

Αναφορά Project Flex-Bison

Στοιχεία ομάδας:

Όλα τα μέλη της ομάδας είναι στο 3^ο έτος.

Όνομα	ΑΜ	email
Βασιλάκη Ελένη	1070939	vasilaki.eleni2000@gmail.com
Γιαννοπούλου Αρχοντία	1070933	arhodia30@gmail.com
Λιούμη Κυριακή	1067410	kyriaki.lm@gmail.com

BNF: (η τελική μορφή μαζί με τα structs και τα σχόλια του ερωτήματος 4)

(ο σχολιασμός γίνεται με το κόκκινο χρώμα)

Όπου «ε» το κενό

Αρχή της γραμματικής :

`<start> ::= <program>`

Αρχή του προγράμματος:

`<program> ::= PROGRAM <name> <lines> <body>`

Δυνατότητα αλλαγής μιας γραμμής ή πολλών γραμμών :

`<lines> ::= CHANGE_LINE`

`| CHANGE_LINE <lines>`

Υπάρχει η επιλογή σε περίπτωση που δεν θέλουμε να βάλουμε συνάρτηση ή σε περίπτωση που δεν θέλουμε να βάλουμε structs.

`<body> ::= <main>`

`| <structs> <main>`

`| <structs> <functions> <main>`

Τα structs είναι προαιρετικά, οπότε υπάρχει η δυνατότητα να βάλουμε από 0 μέχρι όσες φορές θέλουμε:

<structs> ::= ε
| <struct> <structs>

Η περίπτωση να έχουμε ή απλό struct ή με typedef:

<struct> ::= STRUCT NAME <lines> <var> ENDSTRUCT <lines>
| TYPEDEF STRUCT NAME <lines> <var> NAME ENDSTRUCT <lines>

Περίπτωση μίας ή πολλών συναρτήσεων. Η περίπτωση που δεν θέλουμε καμία, καλύπτεται πιο πάνω στο body.

<functions> ::= <function>
| <function> <functions>

Υλοποίηση συνάρτησης:

<function> ::= FUNCTION NAME LEFTBRA <list> RIGHTBRA <lines> <var> <commands> RETURN NAME <lines>

Τα ορίσματα της συνάρτησης. Δυνατότητα επιλογής ανάμεσα σε μια μεταβλητή, πολλές μεταβλητές και πίνακα:

<list> ::= NAME
| NAME COMMA <list>
| NAME LEFTSTRBRA INT RIGHTSTRBRA
| NAME LEFTSTRBRA INT RIGHTSTRBRA COMMA <list>

Δυνατότητα δήλωσης πολλών μεταβλητών ή και καμίας:

<var> ::= ε
| <var2> <var>

Ορισμός μεταβλητών:

<var2> ::= VARS <lines> <multiple>

Δυνατότητα επιλογής ανάμεσα σε μια μεταβλητή, πολλές μεταβλητές και πίνακα, στην ίδια γραμμή ή σε διαφορετική γραμμή:

<multiple> ::= ε
| <type_specifier> <list> ΕΡΟΤΙΜΑΤΙΚΟ <lines> <multiple>

Τύποι μεταβλητών. Ο τρίτος τύπος είναι για το struct:

<type_specifier> ::= CHAR
| INTEGER

| NAME

Δυνατότητα επιλογής πολλών εντολών :

```
<commands> ::= <command>  
            | <command> <commands>
```

Είδη εντολών :

```
<command> ::= <comment>  
            | <loop>  
            | <conditions>  
            | <print>  
            | <assignment>  
            | <big_comment>
```

Είδη loop:

```
<loop> ::= <while>  
         | <for>
```

Υλοποίηση της while:

```
<while> ::= WHILE LEFTBRA <condition> RIGHTBRA <lines> <break> <commands> <break> ENDWHILE  
<lines>
```

Υλοποίηση της for :

```
<for> ::= FOR NAME ANOKATOTEL ISON INT TO INT STEP INT <lines> <break> <commands> <break>  
ENDFOR <lines>
```

Είδη condition :

```
<conditions> ::= <if>  
              | <switch>
```

Υλοποίηση της if:

```
<if> ::= IF LEFTBRA <condition> RIGHTBRA THEN <lines> <commands> <elseif2> <else> ENDIF <lines>
```

Υλοποίηση της else if. Δυνατότητα επιλογής 0,1 ή πολλές φορές:

```
<elseif2> ::= ε  
          | ELSEIF <lines> <commands> <elseif2>
```

Υλοποίηση της else . Δυνατότητα επιλογής 0 ή 1 φορές

<else> ::= ε

| ELSE <lines> <commands>

Υλοποίηση switch:

<switch> ::= SWITCH LEFTBRA <condition> RIGHTBRA <lines> <case2>

<default> ENDSWITCH <lines>

Δυνατότητα επιλογής πόσες φορές θα χρησιμοποιηθεί το case.

<case2> ::= ε

| <case> <case2>

Υλοποίηση του case.

<case> ::= CASE LEFTBRA <condition> RIGHTBRA ANOKATOTEL <lines> <break> <commands> <break>

Υλοποίηση του default. Επιλογή 0 ή 1 φορά.

<default> ::= ε

| DEFAULT ANOKATOTEL <lines> <break> <commands> <break>

Υλοποίηση του print με και χωρίς εκτύπωση μεταβλητής:

<print> ::= PRINT LEFTBRA <words> RIGHTBRA EROTIMATIKO <lines>

| PRINT LEFTBRA <words> LEFTSTRBRA COMMA NAME RIGHTSTRBRA RIGHTBRA
EROTIMATIKO <lines>

Υλοποίηση συνθηκών :

<condition> ::= NAME <symbol> NAME

| <condition> <logic> <condition>

Σύμβολά σύγκρισης

<symbol> ::= MEGALYTERO

| MIKROTERO

| ISON ISON

| THAYMASTIKO

Λογικές πράξεις:

<logic> ::= <and>

| <or>

Υλοποίηση εντολής ανάθεσης :

<assignment> ::= NAME ISON <expression> EROTIMATIKO <lines>

Επιλογή του δεύτερου μέλους της εξίσωσης. Οι επιλογές είναι : συνάρτηση, μεταβλητή, πράξη μεταξύ μεταβλητών και χρήση παρενθέσεων

<expression> ::= INT

| <func>

| NAME

| <expression> <symbol2> <expression>

| LEFTBRA <expression> RIGHTBRA

Πράξεις :

<symbol2> SYN

| PLIN

| MUL

| DIV

| POW

Κλήση συνάρτησης:

<func> ::= NAME LEFTBRA <list> RIGHTBRA

Υλοποίηση break. Δυνατότητα επιλογής 0 ή 1 φορά

<break> ::= ε

| <break> EROTIMATIKO <lines>

Υλοποίηση σχολίου

<comment> ::= TISEKATO <words> <lines>

Υλοποίηση σχολίου του 4^{ου} ερωτήματος. Επιλογή να είναι σε μία γραμμή ή σε πολλές

<big_comment> ::= COMSTART <words> COMEND <lines>

| COMSTART CHANGE_LINE <test> COMEND <lines>

Δυνατότητα επιλογής τι θα βάλουμε μέσα στο σχόλιο των πολλών γραμμών. Οι επιλογές είναι : κείμενο, αλλαγή γραμμών , εντολές

```
<test> ::= ε
        | <words> CHANGE_LINE <test>
        | <lines> <test>
        | <command> <test>
```

Υλοποίηση της main :

```
<main> ::= STARTMAIN <lines> <var> <commands> ENDMAIN <lines>
```

Δυνατότητα να γράψουμε κείμενο με κενά ανάμεσα στις λέξεις. Χρησιμοποιείται στα σχόλια και το print.

```
<words> ::= NAME
        | NAME <words>
```

Flex:

```
%{
```

Δήλωση βιβλιοθηκών που χρησιμοποιούνται:

```
#include "y.tab.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Δήλωση συναρτήσεων που χρειάζονται και αρχείων για το input και το output

```
extern int yylex();
extern FILE *yyin;
extern FILE *yyout;
```

Δήλωση μεταβλητής που θα μετράει τις γραμμές για να βλέπουμε σε ποια γραμμή βρίσκεται το σφάλμα.

```
int line_count=1;

%}
```

%option noyywrap

Ορισμοί ονομάτων :

digit [0-9]

letter [a-zA-Z]

num {digit}+

%%

Δήλωση δεσμευμένων λέξεων και συμβόλων που χρησιμοποιούμε στο πρόγραμμα. Τα αντιμετωπίζουμε όλα ως string, έτσι ώστε να εκτυπώνονται οι λέξεις. Μετά κάνουμε return την συγκεκριμένη δεσμευμένη λέξη που θέλουμε κάθε φορά, ακριβώς έτσι όπως είναι (ECHO και return).

"PROGRAM" {yyval.string = strdup(yytext); ECHO;return PROGRAM;}

"STARTMAIN" {yyval.string = strdup(yytext); ECHO;return STARTMAIN;}

"VARS" {yyval.string = strdup(yytext); ECHO;return VARS;}

"CHAR" {yyval.string = strdup(yytext); ECHO;return CHAR;}

"INTEGER" {yyval.string = strdup(yytext); ECHO;return INTEGER;}

"STRUCT" {yyval.string = strdup(yytext); ECHO;return STRUCT;}

"TYPEDEF" {yyval.string = strdup(yytext); ECHO;return TYPEDEF;}

"ENDSTRUCT" {yyval.string = strdup(yytext); ECHO;return ENDSTRUCT;}

"FUNCTION" {yyval.string = strdup(yytext); ECHO;return FUNCTION;}

"RETURN" {yyval.string = strdup(yytext); ECHO;return RETURN;}

"END_FUNCTION" {yyval.string = strdup(yytext); ECHO;return END_FUNCTION;}

"ENDMAIN" {yyval.string = strdup(yytext); ECHO;return ENDMAIN;}

"WHILE" {yyval.string = strdup(yytext); ECHO;return WHILE;}

"ENDWHILE" {yyval.string = strdup(yytext); ECHO;return ENDWHILE;}

"FOR" {yyval.string = strdup(yytext); ECHO;return FOR;}

"TO" {yyval.string = strdup(yytext); ECHO;return TO;}

"STEP" {yyval.string = strdup(yytext); ECHO;return STEP;}

"ENDFOR" {yyval.string = strdup(yytext); ECHO;return ENDFOR;}

"IF" {yyval.string = strdup(yytext); ECHO;return IF;}

"THEN" {yyval.string = strdup(yytext); ECHO;return THEN;}

```

"ELSEIF"    {yyval.string = strdup(yytext); ECHO;return ELSEIF;}
"ELSE"      {yyval.string = strdup(yytext); ECHO;return ELSE;}
"ENDIF"     {yyval.string = strdup(yytext); ECHO;return ENDIF;}
"SWITCH"    {yyval.string = strdup(yytext); ECHO;return SWITCH;}
"CASE"      {yyval.string = strdup(yytext); ECHO;return CASE;}
"DEFAULT"   {yyval.string = strdup(yytext); ECHO;return DEFAULT;}
"ENDSWITCH" {yyval.string = strdup(yytext); ECHO;return ENDSWITCH;}

"PRINT"     {yyval.string = strdup(yytext); ECHO;return PRINT;}
"BREAK"     {yyval.string = strdup(yytext); ECHO;return BREAK;}
"AND"       {yyval.string = strdup(yytext); ECHO;return AND;}
"OR"        {yyval.string = strdup(yytext); ECHO;return OR;}

"("         {yyval.string = strdup(yytext); ECHO;return LEFTBRA; }
"["         {yyval.string = strdup(yytext); ECHO;return LEFTSTRBRA; }
">"        {yyval.string = strdup(yytext); ECHO;return MEGALYTERO ; }
"<"        {yyval.string = strdup(yytext); ECHO;return MIKROTERO ; }
"!"         {yyval.string = strdup(yytext); ECHO;return THAYMASTIKO ; }
"="         {yyval.string = strdup(yytext); ECHO;return ISON ; }
", "        {yyval.string = strdup(yytext); ECHO;return COMMA ; }
"]"         {yyval.string = strdup(yytext); ECHO;return RIGHTSTRLBRA ;}
")"         {yyval.string = strdup(yytext); ECHO;return RIGHTBRA ; }
":"         {yyval.string = strdup(yytext); ECHO;return ANOKATOTEL ; }
";"         {yyval.string = strdup(yytext); ECHO;return EROTIMATIKO ; }
"+"         {yyval.string = strdup(yytext); ECHO;return SYN ; }
"_"         {yyval.string = strdup(yytext); ECHO;return PLIN; }
"*"         {yyval.string = strdup(yytext); ECHO;return MUL ; }
"/"         {yyval.string = strdup(yytext); ECHO;return DIV; }
"^"         {yyval.string = strdup(yytext); ECHO;return POW; }
"%"         {yyval.string = strdup(yytext); ECHO;return TISEKATO ; }
"/*"        {yyval.string = strdup(yytext); ECHO;return COMSTART; }

```



```
"*/"      {yyval.string = strdup(yytext); ECHO;return COMEND; }
```

```
[\n]      {line_count++ ;yyval.string = strdup(yytext); ECHO; return CHANGE_LINE ; }
```

Εδώ δεν έχουμε return γιατί δεν χρειάζεται να μας επιστρέφει τίποτα για τα κενά

```
[ \t]*    {yyval.string = strdup(yytext); ECHO;}
```

Δήλωση αριθμών

```
{num}     {yyval.string = strdup(yytext); ECHO; return INT; }
```

Δήλωση ονομάτων για μεταβλητές κ.α

```
[a-zA-Z_][a-zA-Z0-9_]* {yyval.string = strdup(yytext); ECHO; return NAME; }
```

```
%%
```

Bison:

Καθώς έχουν εξηγηθεί ήδη οι κανόνες της γραμματικής από το bnf. Εδώ θα αναλυθούν μόνο τα επιπλέον στοιχεία. Ωστόσο έχουμε παραθέσει όλο το bison για να υπάρχει μια συνέχεια.

```
%{
```

Δήλωση βιβλιοθηκών που χρησιμοποιούνται

```
#include <stdio.h>
```

```
#include <string.h>
```

Δήλωση αρχείων για input και output

```
extern FILE *yyin;
```

```
extern FILE *yyout;
```

Δήλωση συναρτήσεων που χρειάζονται:

```
extern int yylex();
```

```
extern int yyparse();
```

```
extern char* yytext();
```

```
extern int line_count;
```

```
void yyerror(const char *s);
```

Δήλωση μεταβλητής που θα χρησιμοποιηθεί για τα errors

```
int error=0;
```

```
%}
```

Δηλώσεις bison :

Δεσμευμένες λέξεις: (σε string για να μπορούν να εκτυπώνονται στο output)

```
%token<string> PROGRAM NAME CHANGE_LINE STARTMAIN ENDMAIN FUNCTION LEFTBRA  
RIGHTBRA
```

```
%token<string> RETURN END_FUNCTION VARS CHAR INTEGER EROTIMATIKO INT BREAK
```

```
%token<string> PRINT LEFTSTRBRA COMMA RIGHTSTRLBRA TISEKATO WHILE ENDWHILE
```

```
%token<string> FOR ANOKATOTEL ISON TO STEP ENDFOR MEGALYTERO MIKROTERO THAYMASTIKO  
AND OR
```

```
%token<string> IF THEN ELSEIF ELSE ENDIF SWITCH CASE DEFAULT ENDSWITCH SYN PLIN MUL DIV  
POW
```

```
%token<string> STRUCT TYPEDEF ENDSTRUCT TEXT COMSTART COMEND SPACE
```

```
%left SYN PLIN
```

```
%left MUL DIV
```

```
%right POW
```

Ορισμός που είναι η αρχή των κανόνων μας

```
%start start
```

Για υλοποίηση string

```
%union{
```

```
char* string;
```

```
}
```

Δήλωση ως string τους τύπους

```
%type<string> start program body function functions main var type_specifier var2
```

```
%type<string> commands command comment loop while for list condition symbol logic
```

```
%type<string> print conditions if switch assignment func break symbol2 lines multiple
```

```
%type<string> structs struct big_comment
```

```
%%
```

Το start χρησιμοποιείται για να μπορούμε να εκτυπώσουμε όλο το πρόγραμμα με μία εντολή. Με την εντολή fprintf εκτυπώνουμε σε αρχείο

```
start: program      {fprintf(yyout, $1);};
```

```
program: PROGRAM NAME lines body ;
```

```
lines: CHANGE_LINE
```

```
| CHANGE_LINE lines
```

```
;
```

```
body:main
```

```
|structs functions main
```

```
;
```

```
structs: %empty {}
```

```
|struct structs
```

```
;
```

Ο τύπος που έχει χρησιμοποιηθεί για τα structs είναι ο “ kiriaki “. Και με τις if ελέγχουμε αν η δήλωση είναι σωστή. Αν ο τύπος που δηλωθεί έχει άλλο όνομα θα έχουμε σφάλμα. Επίσης αν δημιουργηθούν δύο struct με το ίδιο όνομα θα εμφανιστεί πάλι σφάλμα.

```
struct: STRUCT NAME lines var ENDSTRUCT lines {
```

```
    if (strcmp($2,"kiriaki") == 0 && count < 1) {count++;}
```

```
    else if (strcmp($2,"kiriaki") != 0){ yyerror("Wrong type for the struct\n") ;}
```

```
    else {yyerror("The struct already exists\n");};}
```

Ομοίως με παραπάνω η επεξήγηση του κώδικα, μόνο που εδώ έχουμε πιο πολλές περιπτώσεις στα if , αφού θέλουμε να έχουμε το ίδιο όνομα και στην αρχή και στο τέλος του struct. Βάλαμε ένα comment καθώς το όνομα την δεύτερη φορά αναγνωριζόταν ως μεταβλητή και είχαμε error.

```
| TYPEDEF STRUCT NAME lines var big_comment NAME ENDSTRUCT lines {
```

```
    if (strcmp($3,"kiriaki") == 0 && strcmp($7,"kiriaki") == 0 && count < 1) {count++;}
```

```
    else if (strcmp($3,"kiriaki") != 0){ yyerror("Wrong type for the struct\n") ;}
```

```
    else if (strcmp($3,"kiriaki") == 0 && strcmp($7,"kiriaki") != 0){ yyerror("The name at the  
end is not the same with the beginning \n") ;}
```

```
    else {yyerror("The struct already exists\n");};}
```

```
;
```

```
functions: function
```

```
| function functions
```

```
;
```

function: FUNCTION NAME LEFTBRA list RIGHTBRA lines var commands

RETURN NAME lines END_FUNCTION lines

;

list: NAME

| NAME COMMA list

| NAME LEFTSTRBRA INT RIGHTSTRLBRA

| NAME LEFTSTRBRA INT RIGHTSTRLBRA COMMA list;

var: %empty {}

| var2 var

;

var2: VARS lines multiple

;

multiple: %empty {}

| type_specifier list EROTIMATIKO lines multiple

;

Με το κομμάτι κώδικα ελέγχουμε αν έχει δημιουργηθεί struct με τέτοιο όνομα. Αν δεν έχει δημιουργηθεί τότε δεν μπορούμε να δηλώσουμε μεταβλητή με τέτοιο τύπο. Οπότε βγάζει σφάλμα.

type_specifier: CHAR

| INTEGER

| NAME {if (strcmp(\$1,"kiriaki")!=0){yyerror("This type does not exist");}

else {};};

;

commands: command

| command commands

;

command:comment

|loop

|conditions

|print

|assignment

```
| big_comment
;
loop: while
    | for
;
while: WHILE LEFTBRA condition RIGHTBRA lines break commands
break ENDWHILE lines
;
for: FOR NAME ANOKATOTEL ISON INT TO INT STEP INT lines
break commands break ENDFOR lines
;
conditions: if
    | switch
;
if: IF LEFTBRA condition RIGHTBRA THEN lines commands elseif2 else ENDIF lines
;
elseif2: %empty {}
    | ELSEIF lines commands elseif2
;
else: %empty
    | ELSE lines commands
;
switch: SWITCH LEFTBRA condition RIGHTBRA lines case2 default ENDSWITCH lines
;
case2: %empty {}
    | case case2
;
case: CASE LEFTBRA condition RIGHTBRA ANOKATOTEL lines break commands break
;
```

```
default: %empty {}
    | DEFAULT ANOKATOTEL lines break commands break
;

print: PRINT LEFTBRA words RIGHTBRA EROTIMATIKO lines
    | PRINT LEFTBRA words LEFTSTRBRA COMMA NAME RIGHTSTRBRA RIGHTBRA EROTIMATIKO lines
;

condition: NAME symbol NAME
    | condition logic condition;
symbol: MEGALYTERO
    | MIKROTERO
    | ISON ISON
    | THAYMASTIKO
;

logic: AND
    | OR
;

assignment: NAME ISON expression EROTIMATIKO lines
;

expression: INT
    | func
    | NAME
    | expression symbol2 expression
    | LEFTBRA expression RIGHTBRA
;

symbol2: SYN
    | PLIN
    | MUL
    | DIV
    | POW
```

```

;
func: NAME LEFTBRA list RIGHTBRA
;
break: %empty {}
        | BREAK EROTIMATIKO lines
;
comment: TISEKATO NAME lines
;
big_comment: COMSTART NAME COMEND lines
        | COMSTART CHANGE_LINE test COMEND lines
;
test: %empty {}
        | NAME CHANGE_LINE test
        | lines test
        | command test
;
main: STARTMAIN lines var commands ENDMAIN lines
;
words: NAME
        | NAME words
;
%%

```

Συνάρτηση για τα errors. Αν βρεθεί error , σταματά η εκτέλεση του προγράμματος και εμφανίζεται στο αρχείο με την έξοδο το κατάλληλο μήνυμα με το error και την γραμμή στην οποία βρίσκεται.

```

void yyerror(const char *s) {
    printf("error %s\n",s);

    fprintf(yyout,"\n%s at line %d",s,line_count);

    error++;
}

```

Συνάρτηση main. Διαβάζεται το αρχείο εισόδου και τα αποτελέσματα γράφονται στο αρχείο εξόδου.

```
int main ( int argc, char **argv )
{
    ++argv; --argc;
    if ( argc > 0 )
        yyin = fopen( argv[0], "r" );
    else
        yyin = stdin;
    yyout = fopen ( "output", "w" );
    yyparse ();
    printf("errors: %d",error);
    return 0;}
```

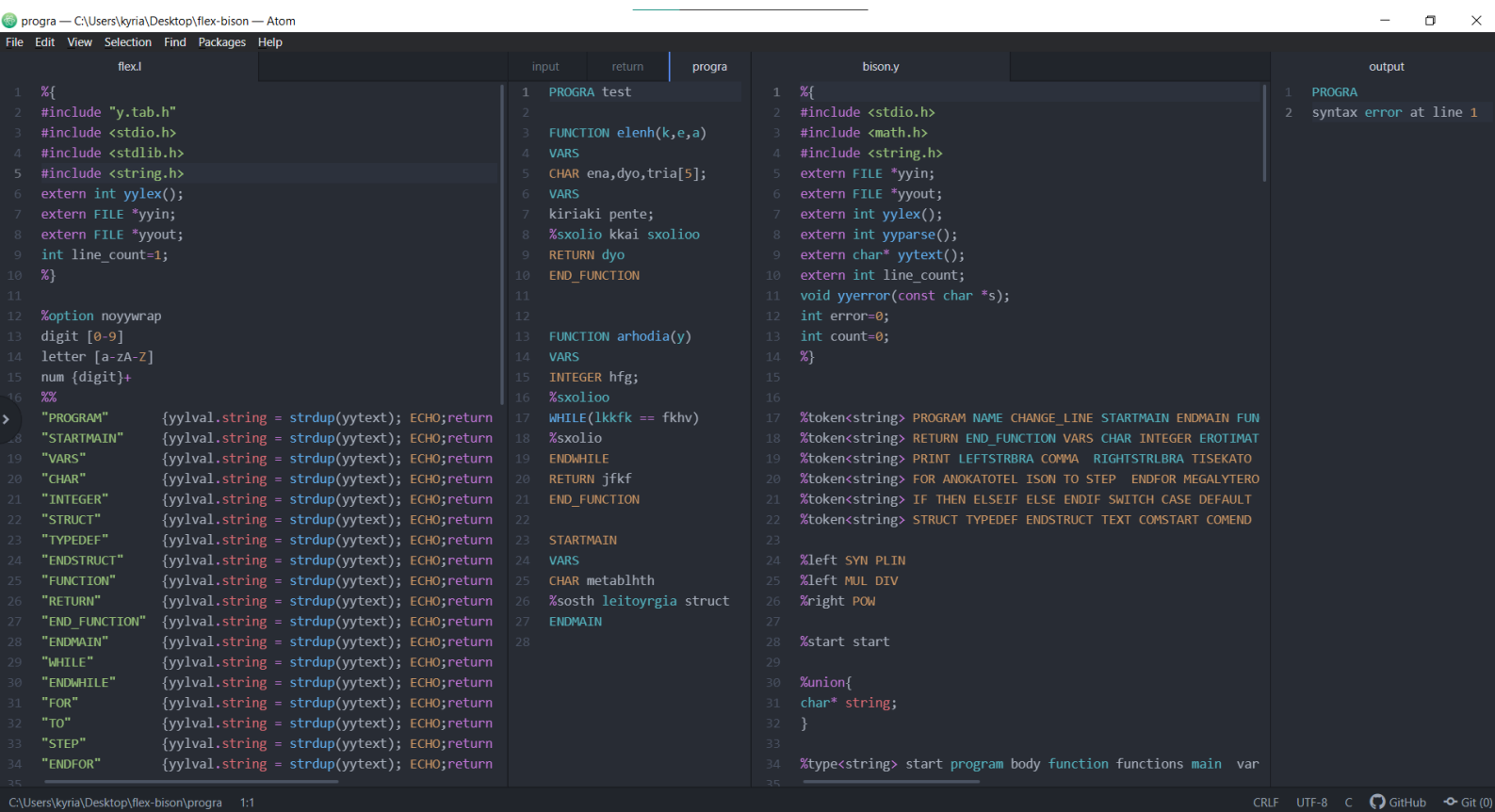

Screenshots:

Δεν μπορούμε να βάλουμε όλα τα πιθανά errors , οπότε θα βάλουμε 2-3 παραδείγματα από κάθε ερώτημα.

Σε κάθε screenshot βλέπουμε 4 στήλες. Αριστερά είναι το flex, μετά το input, μετά το bison και δεξιά το output.

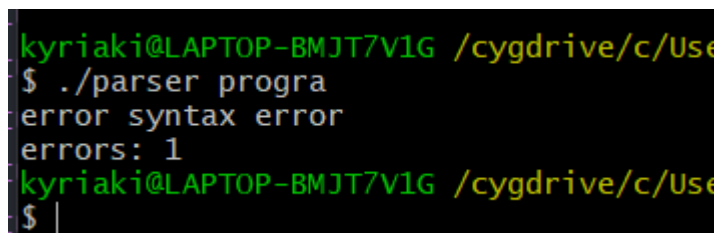
Παραδείγματα από Ερώτημα 1 :

- Έχοντας γράψει λάθος τη δεσμευμένη λέξη PROGRAM, βλέπουμε ότι στο output δεν εκτυπώνεται όλο το input καθώς προκύπτει syntax error στην γραμμή 1.



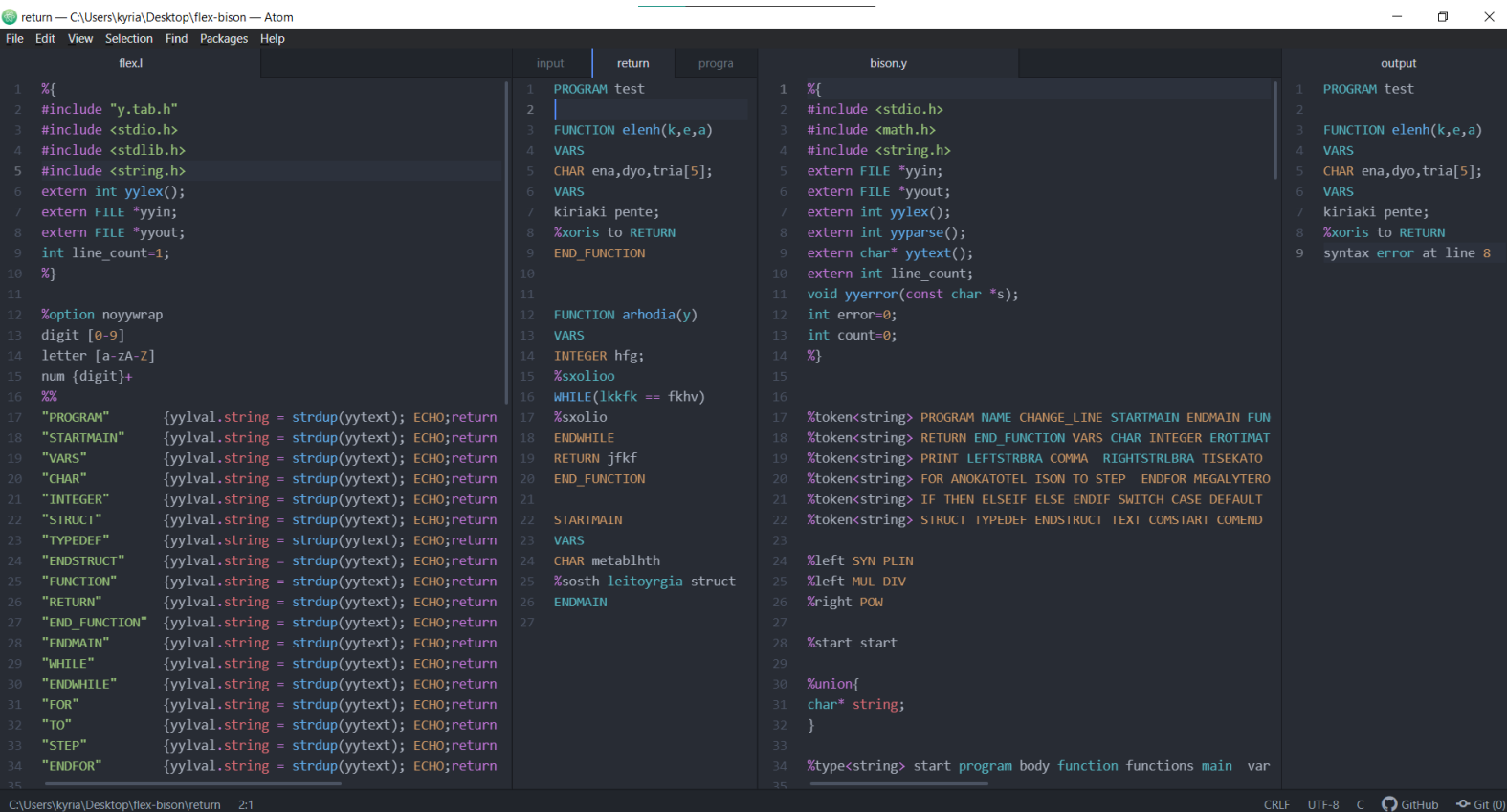
```
1  %{
2  #include "y.tab.h"
3  #include <stdio.h>
4  #include <stdlib.h>
5  #include <string.h>
6  extern int yylex();
7  extern FILE *yyin;
8  extern FILE *yyout;
9  int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17
18 "PROGRAM"      {yylval.string = strdup(yytext); ECHO;return 17
19 "STARTMAIN"    {yylval.string = strdup(yytext); ECHO;return 18
20 "VARS"         {yylval.string = strdup(yytext); ECHO;return 19
21 "CHAR"        {yylval.string = strdup(yytext); ECHO;return 20
22 "INTEGER"      {yylval.string = strdup(yytext); ECHO;return 21
23 "STRUCT"       {yylval.string = strdup(yytext); ECHO;return 22
24 "TYPEDEF"      {yylval.string = strdup(yytext); ECHO;return 23
25 "ENDSTRUCT"    {yylval.string = strdup(yytext); ECHO;return 24
26 "FUNCTION"     {yylval.string = strdup(yytext); ECHO;return 25
27 "RETURN"       {yylval.string = strdup(yytext); ECHO;return 26
28 "END_FUNCTION" {yylval.string = strdup(yytext); ECHO;return 27
29 "ENDMAIN"      {yylval.string = strdup(yytext); ECHO;return 28
30 "WHILE"        {yylval.string = strdup(yytext); ECHO;return 29
31 "ENDWHILE"     {yylval.string = strdup(yytext); ECHO;return 30
32 "FOR"          {yylval.string = strdup(yytext); ECHO;return 31
33 "TO"           {yylval.string = strdup(yytext); ECHO;return 32
34 "STEP"         {yylval.string = strdup(yytext); ECHO;return 33
35 "ENDFOR"       {yylval.string = strdup(yytext); ECHO;return 34
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

Βλέπουμε και το αποτέλεσμα στο Cygwin. Χωρίς να εκτυπώνεται εκεί το πρόγραμμα, μόνο τα errors. Το πρόγραμμα εκτυπώνεται μόνο στο output.



```
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c/Use
$ ./parser progra
error syntax error
errors: 1
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c/Use
$ |
```

2. Στην είσοδο αυτή έχουμε παραλείψει από την μια συνάρτηση το RETURN. Έτσι βλέπουμε στο output δεξιά ότι υπάρχει syntax error στην γραμμή 8, δηλαδή στην γραμμή που θα έπρεπε να βρίσκεται το RETURN.



```
return — C:\Users\kyria\Desktop\flex-bison — Atom
File Edit View Selection Find Packages Help

flex.l      input      return     progra     bison.y      output

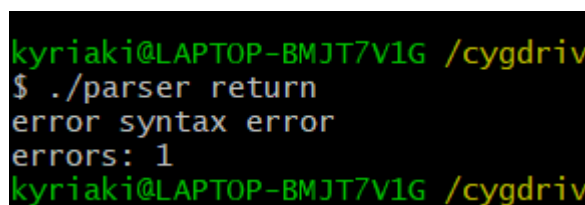
1  %{
2  #include "y.tab.h"
3  #include <stdio.h>
4  #include <stdlib.h>
5  #include <string.h>
6  extern int yylex();
7  extern FILE *yyin;
8  extern FILE *yyout;
9  int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17 "PROGRAM"      {yylval.string = strdup(yytext); ECHO;return
18 "STARTMAIN"    {yylval.string = strdup(yytext); ECHO;return
19 "VARS"          {yylval.string = strdup(yytext); ECHO;return
20 "CHAR"          {yylval.string = strdup(yytext); ECHO;return
21 "INTEGER"       {yylval.string = strdup(yytext); ECHO;return
22 "STRUCT"        {yylval.string = strdup(yytext); ECHO;return
23 "TYPEDEF"       {yylval.string = strdup(yytext); ECHO;return
24 "ENDSTRUCT"     {yylval.string = strdup(yytext); ECHO;return
25 "FUNCTION"      {yylval.string = strdup(yytext); ECHO;return
26 "RETURN"        {yylval.string = strdup(yytext); ECHO;return
27 "END_FUNCTION"  {yylval.string = strdup(yytext); ECHO;return
28 "ENDMAIN"       {yylval.string = strdup(yytext); ECHO;return
29 "WHILE"         {yylval.string = strdup(yytext); ECHO;return
30 "ENDWHILE"      {yylval.string = strdup(yytext); ECHO;return
31 "FOR"           {yylval.string = strdup(yytext); ECHO;return
32 "TO"            {yylval.string = strdup(yytext); ECHO;return
33 "STEP"          {yylval.string = strdup(yytext); ECHO;return
34 "ENDFOR"        {yylval.string = strdup(yytext); ECHO;return
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

input      return     progra     bison.y      output

1  PROGRAM test
2  |
3  FUNCTION elenh(k,e,a)
4  VARS
5  CHAR ena,dyo,tria[5];
6  VARS
7  kiriaki pente;
8  %xoris to RETURN
9  END_FUNCTION
10
11
12 FUNCTION arhodia(y)
13 VARS
14 INTEGER hfg;
15 %sxoliao
16 WHILE(lkkfk == fkhv)
17 %sxolio
18 ENDWHILE
19 RETURN jfkf
20 END_FUNCTION
21
22 STARTMAIN
23 VARS
24 CHAR metablth
25 %soth leitoyrgia struct
26 ENDMAIN
27

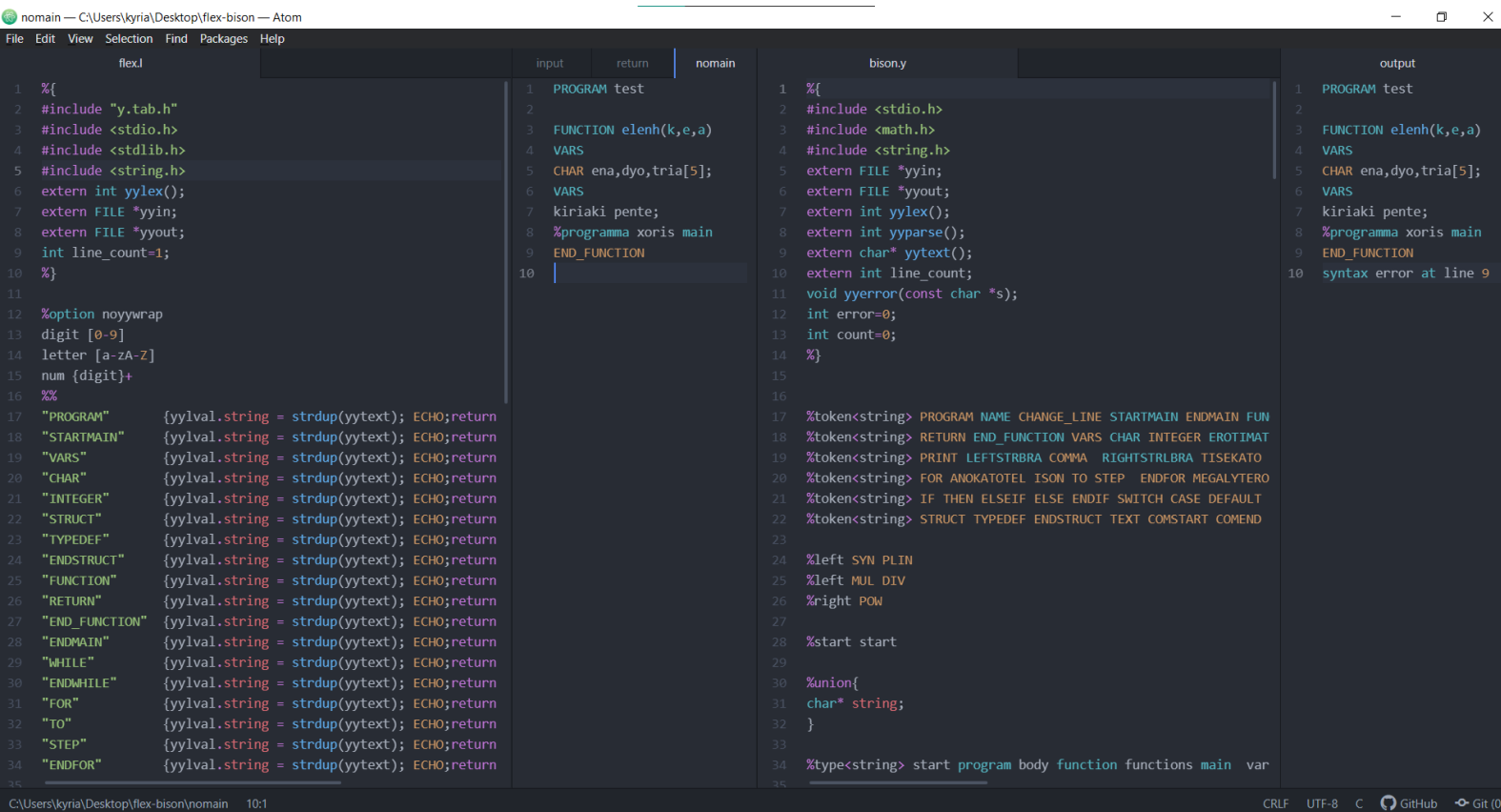
1  %{
2  #include <stdio.h>
3  #include <math.h>
4  #include <string.h>
5  extern FILE *yyin;
6  extern FILE *yyout;
7  extern int yylex();
8  extern int yyparse();
9  extern char* yytext();
10 extern int line_count;
11 void yyerror(const char *s);
12 int error=0;
13 int count=0;
14 %}
15
16
17 %token<string> PROGRAM NAME CHANGE_LINE STARTMAIN ENDMAN FUN
18 %token<string> RETURN END_FUNCTION VARS CHAR INTEGER EROTIMAT
19 %token<string> PRINT LEFTSTRBRA COMMA RIGHTSTRBRA TISEKATO
20 %token<string> FOR ANOKATOTEL ISON TO STEP ENDFOR MEGALYTERO
21 %token<string> IF THEN ELSEIF ELSE ENDIF SWITCH CASE DEFAULT
22 %token<string> STRUCT TYPEDEF ENDSTRUCT TEXT COMSTART COMEND
23
24 %left SYN PLIN
25 %left MUL DIV
26 %right POW
27
28 %start start
29
30 %union{
31 char* string;
32 }
33
34 %type<string> start program body function functions main var
35
```

Εκτέλεση στο Cygwin:



```
kyriaki@LAPTOP-BMJT7V1G /cygdriv
$ ./parser return
error syntax error
errors: 1
kyriaki@LAPTOP-BMJT7V1G /cygdriv
```

3. Στο Input αυτό δεν έχουμε βάλει καθόλου main. Επειδή όμως η main πρέπει να υπάρχει υποχρεωτικά έχουμε error στην γραμμή 9, δηλαδή στην γραμμή που αναμενόταν να υπάρχει η main.



```
1  %{
2  #include "y.tab.h"
3  #include <stdio.h>
4  #include <stdlib.h>
5  #include <string.h>
6  extern int yylex();
7  extern FILE *yyin;
8  extern FILE *yyout;
9  int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17 "PROGRAM" {yylval.string = strdup(yytext); ECHO;return}
18 "STARTMAIN" {yylval.string = strdup(yytext); ECHO;return}
19 "VARS" {yylval.string = strdup(yytext); ECHO;return}
20 "CHAR" {yylval.string = strdup(yytext); ECHO;return}
21 "INTEGER" {yylval.string = strdup(yytext); ECHO;return}
22 "STRUCT" {yylval.string = strdup(yytext); ECHO;return}
23 "TYPEDEF" {yylval.string = strdup(yytext); ECHO;return}
24 "ENDSTRUCT" {yylval.string = strdup(yytext); ECHO;return}
25 "FUNCTION" {yylval.string = strdup(yytext); ECHO;return}
26 "RETURN" {yylval.string = strdup(yytext); ECHO;return}
27 "END_FUNCTION" {yylval.string = strdup(yytext); ECHO;return}
28 "ENDMAIN" {yylval.string = strdup(yytext); ECHO;return}
29 "WHILE" {yylval.string = strdup(yytext); ECHO;return}
30 "ENDWHILE" {yylval.string = strdup(yytext); ECHO;return}
31 "FOR" {yylval.string = strdup(yytext); ECHO;return}
32 "TO" {yylval.string = strdup(yytext); ECHO;return}
33 "STEP" {yylval.string = strdup(yytext); ECHO;return}
34 "ENDFOR" {yylval.string = strdup(yytext); ECHO;return}
35
```

```
1  PROGRAM test
2
3  FUNCTION elenh(k,e,a)
4  VARS
5  CHAR ena,dyo,tria[5];
6  VARS
7  kiriaki pente;
8  %programma xoris main
9  END_FUNCTION
10
```

```
1  %{
2  #include <stdio.h>
3  #include <math.h>
4  #include <string.h>
5  extern FILE *yyin;
6  extern FILE *yyout;
7  extern int yylex();
8  extern int yyparse();
9  extern char* yytext();
10 extern int line_count;
11 void yyerror(const char *s);
12 int error=0;
13 int count=0;
14 %}
15
16
17 %token<string> PROGRAM NAME CHANGE_LINE STARTMAIN ENDMAN FUN
18 %token<string> RETURN END_FUNCTION VARS CHAR INTEGER EROTIMAT
19 %token<string> PRINT LEFTSTRBRA COMMA RIGHTSTRBRA TISEKATO
20 %token<string> FOR ANOKATOTEL ISON TO STEP ENDFOR MEGALYTERO
21 %token<string> IF THEN ELSEIF ELSE ENDIF SWITCH CASE DEFAULT
22 %token<string> STRUCT TYPEDEF ENDSTRUCT TEXT COMSTART COMEND
23
24 %left SYN PLIN
25 %left MUL DIV
26 %right POW
27
28 %start start
29
30 %union{
31 char* string;
32 }
33
34 %type<string> start program body function functions main var
35
```

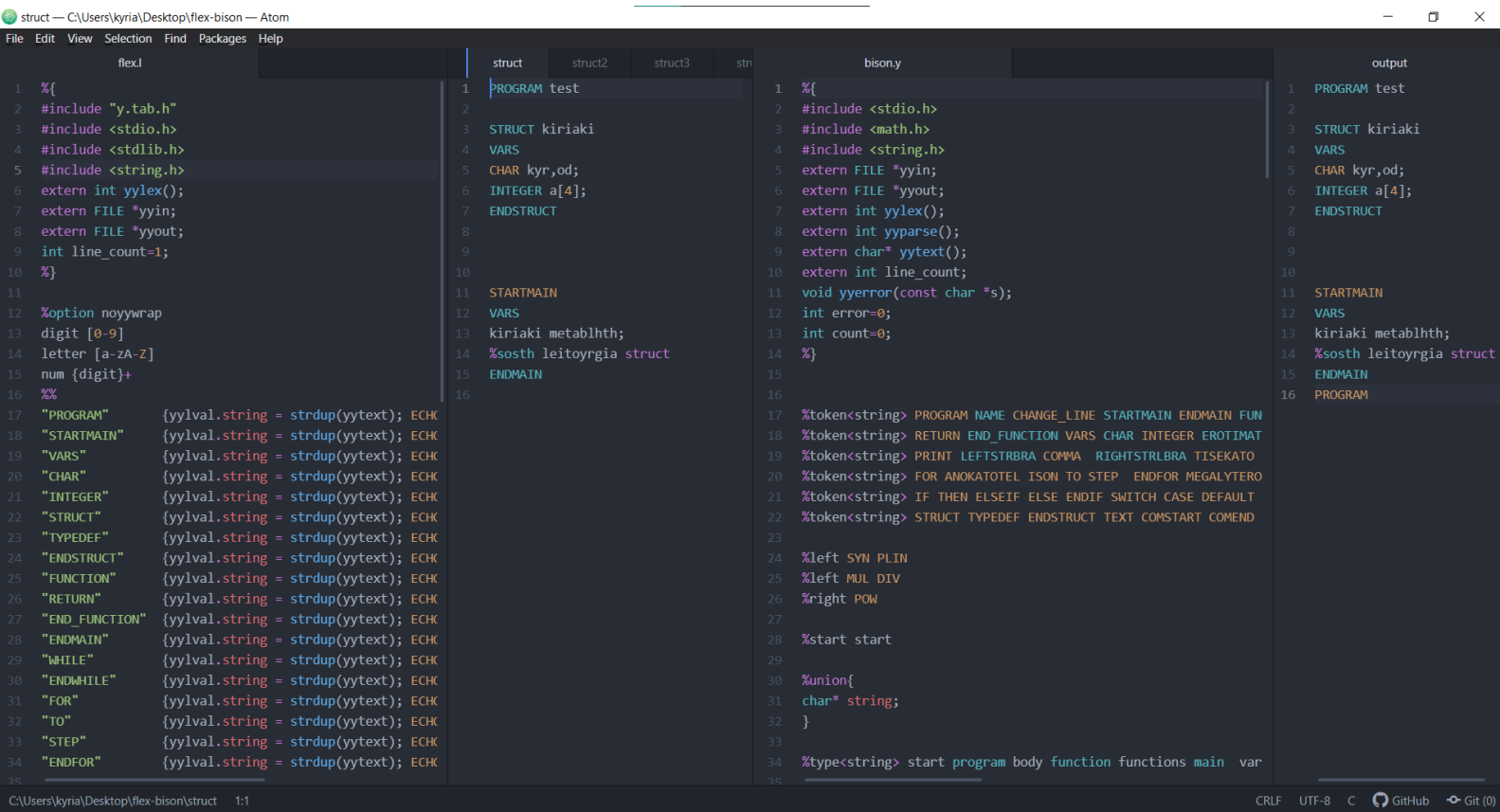
```
1  PROGRAM test
2
3  FUNCTION elenh(k,e,a)
4  VARS
5  CHAR ena,dyo,tria[5];
6  VARS
7  kiriaki pente;
8  %programma xoris main
9  END_FUNCTION
10 syntax error at line 9
```

Εκτέλεση στο Cygwin:

```
kyriaki@LAPTOP-BMJT7V1G /cygdriv
$ ./parser nomain
error syntax error
errors: 1
kyriaki@LAPTOP-BMJT7V1G /cygdriv
$
```

Παραδείγματα από Ερώτημα 2 :

1. Εδώ βλέπουμε την σωστή λειτουργία ενός struct και έτσι στο Output εκτυπώνεται όλη η είσοδος, χωρίς error.



```
struct — C:\Users\kyria\Desktop\flex-bison — Atom
File Edit View Selection Find Packages Help

flex.l
1 %{
2 #include "y.tab.h"
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <string.h>
6 extern int yylex();
7 extern FILE *yyin;
8 extern FILE *yyout;
9 int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17 "PROGRAM" {yyval.string = strdup(yytext); ECHO; }
18 "STARTMAIN" {yyval.string = strdup(yytext); ECHO; }
19 "VARS" {yyval.string = strdup(yytext); ECHO; }
20 "CHAR" {yyval.string = strdup(yytext); ECHO; }
21 "INTEGER" {yyval.string = strdup(yytext); ECHO; }
22 "STRUCT" {yyval.string = strdup(yytext); ECHO; }
23 "TYPEDEF" {yyval.string = strdup(yytext); ECHO; }
24 "ENDSTRUCT" {yyval.string = strdup(yytext); ECHO; }
25 "FUNCTION" {yyval.string = strdup(yytext); ECHO; }
26 "RETURN" {yyval.string = strdup(yytext); ECHO; }
27 "END_FUNCTION" {yyval.string = strdup(yytext); ECHO; }
28 "ENDMAIN" {yyval.string = strdup(yytext); ECHO; }
29 "WHILE" {yyval.string = strdup(yytext); ECHO; }
30 "ENDWHILE" {yyval.string = strdup(yytext); ECHO; }
31 "FOR" {yyval.string = strdup(yytext); ECHO; }
32 "TO" {yyval.string = strdup(yytext); ECHO; }
33 "STEP" {yyval.string = strdup(yytext); ECHO; }
34 "ENDFOR" {yyval.string = strdup(yytext); ECHO; }
35

struct
1 PROGRAM test
2
3 STRUCT kiriaki
4 VARS
5 CHAR kyr,od;
6 INTEGER a[4];
7 ENDSTRUCT
8
9
10
11 STARTMAIN
12 VARS
13 kiriaki metablth;
14 %sosth leitoyrgia struct
15 ENDMAIN
16

bison.y
1 %{
2 #include <stdio.h>
3 #include <math.h>
4 #include <string.h>
5 extern FILE *yyin;
6 extern FILE *yyout;
7 extern int yylex();
8 extern int yyparse();
9 extern char* yytext();
10 extern int line_count;
11 void yyerror(const char *s);
12 int error=0;
13 int count=0;
14 %}
15
16
17 %token<string> PROGRAM NAME CHANGE_LINE STARTMAIN ENDMAIN FUN
18 %token<string> RETURN END_FUNCTION VARS CHAR INTEGER EROTIMAT
19 %token<string> PRINT LEFTSTRBRA COMMA RIGHTSTRBRA TISEKATO
20 %token<string> FOR ANOKATOTEL ISON TO STEP ENDFOR MEGALYTERO
21 %token<string> IF THEN ELSEIF ELSE ENDIF SWITCH CASE DEFAULT
22 %token<string> STRUCT TYPEDEF ENDSTRUCT TEXT COMSTART COMEND
23
24 %left SYN PLIN
25 %left MUL DIV
26 %right POW
27
28 %start start
29
30 %union{
31 char* string;
32 }
33
34 %type<string> start program body function functions main var

output
1 PROGRAM test
2
3 STRUCT kiriaki
4 VARS
5 CHAR kyr,od;
6 INTEGER a[4];
7 ENDSTRUCT
8
9
10
11 STARTMAIN
12 VARS
13 kiriaki metablth;
14 %sosth leitoyrgia struct
15 ENDMAIN
16 PROGRAM
```

Εκτέλεση στο Cygwin:

```
kyriaki@LAPTOP-BMJT7V1G /cyg
$ ./parser struct
errors: 0
kyriaki@LAPTOP-BMJT7V1G /cyg
$ |
```

2. Έχοντας βάλει δύο φορές το ίδιο struct, βλέπουμε ότι εμφανίζεται το κατάλληλο error.

```
struct2 C:\Users\kyria\Desktop\flex-bison — Atom
File Edit View Selection Find Packages Help

flex.l struct struct2 struct3 bison.y output

1 %{
2 #include "y.tab.h"
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <string.h>
6 extern int yylex();
7 extern FILE *yyin;
8 extern FILE *yyout;
9 int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17 "PROGRAM" {yyval.string = strdup(yytext); ECHK
18 "STARTMAIN" {yyval.string = strdup(yytext); ECHK
19 "VARS" {yyval.string = strdup(yytext); ECHK
20 "CHAR" {yyval.string = strdup(yytext); ECHK
21 "INTEGER" {yyval.string = strdup(yytext); ECHK
22 "STRUCT" {yyval.string = strdup(yytext); ECHK
23 "TYPEDEF" {yyval.string = strdup(yytext); ECHK
24 "ENDSTRUCT" {yyval.string = strdup(yytext); ECHK
25 "FUNCTION" {yyval.string = strdup(yytext); ECHK
26 "RETURN" {yyval.string = strdup(yytext); ECHK
27 "END_FUNCTION" {yyval.string = strdup(yytext); ECHK
28 "ENDMAIN" {yyval.string = strdup(yytext); ECHK
29 "WHILE" {yyval.string = strdup(yytext); ECHK
30 "ENDWHILE" {yyval.string = strdup(yytext); ECHK
31 "FOR" {yyval.string = strdup(yytext); ECHK
32 "TO" {yyval.string = strdup(yytext); ECHK
33 "STEP" {yyval.string = strdup(yytext); ECHK
34 "ENDFOR" {yyval.string = strdup(yytext); ECHK
35

1 PROGRAM test
2
3 STRUCT kiriaki
4 VARS
5 CHAR kyr,od;
6 INTEGER a[4];
7 ENDSTRUCT
8
9 STRUCT kiriaki
10 VARS
11 CHAR kyr,od;
12 INTEGER a[4];
13 ENDSTRUCT
14
15
16 STARTMAIN
17 VARS
18 kiriaki metablth;
19 %exoyme dyo fores to idio struct
20 ENDMAIN
21

1 %{
2 #include <stdio.h>
3 #include <math.h>
4 #include <string.h>
5 extern FILE *yyin;
6 extern FILE *yyout;
7 extern int yylex();
8 extern int yyparse();
9 extern char* yytext();
10 extern int line_count;
11 void yyerror(const char *s);
12 int error=0;
13 int count=0;
14 %}
15
16 STARTMAIN
17 %token<string> PROGRAM NAME CHANGE_LINE STARTMAIN ENDMA
18 %token<string> RETURN END_FUNCTION VARS CHAR INTEGER ER
19 %token<string> PRINT LEFTSTRBRA COMMA RIGHTSTRBRA TIS
20 %token<string> FOR ANOKATOTEL ISON TO STEP ENDFOR MEGA
21 %token<string> IF THEN ELSEIF ELSE ENDIF SWITCH CASE DE
22 %token<string> STRUCT TYPEDEF ENDSTRUCT TEXT COMSTART C
23
24 %left SYN PLIN
25 %left MUL DIV
26 %right POW
27
28 %start start
29
30 %union{
31 char* string;
32 }
33
34 %type<string> start program body function functions mai
35

1 PROGRAM test
2
3 STRUCT kiriaki
4 VARS
5 CHAR kyr,od;
6 INTEGER a[4];
7 ENDSTRUCT
8
9 STRUCT kiriaki
10 VARS
11 CHAR kyr,od;
12 INTEGER a[4];
13 ENDSTRUCT
14
15
16 STARTMAIN
17 The struct already exists
18 at line 16
19 VARS
20 kiriaki metablth;
21 %exoyme dyo fores to idio struct
22 ENDMAIN
23 PROGRAM
```

Εκτέλεση στο Cygwin:

```
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c/Users/kyria/Desktop
$ ./parser struct2
error The struct already exists

errors: 1
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c/Users/kyria/Desktop
```

3. Έχουμε βάλει λάθος τύπο στο struct, αφού έχουμε δηλώσει να γίνεται αποδεκτός μόνο ο τύπος “kiriaki”. Έτσι εμφανίζεται το κατάλληλο error στο output.

```
flex.l
1 %{
2 #include "y.tab.h"
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <string.h>
6 extern int yylex();
7 extern FILE *yyin;
8 extern FILE *yyout;
9 int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17
18 "PROGRAM" {yylval.string = strdup; break;}
19 "STARTMAIN" {yylval.string = strdup; break;}
20 "VARS" {yylval.string = strdup; break;}
21 "CHAR" {yylval.string = strdup; break;}
22 "INTEGER" {yylval.string = strdup; break;}
23 "STRUCT" {yylval.string = strdup; break;}
24 "TYPEDEF" {yylval.string = strdup; break;}
25 "ENDSTRUCT" {yylval.string = strdup; break;}
26 "FUNCTION" {yylval.string = strdup; break;}
27 "RETURN" {yylval.string = strdup; break;}
28 "END_FUNCTION" {yylval.string = strdup; break;}
29 "ENDMAIN" {yylval.string = strdup; break;}
30 "WHILE" {yylval.string = strdup; break;}
31 "ENDWHILE" {yylval.string = strdup; break;}
32 "FOR" {yylval.string = strdup; break;}
33 "TO" {yylval.string = strdup; break;}
34 "STEP" {yylval.string = strdup; break;}
35 "ENDFOR" {yylval.string = strdup; break;}
36 %}

bison.y
1 %{
2 #include <stdio.h>
3 #include <math.h>
4 #include <string.h>
5 extern FILE *yyin;
6 extern FILE *yyout;
7 extern int yylex();
8 extern int yyparse();
9 extern char* yytext();
10 extern int line_count;
11 void yyerror(const char *s);
12 int error=0;
13 int count=0;
14 %}
15
16
17 %token<string> PROGRAM NAME CHAR
18 %token<string> RETURN END_FUNC
19 %token<string> PRINT LEFTSTRBRA
20 %token<string> FOR ANOKATOTEL IF
21 %token<string> IF THEN ELSEIF EL
22 %token<string> STRUCT TYPEDEF EN
23
24 %left SYN PLIN
25 %left MUL DIV
26 %right POW
27
28 %start start
29
30 %union{
31 char* string;
32 }
33
34 %type<string> start program body

output
1 PROGRAM test
2
3 STRUCT kir
4 VARS
5 CHAR kyr,od;
6 INTEGER a[4];
7 ENDSTRUCT
8
9
10 STARTMAIN
11 VARS
12 kiriaki metablth;
13 %lathos onoma gia ton struct afoy exoyme dhlosei mono ton typo kiriaki
14 ENDMAIN
15
```

Εκτέλεση στο Cygwin:

```
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c/U
$ ./parser struct3
error Wrong type for the struct

errors: 1
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c/U
$
```

4. Δηλώνουμε μια μεταβλητή τύπου “kir”. Ενώ ο τύπος του struct είναι ο “kiriaki”.
Βλέπουμε το κατάλληλο μήνυμα.

```
1  %{
2  #include "y.tab.h"
3  #include <stdio.h>
4  #include <stdlib.h>
5  #include <string.h>
6  extern int yylex();
7  extern FILE *yyin;
8  extern FILE *yyout;
9  int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17 "PROGRAM" {yyval.string = strdup
18 "STARTMAIN" {yyval.string = strdup
19 "VARS" {yyval.string = strdup
20 "CHAR" {yyval.string = strdup
21 "INTEGER" {yyval.string = strdup
22 "STRUCT" {yyval.string = strdup
23 "TYPEDEF" {yyval.string = strdup
24 "ENDSTRUCT" {yyval.string = strdup
25 "FUNCTION" {yyval.string = strdup
26 "RETURN" {yyval.string = strdup
27 "END_FUNCTION" {yyval.string = strdup
28 "ENDMAIN" {yyval.string = strdup
29 "WHILE" {yyval.string = strdup
30 "ENDWHILE" {yyval.string = strdup
31 "FOR" {yyval.string = strdup
32 "TO" {yyval.string = strdup
33 "STEP" {yyval.string = strdup
34 "ENDFOR" {yyval.string = strdup
35
1  PROGRAM test
2
3  STRUCT kiriaki
4  VARS
5  CHAR kyr,od;
6  INTEGER a[4];
7  ENDSTRUCT
8
9
10
11 STARTMAIN
12 VARS
13 kir metablth;
14 %lathos typos gia thn dhlosh ths metablthhs
15 ENDMAIN
16
17 %{
18 #include <stdio.h>
19 #include <math.h>
20 #include <string.h>
21 extern FILE *yyin;
22 extern FILE *yyout;
23 extern int yylex();
24 extern int yyparse();
25 extern char* yytext();
26 extern int line_count;
27 void yyerror(const char *s);
28 int error=0;
29 int count=0;
30 %}
31
32 %token<string> PROGRAM NAME CHANGE_LINE START
33 %token<string> RETURN_END_FUNCTION VARS CHAR
34 %token<string> PRINT_LEFTSTRBRA COMMA RIGHTS
35 %token<string> FOR ANOKATOTEL ISON TO STEP E
36 %token<string> IF THEN ELSEIF ELSE ENDIF SWIT
37 %token<string> STRUCT TYPEDEF ENDSTRUCT TEXT
38
39 %left SYN PLIN
40 %left MUL DIV
41 %right POW
42
43 %start start
44
45 %union{
46 char* string;
47 }
48
49 %type<string> start program body function fur
50
```

Εκτέλεση στο Cygwin:

```
kyriaki@LAPTOP-BMJT7V1G /cygdrive,/
$ ./parser struct4
error This type does not exist
errors: 1
kyriaki@LAPTOP-BMJT7V1G /cygdrive,/
$
```

5. Σωστή λειτουργία struct με typedef

The screenshot shows a code editor with three tabs: 'flex.l', 'struct_type...', and 'bison.y'. The 'flex.l' tab contains a flex lexer definition for a simple language. The 'struct_type...' tab shows the definition of a struct 'kiriaki' and its use in the 'PROGRAM' rule. The 'bison.y' tab contains a bison parser definition, including the grammar rules and the main function. The 'output' tab shows the result of the compilation, which is a C program that prints the string 'PROGRAM test'.

```
flex.l
1 %{
2 #include "y.tab.h"
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <string.h>
6 extern int yylex();
7 extern FILE *yyin;
8 extern FILE *yyout;
9 int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17 "PROGRAM" {yyval.string = strdup
18 "STARTMAIN" {yyval.string = strdup
19 "VARS" {yyval.string = strdup
20 "CHAR" {yyval.string = strdup
21 "INTEGER" {yyval.string = strdup
22 "STRUCT" {yyval.string = strdup
23 "TYPEDEF" {yyval.string = strdup
24 "ENDSTRUCT" {yyval.string = strdup
25 "FUNCTION" {yyval.string = strdup
26 "RETURN" {yyval.string = strdup
27 "END_FUNCTION" {yyval.string = strdup
28 "ENDMAIN" {yyval.string = strdup
29 "WHILE" {yyval.string = strdup
30 "ENDWHILE" {yyval.string = strdup
31 "FOR" {yyval.string = strdup
32 "TO" {yyval.string = strdup
33 "STEP" {yyval.string = strdup
34 "ENDFOR" {yyval.string = strdup
35

struct_type...
1 PROGRAM test
2
3 TYPEDEF STRUCT kiriaki
4 VARS
5 CHAR kyr,od;
6 INTEGER a[4];
7 /*typedef struct*/
8 kiriaki ENDSTRUCT
9
10
11
12 STARTMAIN
13 VARS
14 kiriaki metablth;
15 %soth leitoyrgia struct
16 ENDMAIN
17

bison.y
1 %{
2 #include <stdio.h>
3 #include <math.h>
4 #include <string.h>
5 extern FILE *yyin;
6 extern FILE *yyout;
7 extern int yylex();
8 extern int yyparse();
9 extern char* yytext();
10 extern int line_count;
11 void yyerror(const char *s);
12 int error=0;
13 int count=0;
14 %}
15
16
17 %token<string> PROGRAM NAME CHANGE_LINE STARTMAIN ENDMAIN FUI
18 %token<string> RETURN END_FUNCTION VARS CHAR INTEGER EROTIMA
19 %token<string> PRINT LEFTSTRBRA COMMA RIGHTSTRBRA TISEKATO
20 %token<string> FOR ANOKATOTEL ISON TO STEP ENDFOR MEGALYTERI
21 %token<string> IF THEN ELSEIF ELSE ENDIF SWITCH CASE DEFAULT
22 %token<string> STRUCT TYPEDEF ENDSTRUCT TEXT COMSTART COMEND
23
24 %left SYN PLIN
25 %left MUL DIV
26 %right POW
27
28 %start start
29
30 %union{
31 char* string;
32 }
33
34 %type<string> start program body function functions main vai

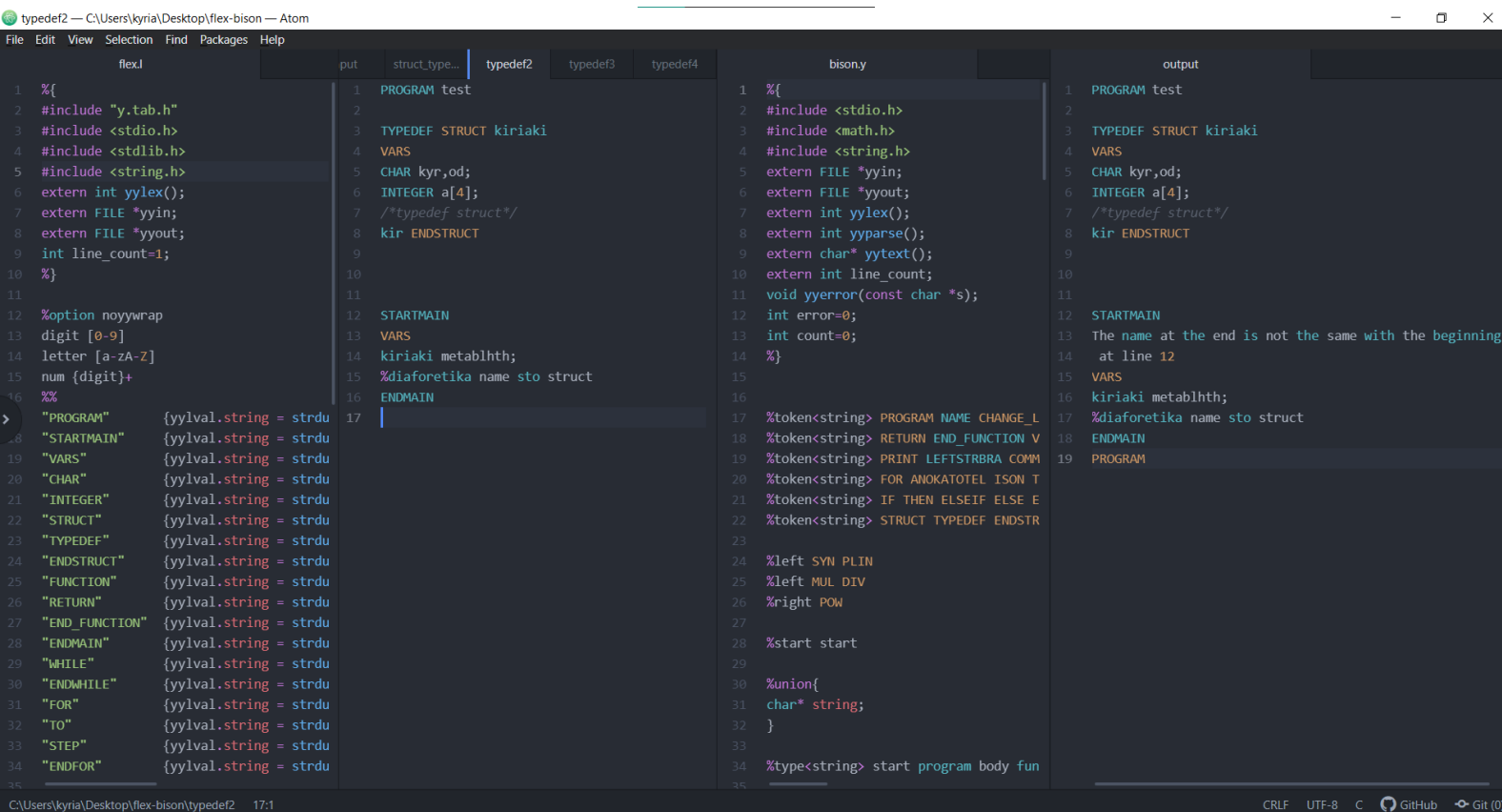
output
1 PROGRAM test
2
3 TYPEDEF STRUCT kiriaki
4 VARS
5 CHAR kyr,od;
6 INTEGER a[4];
7 /*typedef struct*/
8 kiriaki ENDSTRUCT
9
10
11
12 STARTMAIN
13 VARS
14 kiriaki metablth;
15 %soth leitoyrgia struct
16 ENDMAIN
17 PROGRAM
```

Εκτέλεση στο Cygwin:

The terminal window shows the execution of the parser in Cygwin. The user runs the command './parser struct_typedef' and the output is 'errors: 0'.

```
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c
$ ./parser struct_typedef
errors: 0
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c
```


6. Έχουμε βάλει διαφορετικό όνομα στην αρχή και το τέλος του struct. Εμφανίζεται το κατάλληλο error.



```
typedef2 — C:\Users\kyria\Desktop\flex-bison — Atom
File Edit View Selection Find Packages Help

flex.l put struct_type... typedef2 typedef3 typedef4 bison.y output

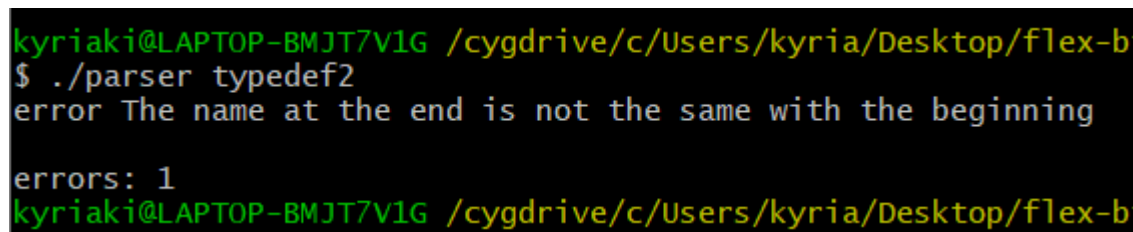
1  %{
2  #include "y.tab.h"
3  #include <stdio.h>
4  #include <stdlib.h>
5  #include <string.h>
6  extern int yylex();
7  extern FILE *yyin;
8  extern FILE *yyout;
9  int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17
18 "PROGRAM" {yyval.string = strdup("PROGRAM");}
19 "STARTMAIN" {yyval.string = strdup("STARTMAIN");}
20 "VARS" {yyval.string = strdup("VARS");}
21 "CHAR" {yyval.string = strdup("CHAR");}
22 "INTEGER" {yyval.string = strdup("INTEGER");}
23 "STRUCT" {yyval.string = strdup("STRUCT");}
24 "TYPEDEF" {yyval.string = strdup("TYPEDEF");}
25 "ENDSTRUCT" {yyval.string = strdup("ENDSTRUCT");}
26 "FUNCTION" {yyval.string = strdup("FUNCTION");}
27 "RETURN" {yyval.string = strdup("RETURN");}
28 "END_FUNCTION" {yyval.string = strdup("END_FUNCTION");}
29 "ENDMAIN" {yyval.string = strdup("ENDMAIN");}
30 "WHILE" {yyval.string = strdup("WHILE");}
31 "ENDWHILE" {yyval.string = strdup("ENDWHILE");}
32 "FOR" {yyval.string = strdup("FOR");}
33 "TO" {yyval.string = strdup("TO");}
34 "STEP" {yyval.string = strdup("STEP");}
35 "ENDFOR" {yyval.string = strdup("ENDFOR");}

1 PROGRAM test
2
3 TYPEDEF STRUCT kiriaki
4 VARS
5 CHAR kyr,od;
6 INTEGER a[4];
7 /*typedef struct*/
8 kir ENDSTRUCT
9
10
11
12 STARTMAIN
13 VARS
14 kiriaki metablth;
15 %diaforetika name sto struct
16 ENDMAIN
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35

1  %{
2  #include <stdio.h>
3  #include <math.h>
4  #include <string.h>
5  extern FILE *yyin;
6  extern FILE *yyout;
7  extern int yylex();
8  extern int yyparse();
9  extern char* yytext();
10 extern int line_count;
11 void yyerror(const char *s);
12 int error=0;
13 int count=0;
14 %}
15
16
17 %token<string> PROGRAM NAME CHANGE_L
18 %token<string> RETURN END_FUNCTION V
19 %token<string> PRINT LEFTSTRBRA COMM
20 %token<string> FOR ANOKATOTEL ISON T
21 %token<string> IF THEN ELSEIF ELSE E
22 %token<string> STRUCT TYPEDEF ENDSTR
23
24 %left SYN PLIN
25 %left MUL DIV
26 %right POW
27
28 %start start
29
30 %union{
31 char* string;
32 }
33
34 %type<string> start program body fun

1 PROGRAM test
2
3 TYPEDEF STRUCT kiriaki
4 VARS
5 CHAR kyr,od;
6 INTEGER a[4];
7 /*typedef struct*/
8 kir ENDSTRUCT
9
10
11
12 STARTMAIN
13 The name at the end is not the same with the beginning
14 at line 12
15 VARS
16 kiriaki metablth;
17 %diaforetika name sto struct
18 ENDMAIN
19 PROGRAM
```

Εκτέλεση στο Cygwin:

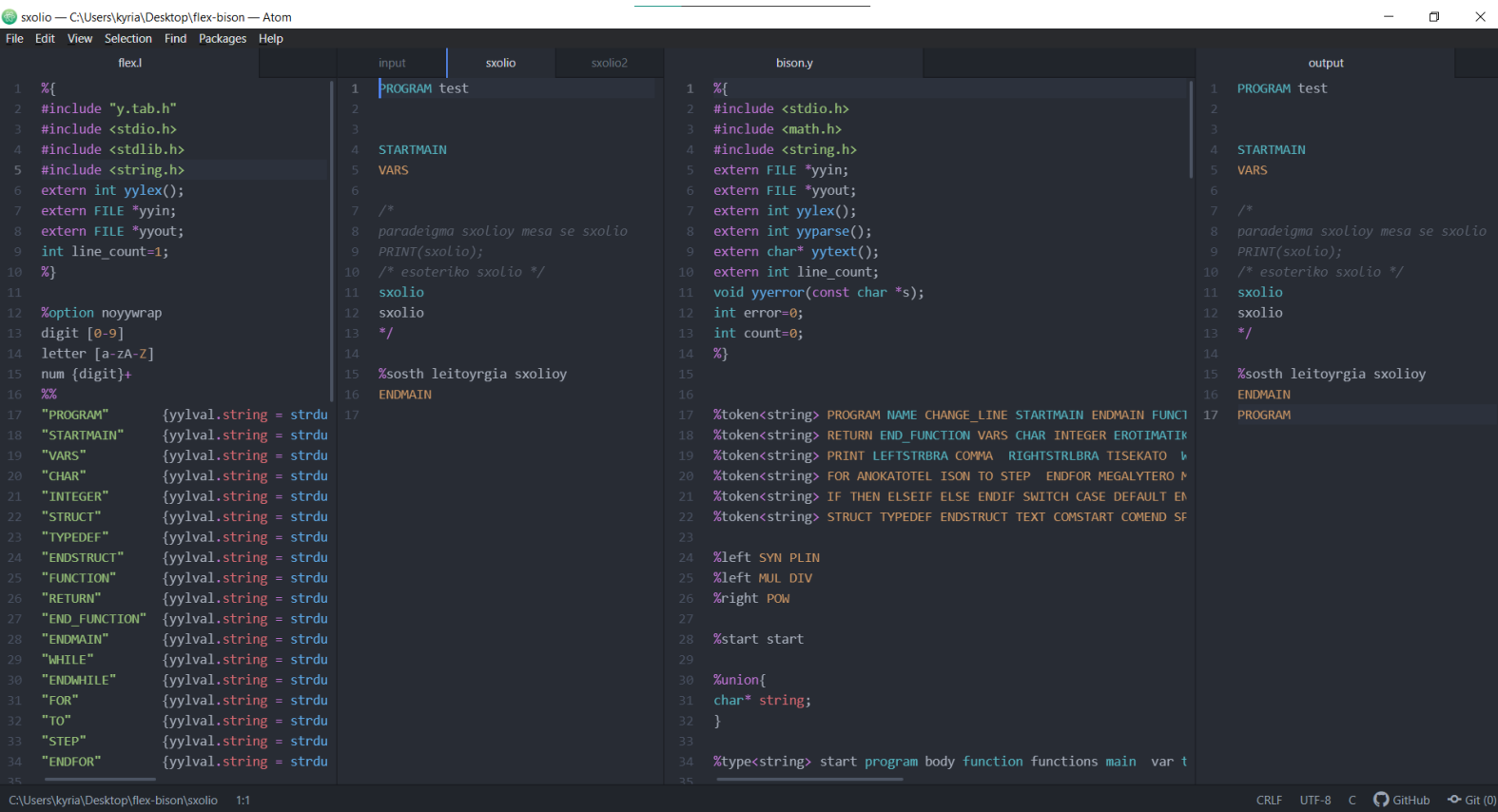


```
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c/Users/kyria/Desktop/flex-b
$ ./parser typedef2
error The name at the end is not the same with the beginning
errors: 1
kyriaki@LAPTOP-BMJT7V1G /cygdrive/c/Users/kyria/Desktop/flex-b
```

Οι υπόλοιπες περιπτώσεις με τα errors στα typedef είναι ίδιες με των απλών structs παραπάνω.

Παραδείγματα από Ερώτημα 4 :

1. Σωστή λειτουργία του σχολίου της μορφής `/* σχόλιο */`



```
flex.l
1 %{
2 #include "y.tab.h"
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <string.h>
6 extern int yylex();
7 extern FILE *yyin;
8 extern FILE *yyout;
9 int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17 "PROGRAM" {yylval.string = strdup("PROGRAM test");}
18 "STARTMAIN" {yylval.string = strdup("STARTMAIN");}
19 "VARS" {yylval.string = strdup("VARS");}
20 "CHAR" {yylval.string = strdup("CHAR");}
21 "INTEGER" {yylval.string = strdup("INTEGER");}
22 "STRUCT" {yylval.string = strdup("STRUCT");}
23 "TYPEDEF" {yylval.string = strdup("TYPEDEF");}
24 "ENDSTRUCT" {yylval.string = strdup("ENDSTRUCT");}
25 "FUNCTION" {yylval.string = strdup("FUNCTION");}
26 "RETURN" {yylval.string = strdup("RETURN");}
27 "END_FUNCTION" {yylval.string = strdup("END_FUNCTION");}
28 "ENDMAIN" {yylval.string = strdup("ENDMAIN");}
29 "WHILE" {yylval.string = strdup("WHILE");}
30 "ENDWHILE" {yylval.string = strdup("ENDWHILE");}
31 "FOR" {yylval.string = strdup("FOR");}
32 "TO" {yylval.string = strdup("TO");}
33 "STEP" {yylval.string = strdup("STEP");}
34 "ENDFOR" {yylval.string = strdup("ENDFOR");}
35

input
1 PROGRAM test
2
3
4 STARTMAIN
5 VARS
6
7 /*
8 paradeigma sxoliov mesa se sxolio
9 PRINT(sxolio);
10 /* esoteriko sxolio */
11 sxolio
12 sxolio
13 */
14
15 %sosth leitoyrgia sxoliov
16 ENDMAN
17

sxolio
1 PROGRAM test
2
3
4 STARTMAIN
5 VARS
6
7 /*
8 paradeigma sxoliov mesa se sxolio
9 PRINT(sxolio);
10 /* esoteriko sxolio */
11 sxolio
12 sxolio
13 */
14
15 %sosth leitoyrgia sxoliov
16 ENDMAN
17 PROGRAM

bison.y
1 %{
2 #include <stdio.h>
3 #include <math.h>
4 #include <string.h>
5 extern FILE *yyin;
6 extern FILE *yyout;
7 extern int yylex();
8 extern int yyparse();
9 extern char* yytext();
10 extern int line_count;
11 void yyerror(const char *s);
12 int error=0;
13 int count=0;
14 %}
15
16 %token<string> PROGRAM NAME CHANGE_LINE STARTMAIN ENDMAN FUNCT
17 %token<string> RETURN END_FUNCTION VARS CHAR INTEGER EROTIMATI
18 %token<string> PRINT LEFTSTRBRA COMMA RIGHTSTRBRA TISEKATO
19 %token<string> FOR ANOKATOTEL ISON TO STEP ENDFOR MEGALYTERO
20 %token<string> IF THEN ELSEIF ELSE ENDIF SWITCH CASE DEFAULT EN
21 %token<string> STRUCT TYPEDEF ENDSTRUCT TEXT COMSTART COMEND SF
22
23
24 %left SYN PLIN
25 %left MUL DIV
26 %right POW
27
28 %start start
29
30 %union{
31 char* string;
32 }
33
34 %type<string> start program body function functions main var t
35
```

Εκτέλεση στο Cygwin :

```
kyriaki@LAPTOP-BMJT7V1G /
$ ./parser sxolio
errors: 0
kyriaki@LAPTOP-BMJT7V1G /
$
```

2. Έχουμε αφαιρέσει το σύμβολο `*/` που κλείνει το σχόλιο και βλέπουμε το error στη γραμμή 14, καθώς κάπου εκεί περιμένει να βρει το σύμβολο.

```
flex.l
1 %{
2 #include "y.tab.h"
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <string.h>
6 extern int yylex();
7 extern FILE *yyin;
8 extern FILE *yyout;
9 int line_count=1;
10 %}
11
12 %option noyywrap
13 digit [0-9]
14 letter [a-zA-Z]
15 num {digit}+
16 %%
17 "PROGRAM" {yylval.string = strdup
18 "STARTMAIN" {yylval.string = strdup
19 "VARS" {yylval.string = strdup
20 "CHAR" {yylval.string = strdup
21 "INTEGER" {yylval.string = strdup
22 "STRUCT" {yylval.string = strdup
23 "TYPEDEF" {yylval.string = strdup
24 "ENDSTRUCT" {yylval.string = strdup
25 "FUNCTION" {yylval.string = strdup
26 "RETURN" {yylval.string = strdup
27 "END_FUNCTION" {yylval.string = strdup
28 "ENDMAIN" {yylval.string = strdup
29 "WHILE" {yylval.string = strdup
30 "ENDWHILE" {yylval.string = strdup
31 "FOR" {yylval.string = strdup
32 "TO" {yylval.string = strdup
33 "STEP" {yylval.string = strdup
34 "ENDFOR" {yylval.string = strdup
35
input
1 PROGRAM test
2
3
4 STARTMAIN
5 VARS
6
7 /*
8 paradeigma la8os sxolioy
9 leipei to teliko symbolo gia na kleisei to sxolio
10 PRINT(sxolio);
11 sxolio
12 sxolio
13
14 ENDMAIN
15
bison.y
1 %{
2 #include <stdio.h>
3 #include <math.h>
4 #include <string.h>
5 extern FILE *yyin;
6 extern FILE *yyout;
7 extern int yylex();
8 extern int yyparse();
9 extern char* yytext();
10 extern int line_count;
11 void yyerror(const char *s);
12 int error=0;
13 int count=0;
14 %}
15
16
17 %token<string> PROGRAM NAME CHANGE_L:
18 %token<string> RETURN END_FUNCTION V:
19 %token<string> PRINT LEFTSTRBRA COMM:
20 %token<string> FOR ANOKATOTEL ISON T:
21 %token<string> IF THEN ELSEIF ELSE E:
22 %token<string> STRUCT TYPEDEF ENDSTRU:
23
24 %left SVN PLIN
25 %left MUL DIV
26 %right POW
27
28 %start start
29
30 %union{
31 char* string;
32 }
33
34 %type<string> start program body func
35
output
1 PROGRAM test
2
3
4 STARTMAIN
5 VARS
6
7 /*
8 paradeigma la8os sxolioy
9 leipei to teliko symbolo gia na kleisei to sxolio
10 PRINT(sxolio);
11 sxolio
12 sxolio
13
14 ENDMAIN
15 syntax error at line 14
```

Εκτέλεση στο Cygwin:

```
kyriaki@LAPTOP-BMJT7V1G /cygdr
$ ./parser sxolio2
error syntax error
errors: 1
kyriaki@LAPTOP-BMJT7V1G /cygdr
$
```

- Έχουμε φτιάξει ένα input με όλες τις εντολές που ζητούνται, το οποίο λειτουργεί σωστά χωρίς errors.

The screenshot shows the Atom editor with four files open: `flex.l`, `input`, `bison.y`, and `output`. The `input` file is the active tab and contains the following code:

```

1  PROGRAM test
2
3  STRUCT kiriaki
4  VARS
5  CHAR kyr,od;
6  INTEGER a[4];
7  ENDSTRUCT
8
9
10 FUNCTION elenh(k,e,a)
11 VARS
12 CHAR ena,dyo,tria[5];
13 VARS
14 kiriaki pente;
15 %sxolio
16 RETURN dyo
17 END_FUNCTION
18
19
20 FUNCTION arhodia(yooo)
21 VARS
22 INTEGER hfg;
23 %sxolio
24 WHILE(1kkfk == fkhv)
25 %sxolio
26 ENDWHILE
27 RETURN jfkf
28 END_FUNCTION
29
30
31 STARTMAIN
32 VARS
33 INTEGER tria;
34
35 %fclose

```

Εκτέλεση στο Cygwin:

The screenshot shows a Cygwin terminal window with the following output:

```

kyriaki@LAPTOP-BM:
$ ./parser input
errors: 0
kyriaki@LAPTOP-BM:

```

Το input δεν χωράει ολόκληρο στο screenshot. Το στέλνουμε λοιπόν μαζί με τα υπόλοιπα αρχεία.

Προσπάθεια ερωτήματος 3 για τον έλεγχο σωστής δήλωσης των ονομάτων που χρησιμοποιούνται σε συναρτήσεις.

```
function: FUNCTION NAME LEFTBRA list RIGHTBRA lines var commands{char* x; int i; int found;

    for(i = 0; i < 4; i++){if(strcmp($2, x) == 0 ){ found=1;}else{}}

    if (found=0) {$2 = x;}

    else if (found>0) {yyerror("The function already exists");}

    ;}
```

Επεξήγηση:

Ορίζουμε έναν πίνακα x από string. Τρέχουμε μια for για τα στοιχεία του πίνακα. Με μια if ελέγχουμε αν το όνομα που δίνουμε υπάρχει ήδη στον πίνακα με την συνάρτηση strcmp() και αν υπάρχει τότε βάζουμε την τιμή 1 στην μεταβλητή found, την οποία χρησιμοποιούμε από κάτω. Στο else δεν γίνεται τίποτα. Με μια if ελέγχουμε αν η μεταβλητή found έχει την τιμή 0. Αν έχει την τιμή 0 σημαίνει ότι δεν βρέθηκε στον πίνακα τέτοιο όνομα, οπότε το εισάγουμε εμείς στον πίνακα. Αλλιώς αν το found >0 , δηλαδή βρέθηκε στον πίνακα τότε καλούμε την yyerror() και βγάζει το κατάλληλο μήνυμα ότι υπάρχει ήδη συνάρτηση με τέτοιο όνομα.

Ομοίως θα γινόταν και ο έλεγχος στον κανόνα με την δήλωση των μεταβλητών στο πρόγραμμα. Θα φτιάχνεται ένας πίνακας που θα αποθηκεύει τα ονόματα όλων των μεταβλητών του προγράμματος και με συγκρίσεις θα καταλήγουμε στο αν έχουν ξαναδηλωθεί οι μεταβλητές και θα παίρνουμε τα κατάλληλα errors.

Με την υλοποίηση αυτή έχουμε segmentation fault και λόγω έλλειψης χρόνου δεν καταφέραμε να το διορθώσουμε. Ωστόσο το παραθέτουμε σαν ιδέα.

Σχόλια και Παραδοχές:

- Το τρίτο ερώτημα δεν τρέχει και δεν είναι ολοκληρωμένο, αλλά η προσπάθεια υλοποίησης φαίνεται από πάνω.
- Στο δεύτερο ερώτημα αναγνωρίζεται μόνο ένας τύπος, για ευκολία.
- Η έξοδος εκτυπώνεται σε ξεχωριστό αρχείο με όνομα output.
- Δημιουργήθηκαν πολλές εισοδοι για τα screenshot δείχνοντας error και μη. Τα στέλνουμε όλα σε περίπτωση που θέλετε να τα χρησιμοποιήσετε. Το βασικό input βρίσκεται μόνο του, ενώ όλα τα υπόλοιπα input για τα παραδείγματα βρίσκονται όλα μαζί σε έναν ξεχωριστό φάκελο μέσα στο zip.