεθεί σε διαφορετικά σημεία του κώδικα. Έπειτα , η πρώτη απόπειρα για την συμπλήρωση των regex εκφράσεων , ώστε να λειτουργεί 100% ο κώδικας μας απέτυχε. Μετά από αρκετό ψάξιμο από τα μέλη της ομάδας , διορθώθηκαν αρκετές εκφράσεις regex , οι οποίες ήταν εξαρχής σε ορισμένα σημεία λάθος, και με αυτήν την διόρθωση καταφέραμε να φέρουμε τον κώδικα σε πλήρη λειτουργία. Παρόλαυτα ο κώδικας μας δεν λειτούργησε αμέσως όπως θα έπρεπε. Χρειάστηκε να συμπληρώσουμε κομμάτι κώδικα , ώστε να πετύχουμε το επιθυμητό αποτέλεσμα , το οποίο έπειτα από ατομικές αλλά και ομαδικές προσπάθειες , το καταφέραμε όπως θα δείτε και παραπάνω αλλά και στο επισυναπτόμενο αρχείο. Όσον αφορά την συνεργασία της ομάδας , για άλλη μια φορά κύλησε ομαλά, με τον κάθε συμμετέχοντα να αναλαμβάνει από ένα τμήμα της εργασίας και στο τέλος , να γίνεται ολική επανάληψη ολόκληρης της εργασίας , ώστε να καταλάβουν όλα τα μέλη , όλα τα μέρη της εργασίας.

Αρμοδιότητες των μελών της ομάδας:

ΗΛΙΑΣ ΔΗΜΑΣ: Συγγραφή κώδικα και δοκιμές για την είσοδο/έξοδο του κώδικα μας.

ΝΙΚΟΛΑΟΣ ΠΑΠΠΑΣ: Συγγραφή εγγράφου τεκμηρίωσης και δοκιμές για την είσοδο/έξοδο του κώδικα μας.

ΡΟΜΑΝΙΟΥΚ ΒΙΚΤΩΡ: Συγγραφή εγγράφου τεκμηρίωσης και δοκιμές για την είσοδο/έξοδο του κώδικα μας.

ΤΖΩΡΤΖΗΣ ΑΓΓΕΛΟΣ: Συγγραφή κώδικα και συγγραφή εγγράφου τεκμηρίωσης.