Άσκηση 1

Δημιουργήστε ένα αρχείο περιγραφής του Flex το οποίο ανοίγει ένα αρχείο το όνομα του οποίου δίνεται ως παράμετρος από τη γραμμή εντολών του προγράμματος λεξικής ανάλυσης. Ο λεξικός αναλυτής θα πρέπει να εμφανίζει το μήνυμα “Αναγνώστηκαν N χαρακτήρες” όπου N το πλήθος των χαρακτήρων του αρχείου που δόθηκε σαν είσοδος εκτός των κενών. Τα κενά θα αγνοούνται. Επίσης ο λεξικός αναλυτής θα πρέπει να εμφανίζει το μήνυμα “Δε δόθηκε όνομα αρχείου” αν δεν δόθηκε όνομα αρχείου εισόδου και το μήνυμα “Δεν μπόρεσα να ανοίξω το αρχείο NAME” όπου NAME είναι το όνομα αρχείου που δόθηκε αλλά το οποίο δεν μπόρεσε να ανοιχθεί είτε γιατί δεν υπάρχει είτε γιατί δεν επαρκούν τα δικαιώματα του χρήστη για ανάγνωση του αρχείου.

Απάντηση

```
%option noyywrap
%{
#include <stdio.h>
#include <stdlib.h>
int noOfChars=0;
%}
%%
" "|\n {}
. { noOfChars++; }
%%
int main(int argc, char *argv[])
{
  if (argc<2) {
     printf("Den dothike onoma arxeiou\n");
     exit(1);
  }
  FILE* fp = fopen(argv[1],"r");
  if (!fp) {
     printf("Den boresa na anoixw to arxeio %s\n",argv[1]);
     exit(1);
  }
  yyrestart(fp);
  yylex();
  printf("Anagnwsthkan %d characthres\n",noOfChars);
  return 0;
}
```

Άσκηση 2

Εξειδικεύστε το πρόγραμμα της άσκησης 1 ώστε να χειρίζεται και τις περιπτώσεις κενών χαρακτήρων και ακεραίων αριθμών ξεχωριστά. Δηλαδή το πρόγραμμα θα πρέπει να εμφανίζει ξεχωριστά πόσους ακεραίους, πόσους κενά διαστήματα και πόσους άλλους χαρακτήρες είχε το αρχείο. Τα κενά διαστήματα είναι ακολουθίες ενός ή περισσοτέρων χαρακτήρων λευκού διαστήματος όπως ονομάζονται (whitespace characters) που είναι το

κενό, \t (tab), \n (new line), \r (return) και \f (form feed). Οι ακέραιοι είναι το 0 ή αριθμοί που ξεκινάνε από 1 μέχρι και 9 και ακολουθούνται ενδεχομένως από περισσότερα ψηφία 0 έως 9. Το πρόγραμμα θα πρέπει όπως και η άσκηση 1 να αναλύει ένα αρχείο εισόδου που δίνεται από την γραμμή εντολών και να εμφανίζει τα ίδια μηνύματα λάθους με την άσκηση 1 αν δεν δοθεί αρχείο εισόδου ή αν δεν μπορεί να ανοιχθεί για ανάγνωση.

```
%option noyywrap
%{
#include <stdio.h>
#include <stdlib.h>
int noOfSpaces=0;
int noOfIntegers=0;
int noOfChars=0;
%}
%%
0|[1-9][0-9]* { noOfIntegers++; }
[ \f\n\r\t]+  { noOfSpaces++; }
. { noOfChars++; }
%%
int main(int argc, char *argv[])
{
  if (argc<2) {
     printf("Den dothike onoma arxeiou\n");
     exit(1);
  }
  FILE* fp = fopen(argv[1],"r");
  if (!fp) {
     printf("Den boresa na anoixw to arxeio %s\n",argv[1]);
     exit(1);
  }
  yyrestart(fp);
  yylex();
  printf("Anagnwsthkan %d kena diasthmata\n",
         noOfSpaces);
  printf("Anagnwsthkan %d akeraioi\n", noOfIntegers);
  printf("Anagnwsthkan %d alloi xarakthres\n", noOfChars);
  return 0;
}
```

Άσκηση 3

Γενικεύστε την Άσκηση 2 ώστε να μπορεί να πάρει την είσοδό της από πολλά αρχεία εισόδου τα ονόματα των οποίων θα δίνονται από την γραμμή εντολών. Δηλαδή όταν τελειώνει η ανάλυση ενός αρχείου εισόδου θα συνεχίζεται με το επόμενο αρχείο εισόδου. Τα αποτελέσματα θα εμφανίζονται στο τέλος συνολικά για όλα τα αρχεία εισόδου που δόθηκαν από την γραμμή εντολών. Αν δεν δοθεί κανένα όνομα αρχείου εισόδου το πρόγραμμα θα συμπεριφέρεται όπως ακριβώς και στην Άσκηση 2 δηλ. θα τερματίζεται με μήνυμα λάθους. Αν κάποιο από τα αρχεία δεν μπορεί να βρεθεί για άνοιγμα ή δεν υπάρχουν επαρκή δικαιώματα για να διαβαστεί, το πρόγραμμα θα πρέπει να εμφανίζει μήνυμα λάθους αλλά η ανάλυση θα πρέπει να συνεχίζεται για τα υπόλοιπα αρχεία.

Απάντηση

```
%option noyywrap
%{
#include <stdio.h>
#include <stdlib.h>
int noOfSpaces=0;
int noOfIntegers=0;
int noOfChars=0;
%}
%%
0|[1-9][0-9]* { noOfIntegers++; }
[ \f\n\r\t]+  { noOfSpaces++; }
. { noOfChars++; }

%%
int main(int argc, char *argv[])
{
  int i;
  if (argc<2) {
     printf("Den dothike onoma arxeiou\n");
     exit(1);
  }
  for (i=1; i<argc; i++) {
     FILE* fp = fopen(argv[i],"r");
     if (!fp) {
       printf("Den boresa na anoixw to arxeio %s\n",argv[i]);
     }
     else {
        yyrestart(fp);
        yylex();
     }
  }
  printf("Anagnwsthkan %d kena diasthmata\n",noOfSpaces);
  printf("Anagnwsthkan %d akeraioi\n", noOfIntegers);
  printf("Anagnwsthkan %d alloi xarakthres\n", noOfChars);
  return 0;
}
```

Άσκηση 4

Δημιουργήστε ένα αρχείο περιγραφής του Flex το οποίο θα παίρνει την είσοδο απευθείας από το πληκτρολόγιο και θα είναι σε θέση να αναγνωρίζει πράξεις της μορφής: <αναγνωριστικό>=<ακέραιος ή πραγματικός><τελεστής><ακέραιος ή πραγματικός> (χωρίς κενά) μέσα στη ροή του κειμένου. Για κάθε πράξη που αναγνωρίζει θα εμφανίζει το μήνυμα «Βρήκα την πράξη ΠΡΑΞΗ» όπου ΠΡΑΞΗ θα είναι το κείμενο που αντιστοιχεί στην πράξη που αναγνωρίστηκε. Η περιγραφή να δοθεί με ορισμούς των ακόλουθων αναγνωριστικών που θα χρησιμοποιηθούν στις κανονικές εκφράσεις των κανόνων αναγνώρισης.

- IDENTIFIER (αναγνωριστικό) μία σειρά χαρακτήρων που ξεκινά με πεζό ή κεφαλαίο γράμμα και συνεχίζεται με μηδέν ή περισσότερα πεζά ή κεφαλαία γράμματα ή τον χαρακτήρα υπογράμμισης (Underscore) που είναι ο χαρακτήρας «_».
- INTEGER (ακέραιος) ένας αριθμός που μπορεί να είναι το μηδέν ή να ξεκινά από άλλο ψηφίο και να έχει και άλλα ψηφία ενδεχομένως μετά το πρώτο ψηφία.
- REAL (πραγματικός) ένας ακέραιος ακολουθούμενος προαιρετικά από μία τελεία που ακολουθείται με την σειρά της από ένα ή περισσότερα ψηφία.
- OPERATOR (τελεστής) ένας από τους χαρακτήρες: + - * /

Απάντηση

```
%option noyywrap
%{
#include <stdio.h>
%}
IDENTIFIER [a-zA-Z][a-zA-Z_]*
INTEGER 0|[1-9][0-9]*
REAL {INTEGER}(\.[0-9]+)?
OPERATOR [-+/*]
%%
{IDENTIFIER}=({REAL}|{INTEGER}){OPERATOR}({REAL}|{INTEGER}){
           printf("Vrhka thn praxh %s\n",yytext);}
.|\n
%%
int main(void)
{
  yylex();
  return 0;
}
```