



**ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ**

**ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ**

**ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ  
& ΠΛΗΡΟΦΟΡΙΚΗΣ**

**ΕΡΓΑΣΙΑ FLEX & BISON ΣΤΟ ΜΑΘΗΜΑ  
ΑΡΧΕΣ ΓΛΩΣΣΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΚΑΙ ΜΕΤΑΦΡΑΣΤΩΝ**

### **Μέλη ομάδας:**

- Αβραμόπουλος Γεώργιος - 1070772 – Έτος 3<sup>ο</sup> – [up1070772@upnet.gr](mailto:up1070772@upnet.gr)
- Κοντογιώργης Νικόλαος – 1070922 – Έτος 3<sup>ο</sup> – [up1070922@upnet.gr](mailto:up1070922@upnet.gr)
- Κωνσταντοπούλου Νεφέλη – 1059628 – Έτος 4<sup>ο</sup> – [up1059628@upnet.gr](mailto:up1059628@upnet.gr)
- Μαραζιώτης Ορέστης – 1064028 – Έτος 4<sup>ο</sup> – [up1064028@upnet.com](mailto:up1064028@upnet.com)

### **Backus-Naur Form notation (BNF):**

```
<A_PARENTHESIS> ::= "("
<D_PARENTHESIS> ::= ")"
<KOMMA> ::= ","
<Q_MARK> ::= ";"
<A_BRACKET> ::= "["
<D_BRACKET> ::= "]"
<ADD> ::= "+"
<SUBTRACT> ::= "-"
<POWER_OF> ::= "^"
<MULTIPLY> ::= "*"
<DIVIDE> ::= "/"
<EQUALS> ::= "="
<BIGGER_THAN> ::= "<"
<SMALLER_THAN> ::= ">"
<LOG_EQUALS> ::= "=="
<NOT_EQUAL> ::= "!="
<COLON> ::= ":"
<ARIST> ::= "'"

<WHILE> ::= "while"
<ENDWHILE> ::= "end_while"
<AND> ::= "and"
<OR> ::= "or"
<FOR> ::= "for"
<COUNTER> ::= "counter"
<TO> ::= "to"
<STEP> ::= "step"
<ENDFOR> ::= "end_for"
<IF> ::= "if"
<THEN> ::= "then"
<ELSE> ::= "else"
<ELSEIF> ::= "else_if"
<ENDIF> ::= "end_if"
<SWITCH> ::= "switch"
<CASE> ::= "case"
<DEFAULT> ::= "default"
<END_SWITCH> ::= "end_switch"
<PRINT> ::= "print"
<BREAK> ::= "break"
<PROGRAM> ::= "program"
<RETURN> ::= "return"
```

```

<THETIKOS_AKER> ::= "thetikos_aker"
<FLOAT> ::= "myfloat"
<STARTMAIN> ::= "start_main"
<ENDMAIN> ::= "end_main"
<FUNCTION> ::= "function"
<VARS> ::= "vars"
<END_FUNCTION> ::= "end_function"
<STRUCT> ::= "struct"
<ENDSTRUCT> ::= "endstruct"
<TYPEDEF> ::= "typedef"
<CHAR> ::= "mychar"
<INTEGER> ::= "myinteger"

<LETTER> ::= a | b | c | d | e | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z |
A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z
<DIGIT> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<STRING> ::= <LETTER> |
             <LETTER><STRING> |
             <DIGIT><STRING>

<main> ::= <program><list_structs><list_function><function><startmain>
          | <program><list_structs><startmain>
          | <program><startmain>
          | <program>

<program> ::= <PROGRAM><STRING>

<list_structs> ::= <list_structs><structs>
                  | <structs>

<structs> ::= <STRUCT><STRING><vars><ENDSTRUCT>
              | <STRUCT><STRING><vars><STRING><ENDSTRUCT>

<vars> ::= <VARS><vars_list>

<vars_list> ::= <CHAR><string_list><Q_MARK><vars_list>
               | <INTEGER><string_list><Q_MARK><vars_list>
               | <CHAR><string_list><Q_MARK>
               | <INTEGER><string_list><Q_MARK>

<string_list> ::= <STRING><A_BRACKET><THETIKOS_AKER><D_BRACKET><KOMMA>
                  <string_list>
                  | <STRING><KOMMA><string_list>
                  | <STRING><A_BRACKET><THETIKOS_AKER><D_BRACKET>
                  | <STRING>

<list_function> ::= <list_function><function>
                   | <function>

```

```

<function> ::= <FUNCTION><STRING><A_PARENTHESEI><string_list><D_PARENTHESEI><vars>
    <entoles><return><END_FUNCTION>
    | <FUNCTION><STRING><A_PARENTHESEI><D_PARENTHESEI><vars><entoles><return>
    <END_FUNCTION>
    | <FUNCTION><STRING><A_PARENTHESEI><string_list><D_PARENTHESEI><entoles>
    <return><END_FUNCTION>
    | <FUNCTION><STRING><A_PARENTHESEI><D_PARENTHESEI><entoles><return>
    <END_FUNCTION>

```

```

<return> ::= <RETURN><THETIKOS_AKER><Q_MARK>
    | <RETURN><STRING><Q_MARK>

```

```

<list_entoles> ::= <list_entoles><entoles>
    | <entoles>

```

```

<entoles> ::= <anathesi>
    | <while>
    | <for>
    | <if>
    | <switch>
    | <print>
    | <break>

```

```

<print> ::= <PRINT><A_PARENTHESEI><ARIST><STRING><ARIST><D_PARENTHESEI><Q_MARK>
    | <PRINT><A_PARENTHESEI><ARIST><STRING><ARIST><KOMMA><string_list>
    <D_PARENTHESEI><Q_MARK>

```

```

<break> ::= <BREAK><Q_MARK>

```

```

<anathesi> ::= <STRING><EQUALS><prajeis><Q_MARK>

```

```

<prajeis> ::= <prajeis><ADD><prajeis>
    | <prajeis><SUBTRACT><prajeis>
    | <prajeis><MULTIPLY><prajeis>
    | <prajeis><DIVIDE><prajeis>
    | <prajeis><POWER_OF><prajeis>
    | <A_PARENTHESEI><prajeis><D_PARENTHESEI>
    | <STRING><ADD><prajeis>
    | <STRING><A_PARENTHESEI><string_list><D_PARENTHESEI><ADD><prajeis>
    | <THETIKOS_AKER><ADD><prajeis>
    | <STRING><SUBTRACT><prajeis>
    | <THETIKOS_AKER><SUBTRACT><prajeis>
    | <STRING><MULTIPLY><prajeis>
    | <STRING><A_PARENTHESEI><string_list><D_PARENTHESEI><MULTIPLY><prajeis>
    | <THETIKOS_AKER><MULTIPLY><prajeis>
    | <STRING><DIVIDE><prajeis>
    | <STRING><A_PARENTHESEI><string_list><D_PARENTHESEI><DIVIDE><prajeis>
    | <THETIKOS_AKER><DIVIDE><prajeis>
    | <STRING><POWER_OF><prajeis>
    | <STRING><A_PARENTHESEI><string_list><D_PARENTHESEI><POWER_OF><prajeis>
    | <THETIKOS_AKER><POWER_OF><prajeis>
    | <STRING><A_PARENTHESEI><string_list><D_PARENTHESEI>

```

```

    | <STRING><A_PARENTHESI><D_PARENTHESI>
    | <THETIKOS_AKER>
<while> ::= <WHILE><condition><list_entoles><ENDWHILE>

<for> ::= <FOR><STRING><COLON><EQUALS><THETIKOS_AKER><TO><THETIKOS_AKER>
    <STEP><THETIKOS_AKER><list_entoles><ENDFOR>

<list_elseif> ::= <list_elseif><elseif>
    | <elseif>

<elseif> ::= <ELSEIF><condition><list_entoles>

<if> ::= <IF><condition><THEN><list_entoles><ENDIF>
    | <IF><condition><THEN><list_entoles><ELSE><list_entoles><ENDIF>
    | <IF><condition><THEN><list_entoles><list_elseif><ENDIF>
    | <IF><condition><THEN><list_entoles><list_elseif><ELSE><list_entoles><ENDIF>

<list_case> ::= <list_case><case>
    | <case>

<case> ::= <CASE><prajeis><COLON><list_entoles>

<switch> ::= <SWITCH><prajeis><list_case><DEFAULT><COLON><list_entoles><ENDSWITCH>
    | <SWITCH><prajeis><list_case><ENDSWITCH>

<and_or> ::= <AND>
    | <AND><and_or>
    | <OR>
    | <OR><and_or>

<condition> ::= <A_PARENTHESI><STRING><BIGGER_THAN><THETIKOS_AKER>
    <D_PARENTHESI>
    | <A_PARENTHESI><STRING><BIGGER_THAN><THETIKOS_AKER>
    <D_PARENTHESI><and_or><condition>
    | <A_PARENTHESI><STRING><BIGGER_THAN><STRING><D_PARENTHESI>
    | <A_PARENTHESI><STRING><BIGGER_THAN><STRING><D_PARENTHESI>
    <and_or><condition>
    | <A_PARENTHESI><STRING><SMALLER_THAN><THETIKOS_AKER>
    <D_PARENTHESI>
    | <A_PARENTHESI><STRING><SMALLER_THAN><THETIKOS_AKER>
    <D_PARENTHESI><and_or><condition>
    | <A_PARENTHESI><STRING><SMALLER_THAN><STRING><D_PARENTHESI>
    | <A_PARENTHESI><STRING><SMALLER_THAN><STRING><D_PARENTHESI>
    <and_or><condition>
    | <A_PARENTHESI><STRING><LOG_EQUALS><THETIKOS_AKER><D_PARENTHESI>
    | <A_PARENTHESI><STRING><LOG_EQUALS><THETIKOS_AKER><D_PARENTHESI>
    <and_or><condition>
    | <A_PARENTHESI><STRING><LOG_EQUALS><STRING><D_PARENTHESI>
    | <A_PARENTHESI><STRING><LOG_EQUALS><STRING><D_PARENTHESI>
    <and_or><condition>
    | <A_PARENTHESI><STRING><NOT_EQUAL><THETIKOS_AKER><D_PARENTHESI>
    | <A_PARENTHESI><STRING><NOT_EQUAL><THETIKOS_AKER><D_PARENTHESI>
    <and_or><condition>
    | <A_PARENTHESI><STRING><NOT_EQUAL><STRING><D_PARENTHESI>

```

```
| <A_PARENTHESI><STRING><NOT_EQUAL><STRING><D_PARENTHESI>
<and_or><condition>
<A_PARENTHESI><condition><D_PARENTHESI>
```

```
<startmain> ::= <STARTMAIN><vars><list_entoles><ENDMAIN>
| <STARTMAIN><list_entoles><ENDMAIN>
```

**Περιεχόμενα αρχείου εισόδου .txt (ορθής λειτουργίας):**

```
PROGRAM orestis
TYPEDEF STRUCT vv
VARS
INTEGER a;
vv ENDSTRUCT
%comment
TYPEDEF STRUCT vv2
VARS
INTEGER bb;
vv2 ENDSTRUCT
TYPEDEF STRUCT vv3
VARS
INTEGER Aaa, b, pinakas[3];
vv3 ENDSTRUCT
FUNCTION orestis()
RETURN 0;
END_FUNCTION
/*
comment lols *****
% comment in comment
*/
STARTMAIN
VARS
INTEGER A, B;
CHAR C, D;
INTEGER abc, i;
FOR i := 1 TO 2 STEP 3
FOR i := 1 TO 2 STEP 3 %comment a = b
a = orestis(a) + orestis(a, vasda);
ENDFOR
a = b;
IF (bb==0) THEN
WHILE (bb>1) AND (bb<100)
b = b + 1;
ENDWHILE
ELSEIF (orestis(bb)==100)
WHILE (bb<20)
PRINT('bb', b);
ENDWHILE
ELSE
SWITCH (bb)
CASE (a):
a = b + bb;
CASE (bb):
```

```

        PRINT('bb', bb);
    DEFAULT:
        PRINT('default');
    ENDSWITCH
ENDIF
ENDFOR

ENDMAIN

```

**Screenshots λειτουργίας (στο αριστερό μέρος βλέπουμε το txt αρχείο με το τρέχον παράδειγμα ενώ στο δεξί μέρος βλέπουμε το terminal με το output του bison):**

- Γενική περίπτωση ορθής δήλωσης μεταβλητών, structs και functions.

[illegible]

- Παράδειγμα όπου η μεταβλητή c δεν έχει δηλωθεί και σε ποιά line υπάρχει.

```

1 PROGRAM TEST2
2
3 TYPEDEF STRUCT vv
4 VAR;
5 INTEGER a;
6 vv ENDSTRUCT
7 %comment
8 TYPEDEF STRUCT vv2
9 VAR;
10 INTEGER b;
11 vv2 ENDSTRUCT
12 TYPEDEF STRUCT vv3
13 VAR;
14 INTEGER Aaa, B;
15 vv3 ENDSTRUCT
16 FUNCTION test2()
17 {
18     RETURN B;
19 }
20 END FUNCTION
21 /*
22 % comment test2 test2 test2
23 */
24 STARTMAIN
25 VAR;
26 INTEGER A, B;
27 CHAR C;
28 INTEGER abc, i;
29 FOR i := 1 TO 2 STOP 3
30     FOR j := 1 TO 2 STOP 3 %comment a = b
31         a = test2(a) + test2(a, abc);
32     ENDFOR
33     a = B;
34     IF (a==0) THEN
35         WHILE (a==1) AND (a==100)
36             B = B + 1;
37         ENWHILE
38     ELSEIF (a==100)
39         WHILE (a==20)
40             PRINT "a", B;
41         ENWHILE
42     ELSE
43         SWITCH (a)
44         CASE (a1)
45             a = B + B;
46         CASE (a2)
47             PRINT "a", B;
48         DEFAULT:
49             PRINT "default";
50         END SWITCH
51     ENDIF
52 ENDFOR
53 ENDMAIN

```

Stack now 0 3 10 9 35 60  
 Entering state 96  
 Next token is token STRING ()  
 UnPiling token STRING ()  
 Entering state 136  
 Reading a token: Next token is token ENDST  
 HCT ()  
 UnPiling token ENDSTRUCT ()  
 Entering state 164  
 Reducing stack by rule 10 (line 100):  
 S1 = token TYPEDEF ()  
 S2 = token STRUCT ()  
 S3 = token STRING ()  
 S4 = token vars ()  
 S5 = token STRING ()  
 S6 = token ENDSTRUCT ()  
 S6 = token struct ()  
 Stack now 0 3 10  
 Entering state 36  
 Reducing stack by rule 7 (line 103):  
 S1 = token list\_structs ()  
 S2 = token struct ()  
 S3 = token list\_structs ()  
 Stack now 0 3  
 Entering state 18  
 Reading a token: Next token is token FUNCT  
 20M ()  
 UnPiling token FUNCTION ()  
 Entering state 6  
 Reading a token: Next token is token STRON  
 G ()  
 UnPiling token STRING ()  
 Entering state 15  
 Reading a token: Next token is token A\_PMA  
 ENTWEL ()  
 UnPiling token A\_PMAENTWEL ()  
 Entering state 41  
 Reading a token: Next token is token B\_PMA  
 ENTWEL ()  
 UnPiling token B\_PMAENTWEL ()  
 Entering state 43  
 Reading a token: Next token is token STRON  
 G ()  
 UnPiling token STRING ()  
 Entering state 16  
 Reading a token: Next token is token EQUAL  
 S ()  
 UnPiling token EQUAL ()  
 Entering state 42  
 Reading a token: Next token is token STRON  
 G ()  
 UnPiling token STRING ()  
 Entering state 50  
 Reading a token: Next token is token Q\_PMA  
 H ()  
 Reducing stack by rule 61 (line 154):  
 S1 = token STRING ()  
 (LINE 17) Variable "c" is  
 not declared  
 make: \*\*\* [Parsefile.c: run] Error 1:  
 src\test2.c(17): Variable "c" is  
 not declared  
 (Error 1)

- Παράδειγμα όπου η struct vv υπάρχει ήδη και σε ποιά line.

```

100     tempvar[10];
101     STRING A BRACKET THETAS_AER B BRACKET (
102         ...
103     );
104 }
105
106 PROGRAM TEST2
107
108 TYPEDEF STRUCT vv
109 VAR;
110 INTEGER a;
111 vv ENDSTRUCT
112 %comment
113 TYPEDEF STRUCT vv2
114 VAR;
115 INTEGER b;
116 vv2 ENDSTRUCT
117 TYPEDEF STRUCT vv3
118 VAR;
119 INTEGER Aaa, B;
120 vv3 ENDSTRUCT
121 FUNCTION test2()
122 {
123     RETURN B;
124 }
125 END FUNCTION
126 /*
127 % comment test2 test2 test2
128 */
129 STARTMAIN
130 VAR;
131 INTEGER A, B;
132 CHAR C;
133 INTEGER abc, i;
134 FOR i := 1 TO 2 STOP 3
135     FOR j := 1 TO 2 STOP 3 %comment a = b
136         a = test2(a) + test2(a, abc);
137     ENDFOR
138     a = B;
139     IF (a==0) THEN
140         WHILE (a==1) AND (a==100)
141             B = B + 1;
142         ENWHILE
143     ELSEIF (a==100)
144         WHILE (a==20)
145             PRINT "a", B;
146         ENWHILE
147     ELSE
148         SWITCH (a)
149         CASE (a1)
150             a = B + B;
151         CASE (a2)
152             PRINT "a", B;
153         DEFAULT:
154             PRINT "default";
155         END SWITCH
156     ENDIF
157 ENDFOR
158 ENDMAIN

```

Stack now 0 3 10 9 35 60  
 Entering state 96  
 Next token is token STRING ()  
 UnPiling token STRING ()  
 Entering state 136  
 Reading a token: Next token is token ENDST  
 HCT ()  
 UnPiling token ENDSTRUCT ()  
 Entering state 164  
 Reducing stack by rule 10 (line 100):  
 S1 = token TYPEDEF ()  
 S2 = token STRUCT ()  
 S3 = token STRING ()  
 S4 = token vars ()  
 S5 = token STRING ()  
 S6 = token ENDSTRUCT ()  
 S6 = token struct ()  
 Stack now 0 3 10 9 35 60 17 44  
 Entering state 47  
 Next token is token Q\_PMAH  
 H ()  
 UnPiling token Q\_PMAH  
 H ()  
 Entering state 164  
 Reading a token: Next token is token STRON  
 G ()  
 Reducing stack by rule 15 (line 100):  
 S1 = token INTEGER ()  
 S2 = token string list ()  
 S3 = token Q\_PMAH  
 H ()  
 S4 = token vars list ()  
 S5 = token vars list ()  
 Stack now 0 3 10 9 35 60 17  
 Entering state 45  
 Reducing stack by rule 11 (line 170):  
 S1 = token VAR; ()  
 S2 = token vars list ()  
 S3 = token vars list ()  
 S4 = token vars list ()  
 Stack now 0 3 10 9 35 60  
 Entering state 96  
 Next token is token STRING ()  
 UnPiling token STRING ()  
 Entering state 136  
 Reading a token: Next token is token ENDST  
 HCT ()  
 UnPiling token ENDSTRUCT ()  
 Entering state 164  
 Reducing stack by rule 10 (line 100):  
 S1 = token TYPEDEF ()  
 S2 = token STRUCT ()  
 S3 = token STRING ()  
 S4 = token vars list ()  
 S5 = token STRING ()  
 S6 = token ENDSTRUCT ()  
 (LINE 11) Struct "vv" was already defined  
 make: \*\*\* [Parsefile.c: run] Error 1:  
 src\test2.c(11): Struct "vv" was already  
 defined  
 (Error 1)



### **Multiline comments:**

Τα σχόλια πολλαπλών γραμμών υποστηρίζονται με το παρακάτω κομμάτι κώδικα στο flex αρχείο:

```
55 %%  
56 <INITIAL>{  
57     "/"*      BEGIN(IN_COMMENT);  
58 }  
59 <IN_COMMENT>{  
60     "/"*      BEGIN(INITIAL);  
61     [^*\n]+   {};  
62     "*"      {};  
63 }
```

### **Παραδοχές:**

- Οι τύποι μεταβλητών είναι οι integer και char.
- Οι έλεγχοι των δηλώσεων των functions είναι ίδια με των variables καθώς και στην περίπτωση όπου υπάρχει κάποια ίδια ή δεν έχει δηλωθεί κάποια.

### **Σχόλια:**

- Όνομα αρχείου flex, project.l
- Όνομα αρχείου bison, project.y
- Όνομα αρχείου input για έλεγχο, test2.txt
- Compile & Execute: cd PROJECTDIR && make clean && make && make run