ATHENS UNIVERSITY OF APPLIED SCIENCES

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΑΘΗΝΑΣ

ATHENS UNIVERSITY OF APPLIED SCIENCES

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΡΙΚΗΣ

ΜΕΤΑΓΛΩΤΤΙΣΤΕΣ

ΠΑΡΙΣ ΛΙΖΑΙ

Πίνακας περιεχομένων

- ✓ Η εργασία αυτή αποσκοπεί στην παρουσίαση των κανονικών εκφράσεων, των αυτομάτων και ενός λεκτικού αναλυτή σε γλώσσα C. Παρακάτω αναλύονται διεξοδικά :
  - οι ακέραιοι,
  - οι εκθετικοί δεκαδικοί,
  - οι μεταβλητές,
  - τα σχόλια
  - οι συμβολοσειρές.

ΚΕΦΑΛΑΙΟ 1

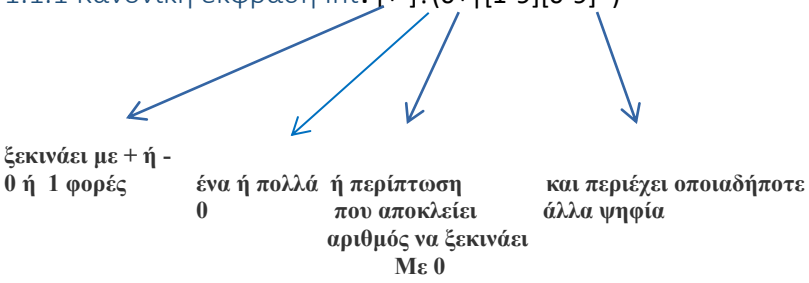
1.1 ΑΚΕΡΑΙΟΙ

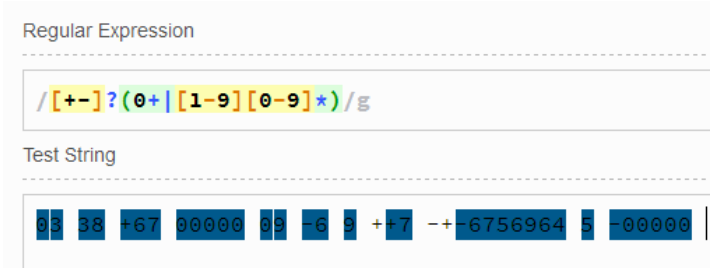
1.1.0 Περιγραφή

Ένα ακέραιος αποτελείται από ένα ή περισσότερα ψηφία 0-9. Ένας ακέραιος με μήκος 2 ή μεγαλύτερο δεν μπορεί να αρχίζει από 0.

- ➢ Παραδείγματα ακέραιων είναι τα 0, 2, +34, -1000 κλπ.

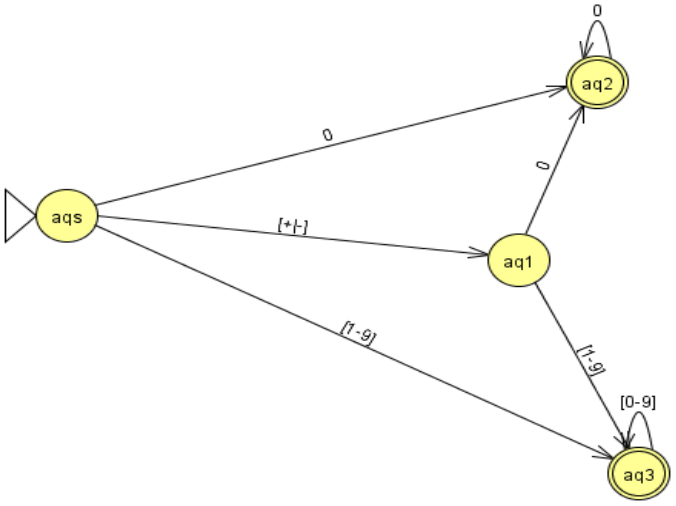
1.1.1 Κανονική έκφραση Int: [+-]?(0+|[1-9][0-9]*)





Σωστά: 0 , 3 +67, 00000, 9, -9
Λάθος: 09, +, -+

1.1.2Αυτόματο και FSM:



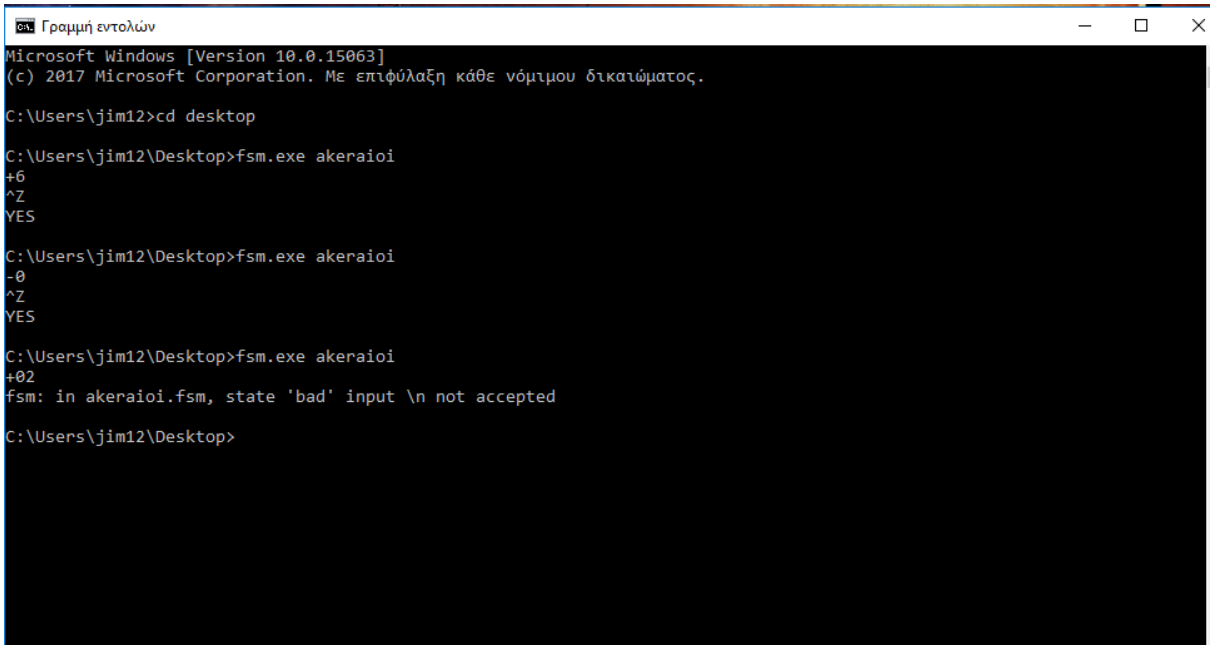
FSM:

```
START=AQS
AQS:0 -> AQ2
       - + -> AQ1
      1-9 -> AQ3
AQ2:0 -> AQ2
      \n ->GOOD
      - + ->BAD
      1-9 ->BAD
AQ1:0 ->AQ2
       1-9 -> AQ3
      - + ->BAD
AQ3:1-9 -> AQ3
      0->AQ3
      - + -> BAD
      \n->GOOD
GOOD(OK):
```

1.1.3 Πίνακας μεταβάσεων:

Πίνακας Μεταβάσεων για τους ακεραίους (κανονική έκφραση: [+-]?(0+|[1-9][0-9]*)			
	+|-	0	[1-9]
aqs	aq1	aq2	aq3
aq1	error	aq2	aq3
aq2	error	aq2	error
aq3	error	aq3	aq3

1.1.4 Εκτελέσεις:



1.2 ΕΚΘΕΤΙΚΟΙ

1.2.0 Περιγραφή

Ένας αριθμός κινητής υποδιαστολής αποτελείται από ένα ακέραιο μέρος (ακέραιος αριθμός) και ένα δεκαδικό μέρος (ακολουθία από ψηφία 0-9) που διαχωρίζονται από μια τελεία. Ένα από τα δυο μέρη μπορεί να λείπει αλλά όχι και τα δυο μαζί.

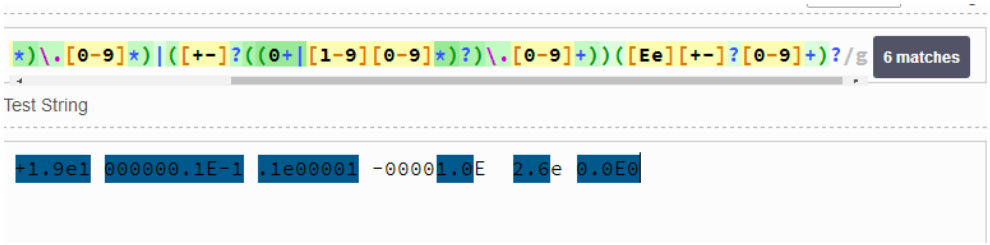
- Παραδείγματα αριθμών κινητής υποδιαστολής (ή floats για συντομία) είναι οι 2.0, 1.23, 100., +.0123 και -52.3E-4. Το σύμβολο E δηλώνει δύναμη του 10. Σε αυτή την περίπτωση, το -52.3E-4 σημαίνει -52.3 * 10-4 και αρμόδιος να το γνωρίζει και να το διαχειριστεί είναι ο σημασιολογικός αναλυτής του μεταγλωττιστή.

1.2.1Κανονική Έκφραση Float:

([+-]?(0+|[1-9][0-9]*)\.[0-9]*|[+-]?\.[0-9]+)([Ee][+-]?[0-9]+)?

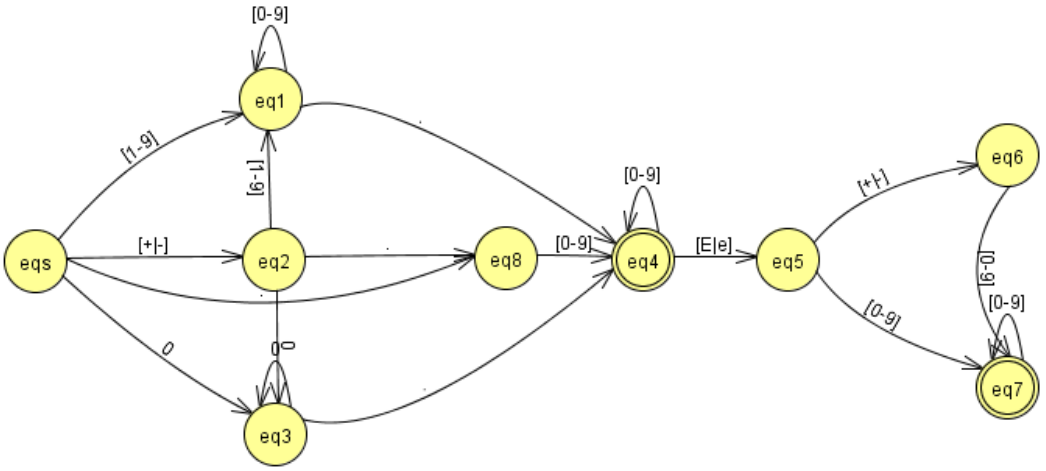
Αναζήτηση για δεκαδικό

Αναζήτηση για E ή e (εάν υπάρχει), μπορεί να ακολουθείται από πρόσημο και ένα ή παραπάνω ψηφία.



Σωστά: +1.1, -21.34, 1.9e-2. 1.9E9
Λάθος: .e-2 , 2. , -2. , +.E-5

1.2.2 Αυτόματο και FSM:



FSM: *Ο κώδικας του αρχείου λόγο μεγέθους βρίσκεται στο αρχείο ek8.fsm*

1.2.3 Πίνακας μετάβασης:

Πίνακας Μεταβάσεων για τους εκθετικούς δεκαδικούς
κανονική έκφραση: (([+-]?(0+|[1-9][0-9]*)\.[0-9]*)|([+-]?\.[0-9]+))([Ee][+-]?[0-9]+)?

	+|-	0	[1-9]	.	E|e
eqs	eq2	eq3	eq1	eq8	error
eq1	error	eq1	eq1	eq4	error
eq2	error	eq3	eq1	eq8	error
eq3	error	eq3	error	eq4	error
eq4	error	eq4	eq4	error	eq5
eq5	eq6	eq7	eq7	error	error
eq6	error	eq7	eq7	error	error
eq7	error	eq7	eq7	error	error
eq8	error	eq4	eq4	error	error

1.2.4 Εκτελέσεις

```
C:\Users\Angela\Desktop>fsm.exe ek8
+0.45
^Z
YES

C:\Users\Angela\Desktop>fsm.exe ek8
+.3
^Z
YES

C:\Users\Angela\Desktop>fsm.exe ek8
3.
^Z
YES

C:\Users\Angela\Desktop>fsm.exe ek8
32.E4
^Z
YES

C:\Users\Angela\Desktop>fsm.exe ek8
-.E-4
fsm: in ek8.fsm, state 'bad' input - not accepted

C:\Users\Angela\Desktop>fsm.exe ek8
-.1E-4
^Z
YES

C:\Users\Angela\Desktop>fsm.exe ek8
+05.03
fsm: in ek8.fsm, state 'bad' input . not accepted
```

1.3 ΜΕΤΑΒΛΗΤΕΣ

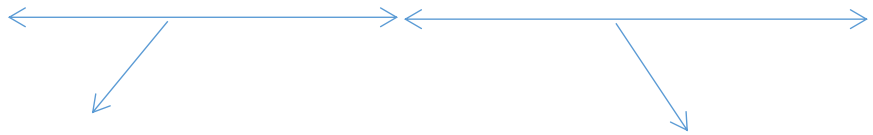
1.3.0 Περιγραφή

Οι μεταβλητές αποτελούν συμβολικά ονόματα θέσεων μνήμης. Το όνομα μιας μεταβλητής μπορεί να περιλαμβάνει λατινικούς χαρακτήρες a .. z, A .. Z αριθμούς 0.. 9 και _ Το όνομα μιας μεταβλητής δεν μπορεί να αρχίζει όμως με αριθμητικό ψηφίο. Στο όνομα ενός αναγνωριστικού διακρίνονται τα πεζά από τα κεφαλαία (case-sensitive). Μπορείτε να χρησιμοποιείτε ελληνικές λέξεις ως ονόματα αναγνωριστικών.

- Αποδεκτά ονόματα μεταβλητών: X, αβ, abcdef, A12345, X1Yssss, my_Name, ο_όνομά_μου, _abcde, a_bc_123, _12345, _12345_ .

1.3.1Κανονική Έκφραση Variable:

[a-zA-Z_A-Ωα-ωόίύέήάώ]([0-9_a-zA-ZA-Ωα-ωόίύέήάώ]*)



Οποιοδήποτε χαρακτήρας φορές

Οποιοσδήποτε χαρακτήρας από 0 έως πολλές

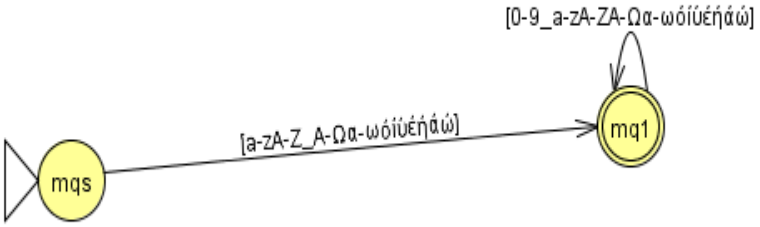
1 φορά εκτός από ψηφίο

```
/[a-zA-Z_A-Ωα-ωόίύέήάώ]([0-9_a-zA-ZA-Ωα-ωόίύέήάώ]*)/g
```

Test String

```
X, αβ, abcdef, A12345, X1Yssss, my_Name,
ο_όνομά_μου,_abcde, a_bc_123, _12345, _12345_
12u  m^%$
```

1.3.2Αυτόματο και FSM:



FSM:

```
START=MQS
MQS:    A-Z a-z _->MQ1
    A-Ω α-ω έ ύ ί ό ά ή ώ ς->MQ1
    0-9->BAD
MQ1:    0-9->MQ1
    A-Z a-z _->MQ1
    A-Ω α-ω έ ύ ί ό ά ή ώ ς->MQ1
    \n->GOOD
GOOD(OK):

```

1.3.3 Πίνακας μετάβασης:

Πίνακας Μεταβάσεων για μεταβλητές
κανονική έκφραση: [a-zA-Z_A-Ωα-ωόίύέήάώ]([0-9_a-zA-ZA-Ωα-ωόίύέήάώ]*)

	[_a-zA-ZA-Ωα-ωέύίόάήώ	[0-9]
mqs	mq1	error
mq1	mq1	mq1

1.3.4 Εκτελέσεις

Γραμμή εντολών

```
C:\Users\jim12\Desktop>fsm.exe metabl
x
^Z
YES

C:\Users\jim12\Desktop>fsm.exe metabl
as
^Z
YES

C:\Users\jim12\Desktop>fsm.exe metabl
Asdfggf64ADDW
^Z
YES

C:\Users\jim12\Desktop>fsm.exe metabl
212swad
fsm: in metabl.fsm, state 'bad' input 1 not accepted
```

1.4 ΣΧΟΛΙΑ

1.4.0 Περιγραφή

Τα σχόλια αρχίζουν με το σύμβολο # και συνεχίζουν μέχρι το τέλος της γραμμής.

- #This is a single line column

Αν επιθυμούμε σχόλια που να επεκτείνονται σε πολλές γραμμές θα πρέπει να τα περικλείσουμε μεταξύ δυο τριάδων διπλών εισαγωγικών """ (3 διπλά εισαγωγικά στην αρχή και το τέλος του σχολίου).

1.4.1Κανονική Έκφραση:

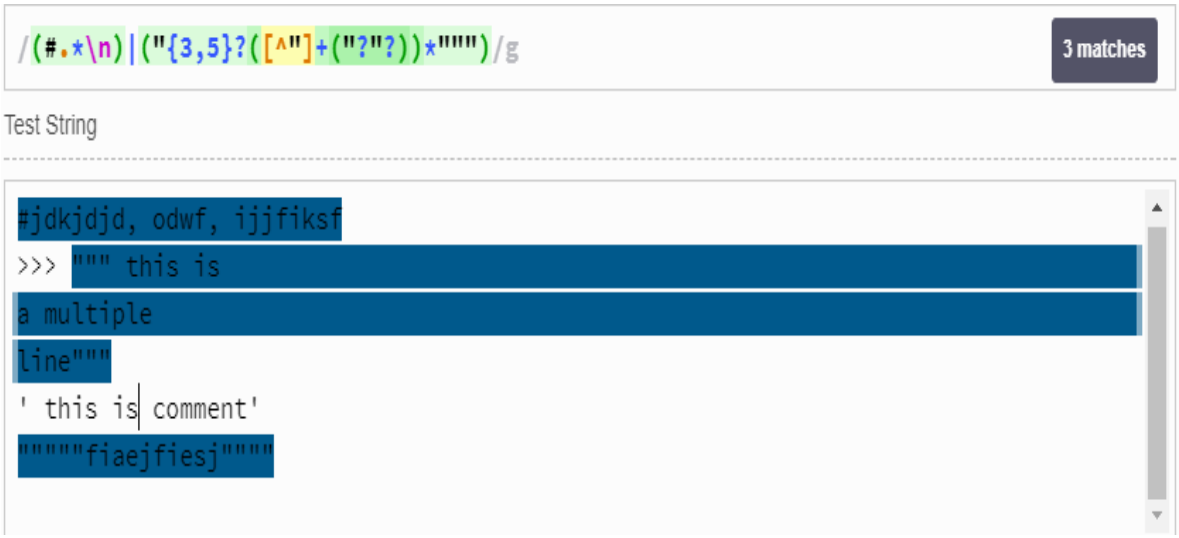
*ΠΑΡΑΤΗΡΗΣΗ(ΠΑΤΗΣΤΕ ΠΑΝΩ)

(#.*\n)|("{3,5}?([^"]+("?"?))*""")

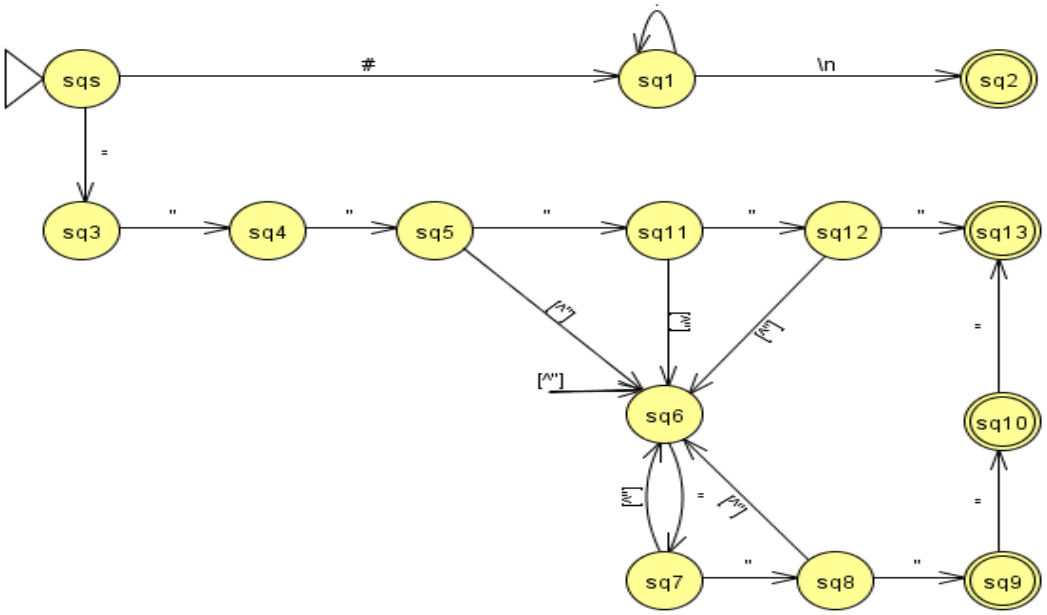
Ψάχνει τα λιγότερα που μπορεί να βρει

Ψάχνει να βρει τον χαρακτήρα (#.)

Το string μέσα στα """...""" που ορίζουν τα σχόλια μία φορά.



1.4.2 Αυτόματο και FSM:




```
1   START=SQS
2   SQS:        #->SQ1
3               \n->BAD
4               "->SQ3
5               *->BAD
6   SQ1:        #->SQ1
7               \n->GOOD
8               "->SQ1
9               *->SQ1
10  SQ2:        #->BAD
11              \n->GOOD
12              "->BAD
13              *->BAD
14  SQ3:        #->BAD
15              \n->BAD
16              "->SQ4
17              *->BAD
18  SQ4:        #->BAD
19              \n->BAD
20              "->SQ5
21              *->BAD
22  SQ5:        #->SQ6
23              \n->SQ6
24              "->SQ11
25              *->SQ6
26  SQ6:        #->SQ6
27              \n->SQ6
28              "->SQ7
29              *->SQ6
30  SQ7:        #->SQ6
31              \n->SQ6
32              "->SQ8
33              *->SQ6
34  SQ8:        #->SQ6
35              \n->SQ6
36              "->SQ9
37              *->SQ6
38  SQ9:        #->BAD
39              \n->GOOD
40              "->SQ10
41              *->BAD
42  SQ10:       #->BAD
43              \n->GOOD
44              "->SQ13
45              *->BAD
46  SQ11:       #->SQ6
47              \n->SQ6
48              "->SQ12
49              *->SQ6
50  SQ12:       #->SQ6
51              \n->SQ6
52              "->SQ13
53              *->SQ6
54  SQ13:       #->BAD
55              \n->GOOD
56              "->BAD
57              *->BAD
58  GOOD(OK):
59
```

1.4.3 Πίνακας μετάβασης:

Πίνακας μετάβασης για τα σχόλια
κανονική έκφραση: (#.*\n)|("{3,5}?([^"]+("?"?))*""")

	#	\n	"	anything else
sqs	sq1	error	sq3	error
sq1	sq1	sq2	sq1	sq1
sq2	error	error	error	error
sq3	error	error	sq4	error
sq4	error	error	sq5	error
sq5	sq6	sq6	sq11	sq6
sq6	sq6	sq6	sq7	sq6
sq7	sq6	sq6	sq8	sq6
sq8	sq6	sq6	sq9	sq6
sq9	error	error	sq10	error
sq10	error	error	sq13	error
sq11	sq6	sq6	sq12	sq6
sq12	sq6	sq6	sq13	sq6
sq13	error	error	error	error

1.4.4 Εκτελέσεις

```
C:\Users\Angela\Desktop>fsm.exe sxoliafsm
"""kl"""
^Z
YES

C:\Users\Angela\Desktop>fsm.exe sxoliafsm
""""""
^Z
YES

C:\Users\Angela\Desktop>fsm.exe sxoliafsm
""""""123""123""123"123""""""
^Z
YES

C:\Users\Angela\Desktop>fsm.exe sxoliafsm
"""123""
^Z
NO

C:\Users\Angela\Desktop>fsm.exe sxoliafsm
""""""""
fsm: in sxoliafsm.fsm, state 'bad' input " not accepted
```

1.5 ΤΕΛΕΣΤΕΣ

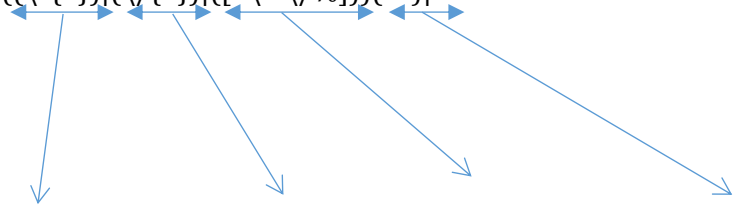
1.5.0 Περιγραφή

Οι αριθμητικοί τελεστές χρησιμοποιούνται για πράξεις μεταξύ δυο αριθμών (τελεστέων):

+ →Προσθήκη 2 τελεστέων ή συν ως πρόσημο x + y, +2
-→ Αφαίρεση δεύτερου τελεστέου από τον πρώτον ή μείον ως πρόσημο x – y, -2
*→ Πολλαπλασιασμός 2 τελεστέων x * y
/→ Διαίρεση αριστερού τελεστέου από τον δεξιό (πάντα δίνει αποτέλεσμα float) x / y
% → Υπόλοιπο – υπόλοιπο διαίρεσης αριστερού τελεστέου από τον δεξιό x % y (remainder of x/y)
// → Βάση διαίρεσης – ακέραιο μέρος διαίρεσης αριστερού τελεστέου από τον δεξιό x // y
** Εκθετικός – αριστερός τελεστέος υψωμένος στη δεξιά δύναμη x**y (x to the power y)

1.5.1 Κανονική Έκφραση Operator:

(([*{2}]|([\/{2}]|([+\-*\/%]))(=?)|=



Τα σύμβολα ακολουθούνται | Τα σύμβολα | Όλοι οι γνωστοί τελεστές | Στην περίπτωση που από = (πχ +=)

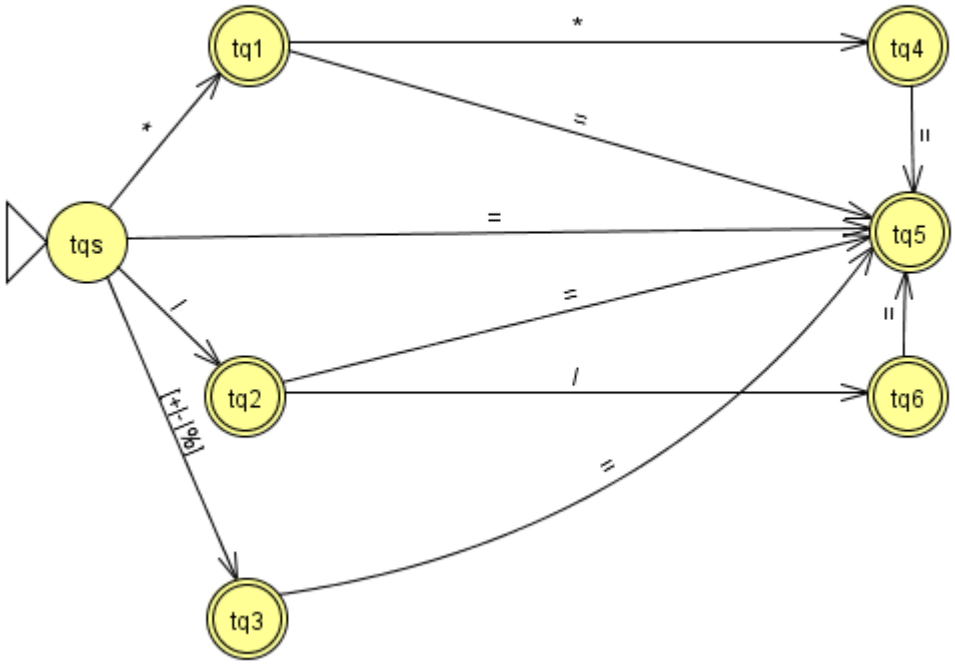
* ακριβώς 2 φορές | / ακριβώς 2 φορές

```
/(([\*{2}]|([\/{2}]|([+\-*\/%]))(=?)|=/g

Test String

20+2 20//2 20/2+5 20//320%320//3+20%3(20//3)*3+20%
x += 5 x = x + 5
x -= 5 x = x - 5
x *= 5 x = x * 5
x /= 5 x = x / 5
x %= 5 x = x % 5
x //= 5 x = x // 5 ==
x **= 5 x = x ** 5 //==
```

1.5.2 Αυτόματο και FSM :




```
START=TQS
TQS:    - + %->TQ3
        \* ->TQ1
        /->TQ2
        =->TQ5
TQ1:    - + %->BAD
        \* ->TQ4
        /->BAD
        =->TQ5
        \n->GOOD
TQ2:    - + %->BAD
        \* ->BAD
        /->TQ6
        =->TQ5
        \n->GOOD
TQ3:    - + %->BAD
        \* ->BAD
        /->BAD
        =->TQ5
        \n->GOOD
TQ4:    - + %->BAD
        \* ->BAD
        /->BAD
        =->TQ5
        \n->GOOD
TQ5:    - + %->BAD
        \* ->BAD
        /->BAD
        =->BAD
        \n->GOOD
TQ6:    - + %->BAD
        \* ->BAD
        /->BAD
        =->TQ5
        \n->GOOD
GOOD(OK):
```

1.5.3 Πίνακας μετάβασης:

Πίνακας Μεταβάσεων για τελεστές				
κανονική έκφραση: ((*{2})|(\/{2})|([+\-*\/%]))(=?)|=				
	[+ - %]	*	/	=
tqs	tq3	tq1	tq2	tq5
tq1	error	tq4	error	tq5
tq2	error	error	tq6	tq5
tq3	error	error	error	tq5
tq4	error	error	eorror	tq5
tq5	error	error	error	error
tq6	error	error	error	tq5

1.5.4 Εκτελέσεις

```
C:\Users\jim12\Desktop>fsm.exe telestes
//=
^Z
YES

C:\Users\jim12\Desktop>fsm.exe telestes
**=
^Z
YES

C:\Users\jim12\Desktop>fsm.exe telestes
+
^Z
YES

C:\Users\jim12\Desktop>fsm.exe telestes
sd5
fsm: in telestes.fsm, state 'tqs' input s not accepted
```

1.6 ΣΥΜΒΟΛΟΣΕΙΡΕΣ

1.6.0 Περιγραφή

Μια συμβολοσειρά είναι μια ακολουθία από χαρακτήρες μέσα από το αλφάβητο της γλώσσας που περικλείονται μεταξύ μονών ή διπλών εισαγωγικών.

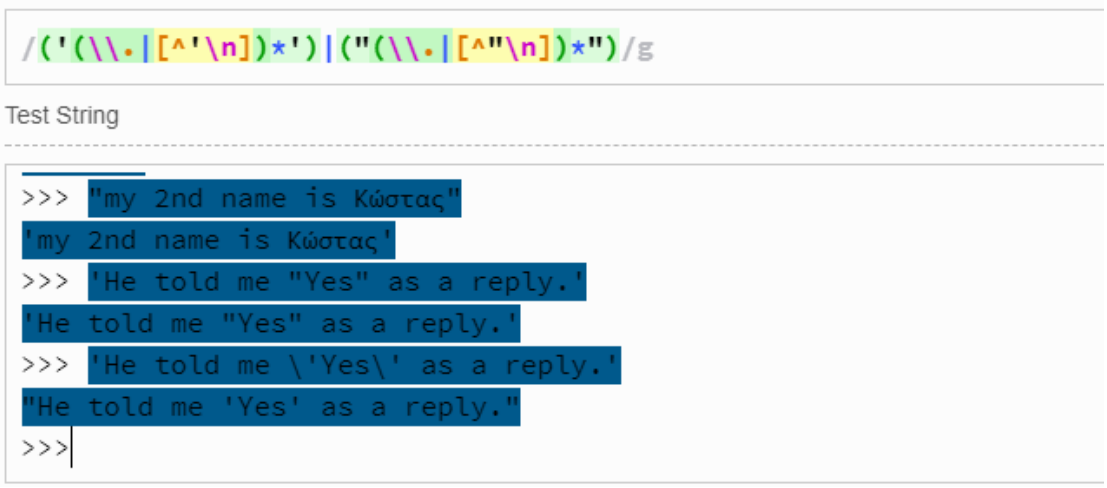
- Παραδείγματα συμβολοσειρών: 'Δημήτρης'
  "My 2nd name is Δημήτρης"
  'He told me "Yes" as a reply.'

Μια συμβολοσειρά μπορεί να επεκτείνεται σε περισσότερες της μιας γραμμές με τη χρήση ενός χαρακτήρα διαφυγής - \n στην αρχή της νέας γραμμής.
Ένας ακόμα χρήσιμος χαρακτήρας διαφυγής είναι ο στηλοθέτης (Tab) \t.

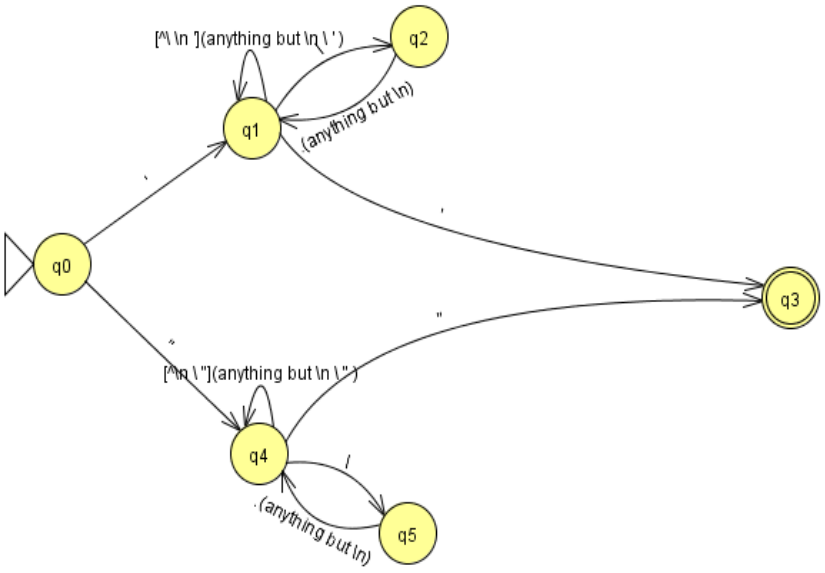
1.6.1 Κανονική έκφραση String: ('(\\.|[^'\n])*')|("(\\.|[^"\n])*")

Εντοπίζει string τα οποία μπορούν να περιέχουν οποιοδήποτε χαρακτήρα εκτός των \n "αλλά στην περίπτωση που εντοπίζει \ μπορεί να ακολουθεί οτιδήποτε μεταξύ άλλων και το "

εντοπιζει string τα οποία μπορούν να περιέχουν οποιοδήποτε χαρακτήρα εκτός των \n ‘αλλά στην περίπτωση που εντοπίζει \ μπορεί να ακολουθεί οτιδήποτε μεταξύ άλλων και το



1.6.2 Αυτόματο και FSM:




```
START=Q0
Q0: '->Q1
    "->Q4
    \n->BAD
    \\->BAD
    *->BAD
Q1: '->Q3
    "->Q1
    \n->BAD
    \\->Q2
    *->Q1
Q2: '->Q1
    "->Q1
    \n->BAD
    \\->Q1
    *->Q1
Q3: '->BAD
    "->BAD
    \n->GOOD
    \\->BAD
    *->BAD
Q4: '->Q4
    "->Q3
    \n->BAD
    \\->Q5
    *->Q4
Q5: '->Q4
    "->Q4
    \n->BAD
    \\->Q4
    *->Q4
GOOD(OK):
```

1.6.3 Πίνακας μετάβασης:

Πίνακας μετάβασης για τις συμβολοσειρές
κανονική έκφραση: ('(\\.|[^'\n])*')|("(\\.|[^"\n])*")

	'	"	\n	\	anything else
q0	q1	q4	error	error	error
q1	q3	q1	error	q2	q1
q2	q1	q1	error	q1	q1
q3	error	error	error	error	error
q4	q4	q3	error	q5	q4
g5	q4	q4	error	q4	q4

1.6.4 Εκτελέσεις

```
C:\Users\Angela\Desktop>fsm.exe string
"bla"
^Z
YES

C:\Users\Angela\Desktop>fsm.exe string
"bla\n"
^Z
YES

C:\Users\Angela\Desktop>fsm.exe string
"bla\"bla"
^Z
YES

C:\Users\Angela\Desktop>fsm.exe string
'bla'
^Z
YES

C:\Users\Angela\Desktop>fsm.exe string
'bla\'bla\'bla'
^Z
YES

C:\Users\Angela\Desktop>fsm.exe string
'bla
bla'
fsm: in string.fsm, state 'bad' input b not accepted

C:\Users\Angela\Desktop>fsm.exe string
'\'\'\'\'\'\'\'\'\\\\\''
^Z
YES
```

ΚΕΦΑΛΑΙΟ 2

Λεκτικός αναλυτής επεξήγηση:

Ο λεκτικός αναλυτής αποτελείται από τρία αρχεία που στη συνέχεια μεταγλωττίζονται στο εκτελέσιμο αρχείο καθώς και τα αρχεία εισόδου και εξόδου. Αναλυτικότερα υπάρχει το lib.h το οποίο περιέχει όλες τις global μεταβλητές και τον ορισμό όλων των συναρτήσεων , το lib.c που περιέχει την υλοποίηση των προαναφερθέντων συναρτήσεων και ουσιαστικά υλοποιεί το πρόγραμμα, και τέλος το lektikos_analuths.c στο οποίο περιέχεται η Main.

Η λειτουργία του λεκτικού αναλυτή είναι η ορθή αξιολόγηση κάθε λεξήματος. Για το σκοπό αυτό διαβάζει χαρακτήρα χαρακτήρα ένα αρχείο εισόδου, το input.txt και γράφει σε ένα αντίστοιχο αρχείο εξόδου το output.txt. Στην περίπτωση όπου αναγνωρίζει ένα ορθό token αναφέρει την γραμμή στην οποία βρήκε το token και το είδος του, ωστόσο στην περίπτωση που αναγνωρίζει ένα σχόλιο το αγνοεί. Αντίστοιχα, στην περίπτωση που εντοπίσει λάθος αναφέρει την γραμμή όπου αυτό βρέθηκε, την λέξη ERROR! Και το σύνολο των λαθών που έχει αναγνωρίσει μέχρι εκείνη τη στιγμή. Συνάμα, στην περίπτωση όπου διαβάσει χαρακτήρα ο οποίος δεν αντιστοιχεί σε κανένα γνωστό token/σχόλιο τότε δεν το θεωρεί error αλλά ως λάθος μη αναγνωρίσιμου χαρακτήρα. Τέλος, όταν πλέον διαβάσει και αξιολογήσει κάθε χαρακτήρα σημειώνει και το πλήθος των σωστών και λάθος λεξημάτων που βρήκε.

Να σημειωθεί πως το πρόγραμμα θεωρεί ως διαχωριστές, που διαχωρίζουν τα λεξήματα εισόδου το ένα από το άλλο, τα tabs(\t), τα κενά( ), οι αλλαγές γραμμής(\n) και το τέλος του αρχείου εισόδου(EOF).

Είναι απαραίτητο να επισημανθεί ότι για τον έλεγχο κάθε διαφορετικού λεξήματος έχουν δημιουργηθεί ειδικές συναρτήσεις αρμόδιες για την κάθε περίσταση. Σαφέστερα, για την ανάλυση ενός εν δυνάμη variable υπάρχει αντίστοιχη συνάρτηση η οποία συμπεραίνει εάν το λέξημα είναι αποδεκτό ή όχι. Ιδιαίτερης μεταχείρισης χρίζει η περίπτωση κατά την οποία συναντάμε + ή – αφού αυτό μπορεί να σημαίνει είτε ότι αναλύουμε integer/float είτε ακόμη και operator. Εξού και το παρακάτω κομμάτι κώδικα κατά το οποίο ελέγχουμε τι βρίσκεται μετά το σύμβολο ώστε να συμπεράνουμε ορθά τι token είναι πραγματικά:

```
else if((ch>='0' && ch<='9') || ch=='.')  //an einai INT h FLOAT h E h Telesths
    integer();                            xwris thn periptwsh na 3ekinaei apo +-
else if(ch=='/'||ch=='='||ch=='%'||ch=='*')
    op();       //an einai operator xwris ta + += - -=
else if(ch=='+' || ch=='-') //e3etazw 3exwrista
{            thn periptwsh na arxizei kati me + -
    ch=fgetc(fp);
    if(ch>='1' && ch<='9')  //einai integer
    { opote pas kateu8eian  sthn katastash EQ1
      currentState=EQ1;
      ch=fgetc(fp);
      integer();
    }
    else if(ch=='.')    //einai integer
    { opote pas kateu8eian sthn katastash EQ8
      currentState=EQ8;
      ch=fgetc(fp);
      integer();
    }
    else if(ch=='0')    //einai integer
    { opote pas kateu8eian sthn katastash EQ3
      currentState=EQ3;
      ch=fgetc(fp);
      integer();
    }
else if(ch=='=')  //einai operator
{  opote pas kateu8eian sthn katastash TQ5
   currentState=TQ5;
   ch=fgetc(fp);
   op();
}
else if(ch==' '||ch=='\n'||ch=='\t'||ch==EOF)
{ //separator einai h telikh katastash +
    currentState=GOOD;
    fprintf(fo,"line:%d token: Operator  \n",newline);

}
else
{
    currentState=BAD;
    op();
}
}
```

Αξίζει , επιπλέον ν' αναφερθούμε στην περίπτωση των string comment καθώς μπορούν να έχουν ένα ή και δύο κοινούς χαρακτήρες. Ένα string, αφενός μεν ξεκινάει με ένα " και μπορεί να ακολουθείται από οποιοδήποτε άλλον χαρακτήρα, αφετέρου δε αποτελείται από δύο " διαμορφώνοντας έτσι την, όχι και τόσο συνηθισμένη, κενή συμβολοσειρά. Ενώ τέλος εάν υπάρχει και τρίτο " τότε πλέον αναφερόμαστε σε ένα σχόλιο. Αυτό επεξηγεί και το παρακάτω κομμάτι κώδικα:

```
void string()
{
State prevState;//OUSIASTIKA H TWRINH KATASTASH
while (ch!=EOF){
switch (currentState) {
    case QS :
    {
        if(ch=='\"')
            currentState=SQ3;
        else if(ch=='\'')
            currentState=Q1;
        else
            currentState=BAD;
        prevState=QS;
        break;
    }
    case SQ3:
    {
        if(ch=='\"')
            currentState=SQ4;
        else if(ch=='\\')
            currentState=Q6;
        else if(ch!='\n')
            currentState=Q4;
        else
            currentState=BAD;
        prevState=SQ3;
        break;
    }
    case SQ4:
    {
        if(ch=='\"')
            {
            currentState=SQ5;
            ch=fgetc(fp);
            comment();
            return;
            }
        else if(ch==' '||ch=='\n'||ch=='\t'||ch==EOF) //separator
            {
            currentState=GOOD;
            fprintf(fo,"line:%d token: String  \n",newline);
            }
        else
            currentState=BAD;
        prevState=SQ4;
        break;
    }
```

```
void comment()
{
State prevState;//OUSIASTIKA H TWRINH KATASTASH
while (ch!=EOF){
switch (currentState) {
//gia na anaqnwrhsei ena comment
case SQ5: 8a "mpei" prwta sth string
            k meta 8a er8ei edw
    {
    if(ch=='\"')
        currentState=SQ11;
    else
        currentState=SQ6;
    prevState=SQ5;
    break;
    }
```

Μια ακόμα ιδιαιτερότητα υπάρχει στο τέλος κάθε συνάρτησης πριν το διάβασμα ενός καινούργιου χαρακτήρα. Λόγω του ότι οι μεταβάσεις καταστάσεων συμβαίνουν “ταχύτατα” για τα μικρά λεξήματα και επειδή έχουμε δύο “flags” (currentState prevState) που πρέπει να ενημερωθούν στην περίπτωση λάθους εμφανίζονται bugs. Ως εκ τούτου δημιουργήθηκε μια if η οποία διορθώνει αυτή την μικρή καθυστέρηση. Επίσης προνοεί και για την ύπαρξη <EOF> στο τέλος ενός αρχείου, ειδάλλως θα έπρεπε να υπάρχει κάποιος seperator αναγκαστικά πριν από το <EOF> για να μπορεί να καταλήγει σε σωστό συμπέρασμα.

```
//px diabazei . opote paei EQ8, enter (newline++) paei currentState=BAD,epeidh omws  xwris to parakatw continue diabazei ena char
//ean meta diabasei 1 einai hdh se katastash BAD opote enw einai swsto to xanei
if(currentState==BAD && prevState!=BAD)     continue;
if(currentState==BAD)                       SyntaxError();
if(ch=='\n')                                newline++;
if (currentState==GOOD)                     correct++;
if ((currentState==GOOD)||(currentState==BAD)&&(prevState==BAD))
    break;

ch=fgetc(fp);
```

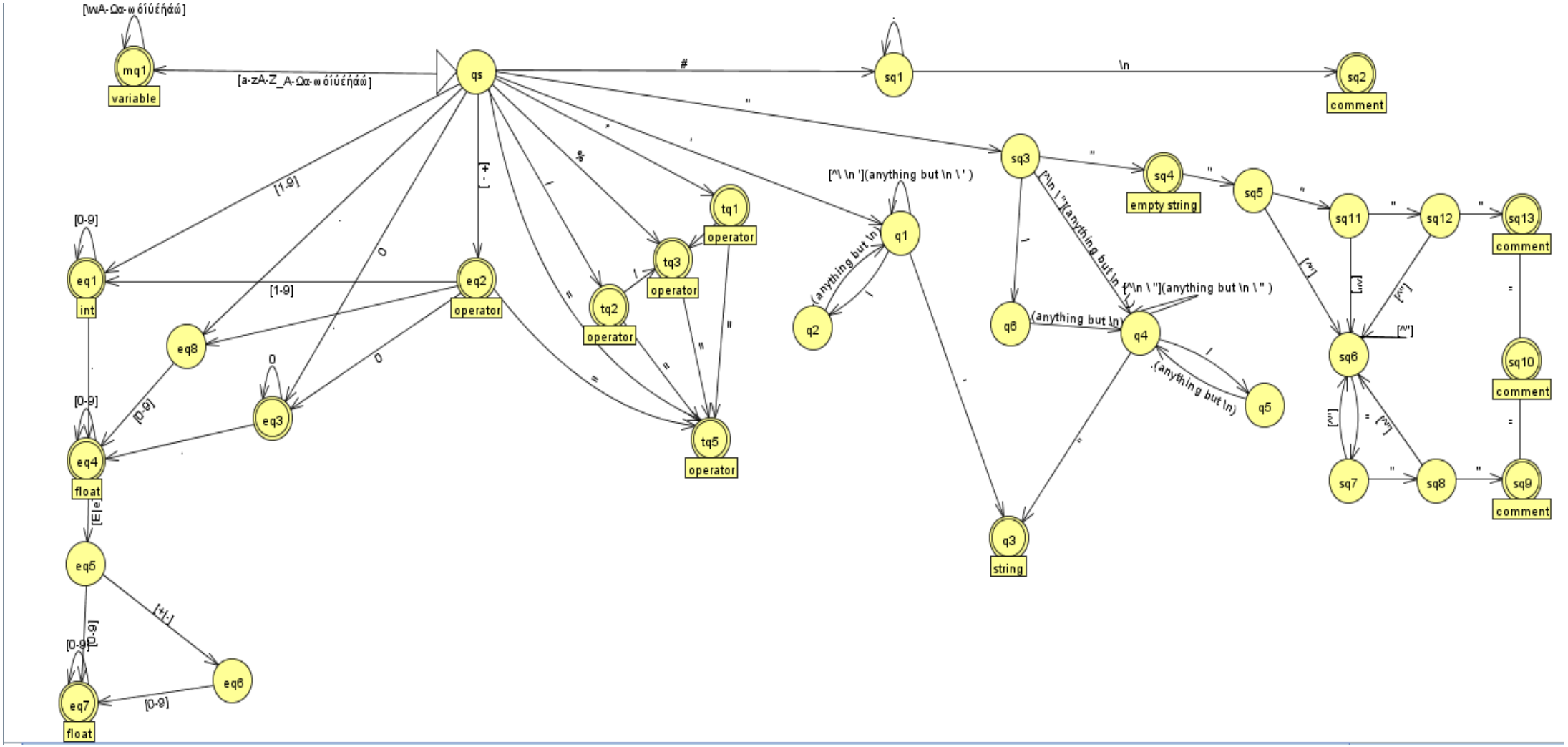
Επιπροσθέτως το πρόγραμμα περιέχει κάποιες βοηθητικές συναρτήσεις για την διεκπεραίωση του σκοπού του. Αναλυτικότερα, περιέχει μια ρουτίνα διαχείρισης λεκτικών σφαλμάτων, την SyntaxError, η οποία καλείται όταν μια ακολουθία χαρακτήρων της συμβολοσειράς εισόδου δεν μπορεί να αποκωδικοποιηθεί σε μορφή λεκτικής μονάδας. Η ρουτίνα αντιμετωπίζει το πρόβλημα με τη μέθοδο του πανικού τυπώνοντας κατάλληλο διευκρινιστικό μήνυμα και αυξάνοντας ένα μετρητή σημαντικών σφαλμάτων (fatal errors) κατά 1, όπως έχει ήδη προαναφερθεί. Υπάρχουν ακόμα οι συναρτήσεις avoidchars και avoidblanks, η πρώτη αφότου εντοπίσει λάθος ακολουθία χαρακτήρων προσπερνάει όλους εκείνους τους χαρακτήρες που δεν αποτελούν διαχωριστές της γλώσσας, ενώ η δεύτερη διαβάζει διαχωριστές μέχρι να εντοπίσει μη λευκό χαρακτήρα.

Ενοποιημένο αυτόματο:

(Λόγω του μεγέθους ακολουθεί παραπομπή του αρχείου. (Ctrl+click))

Παραρτημα

Παραρτημα


[wA-Ωα-ωόίύέήάώ]
mq1
variable
[a-zA-Z_A-Ωα-ωόίύέήάώ]
qs
#
sq1
\n
sq2
comment
"
sq3
sq4
empty string
sq5
sq11
sq12
sq13
comment
sq6
sq10
comment
sq7
sq8
sq9
comment
[^\ \n '](anything but \n \ ')
q1
q2
(anything but \n)
[^\n \ "](anything but \n \ ")
q6
(anything but \n)
q4
q5
(anything but \n)
q3
string
[1-9]
eq1
int
[0-9]
eq2
operator
[+ -]
eq3
0
eq8
[0-9]
eq4
float
[E|e]
eq5
[+|-]
eq6
[0-9]
eq7
float
[0-9]
tq1
operator
/
tq2
operator
*
tq3
operator
=
tq5
operator

Πίνακας μετάβασης για ενοποιημένο αυτόματο

	+|-	0	[1-9]	.	E|e	Α-ΖΑ-Ωα-ωέύί	%	*	/	=	#	\n	"	'	\	anything else	Exit
qs	eq2	eq3	eq1	eq8	mq1	mq1	tq3	tq1	tq2	tq5	sq1	error	sq3	q1	error	error	
eq1	error	eq1	eq1	eq4	error	error	error	error	error	error	error	error	error	error	error	int	int
eq2	error	eq3	eq1	eq8	error	error	error	error	error	tq5	error	error	error	error	error	error	operator
eq3	error	eq3	error	eq4	error	error	error	error	error	error	error	error	error	error	error	error	int
eq4	error	eq4	eq4	error	eq5	error	error	error	error	error	error	error	error	error	error	error	float
eq5	eq6	eq7	eq7	error	error	error	error	error	error	error	error	error	error	error	error	error	
eq6	error	eq7	eq7	error	error	error	error	error	error	error	error	error	error	error	error	error	
eq7	error	eq7	eq7	error	error	error	error	error	error	error	error	error	error	error	error	error	float
eq8	error	eq4	eq4	error	error	error	error	error	error	error	error	error	error	error	error	error	
mq1	error	mq1	mq1	error	mq1	mq1	error	error	error	error	error	error	error	error	error	error	variable
tq1	error	error	error	error	error	error	error	tq3	error	tq5	error	error	error	error	error	error	operator
tq2	error	error	error	error	error	error	error	error	tq3	tq5	error	error	error	error	error	error	operator
tq3	error	error	error	error	error	error	error	error	error	tq5	error	error	error	error	error	error	operator
tq5	error	error	error	error	error	error	error	error	error	error	error	error	error	error	error	error	operator
sq1	sq1	sq1	sq1	sq1	sq1	sq1	sq1	sq1	sq1	sq1	sq1	sq2	sq1	sq1	sq1	sq1	
sq2	error	error	error	error	error	error	error	error	error	error	error	error	error	error	error	error	comment
sq3	q4	q4	q4	q4	q4	q4	q4	q4	q4	q4	q4	error	sq4	q4	q6	q4	
sq4	error	error	error	error	error	error	error	error	error	error	error	error	sq5	error	error	error	string
sq5	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq11	sq6	sq6	sq6	
sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq7	sq6	sq6	sq6	
sq7	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq8	sq6	sq6	sq6	
sq8	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq9	sq6	sq6	sq6	
sq9	error	error	error	error	error	error	error	error	error	error	error	error	sq10	error	error	error	comment
sq10	error	error	error	error	error	error	error	error	error	error	error	error	sq13	error	error	error	comment
sq11	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq12	sq6	sq6	sq6	
sq12	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq6	sq13	sq6	sq6	sq6	
sq13	error	error	error	error	error	error	error	error	error	error	error	error	error	error	error	error	comment
q1	q1	q1	q1	q1	q1	q1	q1	q1	q1	q1	q1	error	q1	q3	q2	q1	
q2	q1	q1	q1	q1	q1	q1	q1	q1	q1	q1	q1	error	q1	q1	q1	q1	
q3	error	error	error	error	error	error	error	error	error	error	error	error	error	error	error	error	string
q4	q4	q4	q4	q4	q4	q4	q4	q4	q4	q4	q4	error	q3	q4	q5	q4	
q5	q4	q4	q4	q4	q4	q4	q4	q4	q4	q4	q4	error	q4	q4	q4	q4	
q6	q4	q4	q4	q4	q4	q4	q4	q4	q4	q4	q4	error	q4	q4	q4	q4	

Παρατήρηση σχετικά με σχόλια

Οι compilers της python παρουσιάζουν την εξής ανωμαλία: Δέχονται να ξεκινάει από 3 μέχρι 5" αλλά στο τέλος δέχονται μόνο 3 ή μόνο 5 ". Στην περίπτωση δηλαδή που τελειώνουν σε 4 πετάει error. Εμείς υιοθετήσαμε την παραδοχή τα σχόλια να μπορούν να ξεκινάνε απο 3 μέχρι 5 και να τελειώνουν απο 3 εως 5 εκτός βέβαια την κενή συμβολοσειρά που αποτελείται απο 6 "