ΜΕΤΑΦΡΑΣΤΕΣ

Ιουνιος 2005
Θέμα 2ο

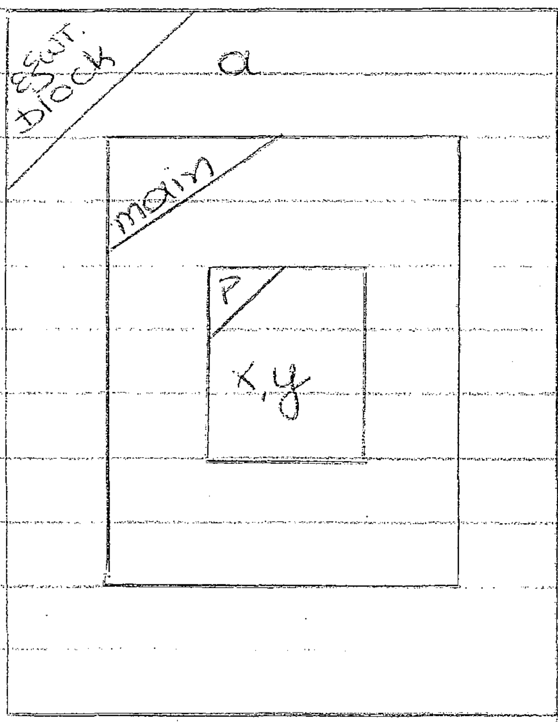
Δίνεται το παρακάτω πρόγραμμα σε C:

```
int a;
main()
{
  a=4;
  P(a,a);
  print(a);
}
void P(int x, int y)
  { ++x;
    ++y;
  }
```


α) Να ορίσετε τα περιβάλλοντα αναφοράς (τοπικό, καθολικό) της συνάρτησης P
β) Ποια η τελική τιμή του a, η οποία τυπώνεται στο τέλος εκτέλεσης του προγράμματος, αν η μεταβίβαση παραμέτρων γίνεται με:
  i) κλήση με τιμή (call by value)
  ii) κλήση με αναφορά (call by reference)
  iii) κλήση με τιμή αποτέλεσμα (call by value result)

Σε ποια περίπτωση από τις παραπάνω εμφανίζεται το φαινόμενο της ψευδωνυμίας, σε ποιο τμήμα του προγράμματος και με ποιον τρόπο;

Δυναμικός

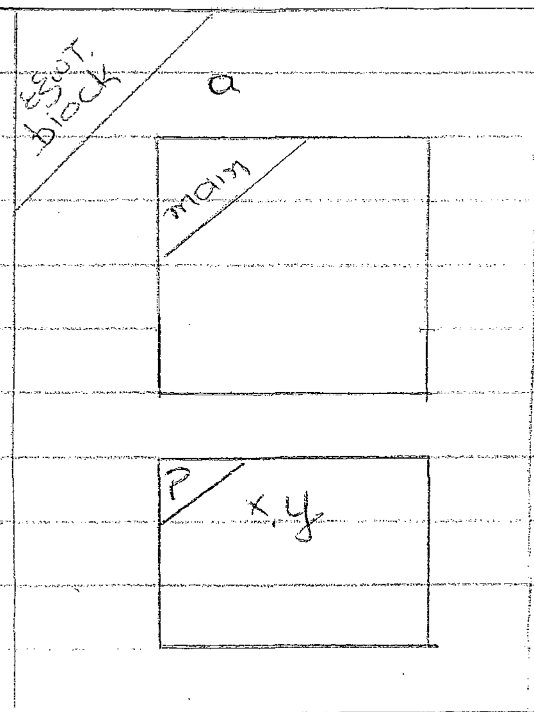


P

Τοπικό: x, y
μη Τοπικό: main, a, P
καθολικό: a, main, P

main
Τοπικό: –
μη Τοπικό και καθολικό:
a, main, P

Στατικός



P

Τοπικό: x, y
μη Τοπικό και καθολικό: a, main

main
Τοπικό: –
μη Τοπικό και καθολικό: a, main, P

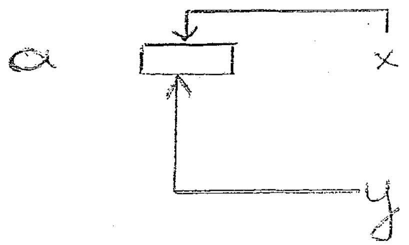
by value

a $\boxed{4}$ → $\boxed{x}$ 4   μετά την κλήση   x=5
a=4   → $\boxed{y}$ 4   "   "   "   y=5

ΠΡΟΣΟΧΗ!

Επειδή με τον τερματισμό της συνάρτησης, οι τυπικές παράμετροι <u>ΔΕΝ</u> επιστρέφονται στη main. Θα τυπωθεί η τιμή 4

ii) call by reference



- αρχικά a=4, καλεί p(4,4), x=4, y=4
- μετά x=5 (a=5, y=5 ← ψευδώνυμα)
- μετά y=6 (x=6, a=6 ← ψευδώνυμα)

ΠΑΡΑΤΗΡΗΣΗ: στο call by reference ισχύουν τα εξής
- στο x πάει η διεύθυνση του a (δηλαδή το x δείχνει (→) στο a)
- στο y πάει η διεύθυνση του a (δηλαδή το y δείχνει (→) στο a)

Αυτά ισχύουν σε Pascal και C

ΠΡΟΣΟΧΗ!

a, x, y → ΨΕΥΔΩΝΥΜΑ, δηλαδή οποιαδήποτε αλλαγή κάνω σε μια από τις τρεις μεταβλητές, αλλάζουν αυτομάτως όλες.

... στο πρόγραμμα μου με τον τερματισμό της συνάρτησης θα τυπωθεί η τιμή 6

iii) call by value result

a 4 ⟶ $\boxed{x}$ 4 μετά την κλήση x=5

a 4 ⟶ $\boxed{y}$ 4 " " " y=5

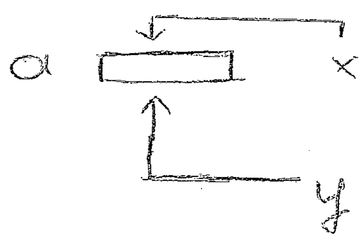
5 a ⟵ $\boxed{5}$ x

5 a ⟵ $\boxed{5}$ y

<u>ΠΡΟΣΟΧΗ</u>

Η call by value result λειτουργεί όπως η call by value <u>ΑΛΛΑ</u> εδώ μετά το τέλος της συνάρτησης έχουμε και <u>αντίστροφη καταχώρηση</u>. Επομένως, στο πρόγραμμα θα τυπωθεί η τιμή 5

Το φαινόμενο της ψευδωνυμίας εμφανίζεται στην περίπτωση του <u>call by reference</u> στο τμήμα εκείνο που καλείται η P(a,a), όπου συμβαίνει το εξής:



Εδώ τα x,y είναι ψευδώνυμα καθώς αναφέρονται στην ίδια θέση μνήμης

Θεμα 2ο

Δινεται Το παρακατω προγραμμα σε Pascal.

```
Program MAIN
  var i, j, k, m: integer
  procedure Q(i, m: integer);
     begin
     i:= i+k;
    m:= j+1
    write(i, j, k, m) ..... (A)
    end;
  procedure P (i, j: integer);
    var k: integer;
    begin
    k:= 3;
    i:=i+k;
   j:=j+k;
    Q(i, j)
    end;
 begin
 i:=1; j:=2; k:=4;
 P(i, k)
 write(i, j, k) ... (B)
 end
```

...θέτοντας ότι χρησιμοποιείται στατικός κανόνας εμβέλειας, να ορίσετε τα περιβάλλοντα αναφοράς (τοπικό, μη τοπικό και καθολικό) των MAIN, Q, P

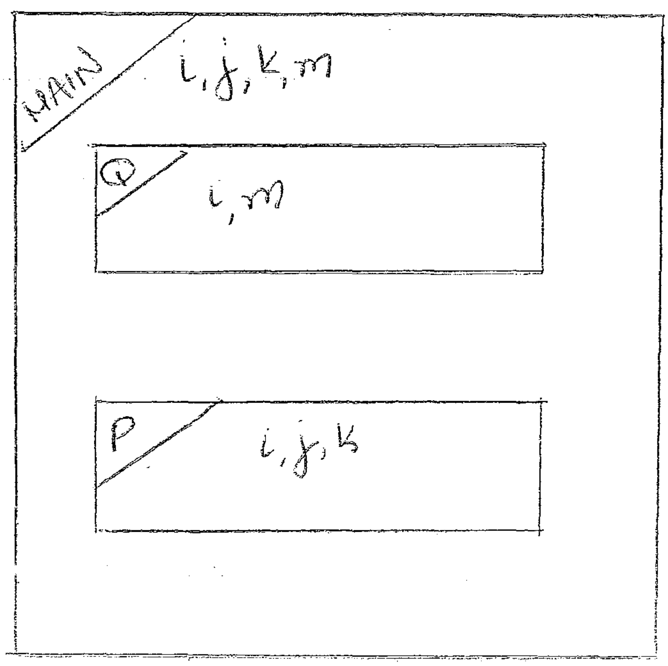
β) Να απαντήσετε το παραπάνω ερώτημα αν χρησιμοποιείται δυναμικός κανόνας εμβέλειας.

γ) Με χρήση στατικού κανόνα εμβέλειας να παρουσιάσετε τις τιμές που θα εκτυπωθούν στις γραμμές (A) και (B) όταν η μεταβίβαση παραμέτρων γίνεται:
(1) με call by value
(2) με call by reference
(3) με call by value-result

δ) Να απαντήσετε στο παραπάνω ερώτημα (γ) όταν χρησιμοποιείται δυναμικός κανόνας εμβέλειας.

Λύση

(α)



[illegible]: i, j, k, m, Q, P

Q

Τοπικό: i, m

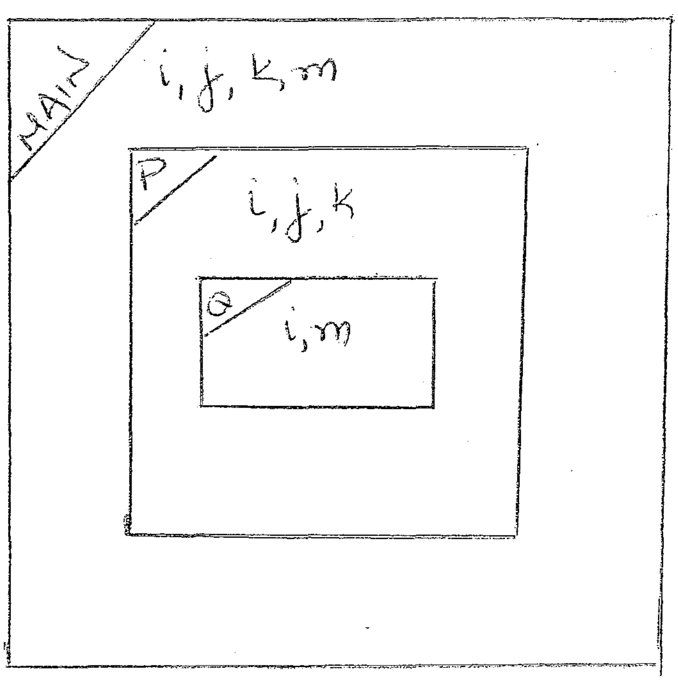
μη τοπικό και καθολικό: j, k, P, Q

P

τοπικό: i, j, k

μη τοπικό και καθολικό: m, P, Q

β)



MAIN

Τοπικό: i, j, k, m, P, Q

P

Τοπικό: i, j, k

Τοπικό και καθολικό: m, P, Q

Q

Τοπικό: i, m

μη τοπικό: j, k, P, Q

καθολικό: P, Q

Στατικός Κανόνας

* call by value

main   P   Q

① → i $\boxed{\cancel{1}}$ $\boxed{4}$ → i $\boxed{\cancel{4}}$ $\boxed{8}$

by result

②

④ → j $\boxed{\cancel{4}}$ $\boxed{7}$ → m $\boxed{\cancel{7}}$ $\boxed{3}$

by result

k $\boxed{B}$ k $\boxed{A}$

j $\boxed{3}$

Άρα (A) → 8, 2, 4, 3

(B) → 1, 2, 4

* call by value result

(A) → 8, 2, 4, 3 (όπως και από πάνω)

(B) → 8, 2, 3 (γιατί το i του ① επιστρέφει στο i της main και το m του Q επιστρέφει στο k του main)

- call by reference

main P Q

$\cancel{1}$ 4 11 ← i ← i

$\cancel{4}$ $\cancel{7}$ 3 ← j ← m ⇝ παίρνω j καθολικό

j 2 k 3

τοπική

(A) 11, 2, 3, 3

(B) 11, 2, 3

Δυναμικός Κανόνας

- call by value

main P Q

i 1 → i $\boxed{\cancel{1}}$ $\boxed{4}$ → i $\boxed{4}$ $\boxed{}$

by result

j 2

k 4 → j $\boxed{4}$ $\boxed{7}$ → m $\boxed{7}$ $\boxed{}$

by result

k $\boxed{3}$ k $\boxed{3}$

j $\boxed{7}$

Άρα (A) → 7, 7, 3, 8

(B) → 1, 2, 4

*call by value result

(A) → 7, 7, 3, 8 (οπως και πριν)
(B) → 7, 2, 8 (Το i Του Q πάει στο i της main και Το m Του Q στο k της main)

*call by reference

main   P   Q

i ~~1~~ ~~4~~ 7 ← i ← i
k ~~4~~ ~~7~~ 8 ← j ← m
j 2   k = 3

(A): 7, 8, 3, 8
(B): 7, 2, 8

Θέμα

Δίνεται το παρακάτω πρόγραμμα σε μια γλώσσα όπου Pascal που χρησιμοποιεί στατικό κανόνα εμβέλειας.

```
Program MAIN;
 var
 i:integer;
 A:array [1...2] of integer;
procedure F(x,y,z:integer);
begin
x:=x+1;
y:=z;
z:=z+1;
end
begin
i:=1;
A[1]:=10;
A[2]:=11;
F(i,A[i],i);
write (i,A[1],A[2]);
end
```

α) Να ορίσετε τα περιβάλλοντα αναφοράς (τοπικά, μη τοπικά, καθολικά), όλων των τμημάτων του προγράμματος

β) Ποιες τιμές τυπώνονται με την τελευταία εντολή στο κυρίως πρόγραμμα αν η μεταβίβαση γίνεται με:

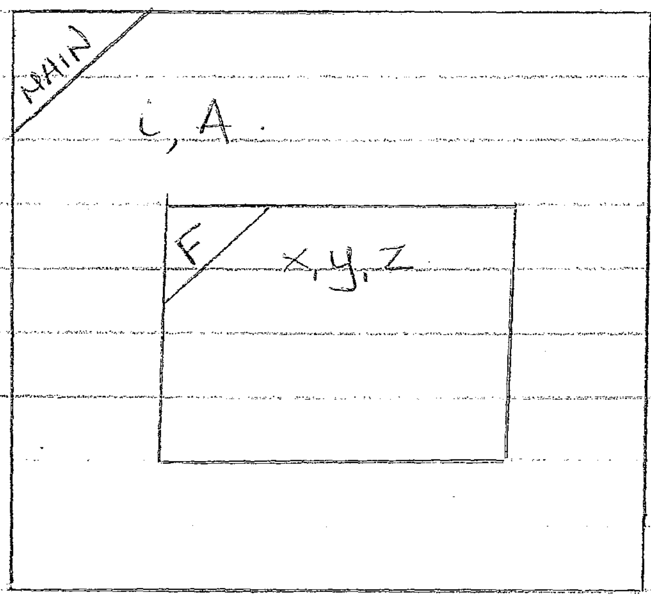
1) call by value

2) call by value-result

3) call by reference

4) call by name

(a) στατικός κανόνας



MAIN

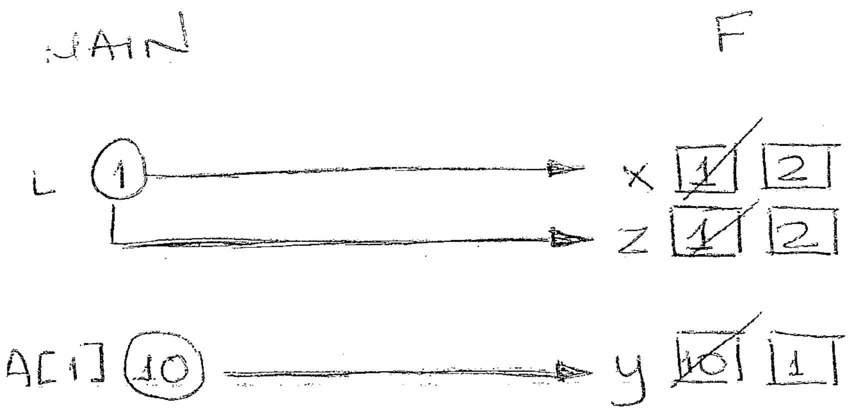
Τοπικό : i, A, F

F

Τοπικό : x, y, z

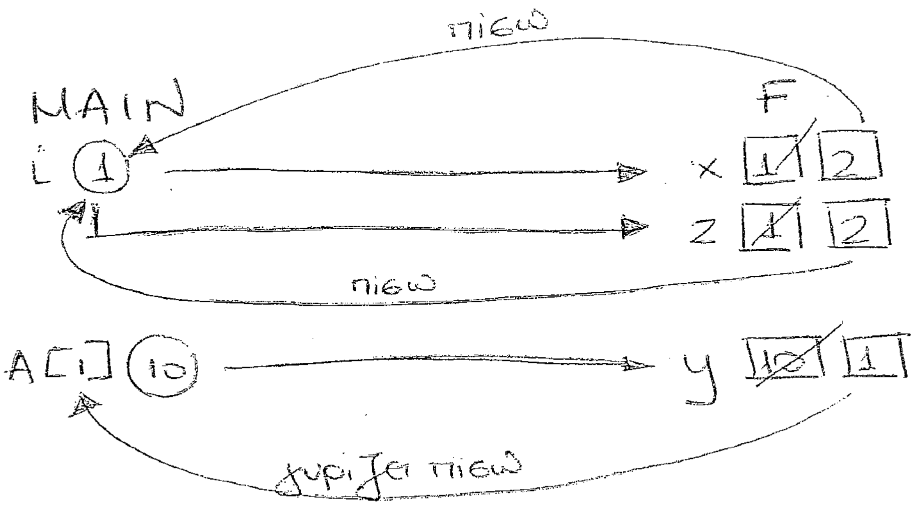
μη Τοπικό και καθολικό : i, A, F

call by value



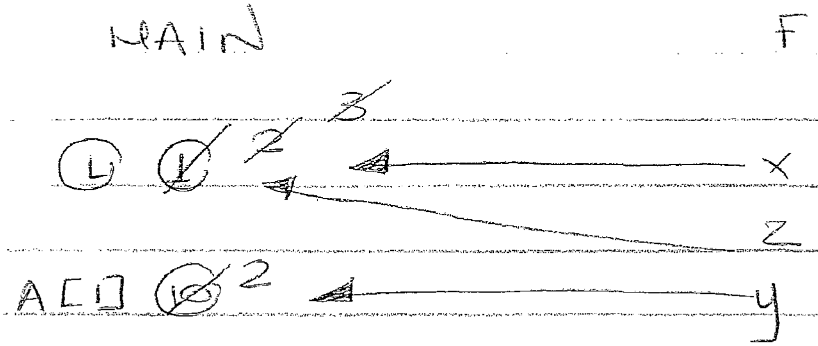
Θα τυπωθούν στη MAIN οι τιμές 1, 10, 11

(2) call by value result



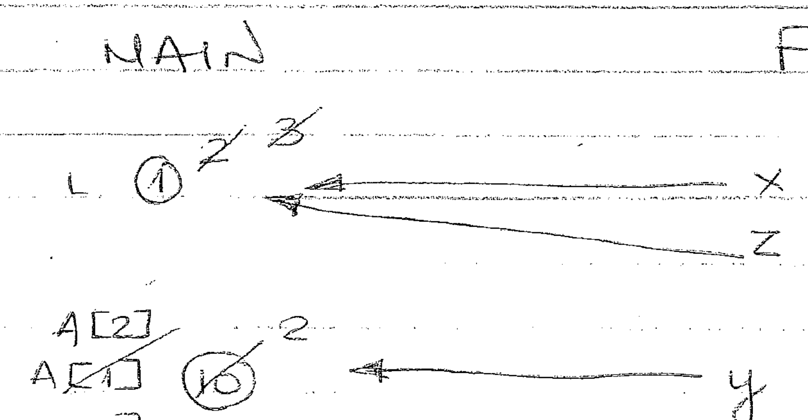
Θα τυπωθούν στη MAIN οι τιμές 2, 1, 11

3) call by reference



Θα τυπωθούν στη MAIN οι τιμές 3, 2, 11

4) call by name



A[3] → απλή αναφορά

Θα τυπωθούν οι τιμές 3, 10, 2

ΠΡΟΣΟΧΗ!

Όταν η πραγματική παράμετρος είναι ακέραια ή πραγματικός το call by name ταυτίζεται με το call by reference. Όταν όμως η πραγματική παράμετρος είναι σταθερά ή παράσταση ταυτίζεται με το call by value.

Γραμματική LL(1) SOS!

Μια γραμματική είναι LL(1) όταν τηρεί τις ακόλουθες προϋποθέσεις:

1) Αν $A \rightarrow a$ και $A \rightarrow b$ πρέπει $FIRST(A) \cap FIRST(A) = \emptyset$, όπου FIRST(A) το πρώτο τερματικό σύμβολο που βρίσκεται δεξιά του κανόνα που αρχίζει με A. Δηλαδή κάθε κανόνας πρέπει να έχει διαφορετικό αρχικό τερματικό σύμβολο δεξιά του.

2) Αν $A \rightarrow \varepsilon$ πρέπει $FIRST(A) \cap FOLLOW(A) = \emptyset$.
Δηλαδή αν υπάρχει κενή παραγωγή τότε θα πρέπει το πρώτο τερματικό σύμβολο στα δεξιά κάθε κανόνα που αρχίζει με A να είναι διαφορετικό από το πρώτο τερματικό σύμβολο που ακολουθεί το A σε κάποιο κανόνα που το A είναι δεξιά του βέλους.

π.χ. $A \rightarrow \alpha$ FIRST(A)
$\rightarrow \ldots A\alpha$ FOLLOW(A)

3) Όχι αριστερή αναδρομή

Εφαρμογή
Είναι η ακόλουθη γραμματική LL(1); Αν όχι, να γίνει
$S \rightarrow ASb \mid BSc \mid \varepsilon$
$A \rightarrow a$

$\Xi \rightarrow aS$.

Λύση

Ο κανόνας με το S δεν έχει πρόβλημα, αφού κάθε δεξιά παραγωγή αρχίζει με μη τερματικό.

Όμως $FIRST(A) \cap FIRST(B) \neq \emptyset$, άρα η (1) παραβιάζεται. Επομένως, η γραμματική δεν είναι LL(1)

$S \rightarrow aSb \mid aSSc \mid \varepsilon$

$S \rightarrow aSx \mid \varepsilon$

$x \rightarrow b \mid Sc$

- $FIRST(S) \cap FIRST(x) = \emptyset$
- $FIRST(S) \cap FOLLOW(S) = \emptyset$

Άρα η γραμματική είναι LL(1)

Αριστερή Αναδρομή

Εφαρμογή

Να ελέγξετε αν η ακόλουθη γραμματική έχει αριστερή αναδρομή. Αν έχει, να την απαλείψετε.

$S \rightarrow Aa \mid b$

$A \rightarrow Ac \mid Sd \mid f$

Η γραμματική αυτή παρουσιάζει άμεση αναδρομή λόγω του κανόνα $A \rightarrow Ac$. Παρουσιάζει όμως και

...μμεση αναδρομή διότι εάν στον κανόνα $S \to Aa$ ...θέσω όπου $A \to Sd$, οδηγούμαι στην $S \to Sda$ δηλαδή ...και πάλι σε άμεση αναδρομή

Πρώτα απαλείφω την έμμεση αριστερή αναδρομή με αντικατάσταση

$S \to Aa \mid b$
$A \to Ac \mid Aad \mid bd \mid f$

Οποτε η γραμματική αυτή έχει μόνο άμεση αριστερ... αναδρομή. Αυτή απαλείφεται σύμφωνα με τον ακόλουθο κανόνα.

Αν $A \to Aa_1 \mid \dots \mid Aa_n \mid \beta_1 \mid \dots \mid \beta_m$, τότε
$A \to \beta_1 B \mid \dots \mid \beta_m B$
$B \to a_1 B \mid \dots \mid a_n B \mid \varepsilon$

Γράφω τους κανόνες που δεν έχουν πρόβλημα με ένα καινούριο μη τερματικό σύμβολο.
Επομένως:
$S \to Aa \mid b$
$A \to bdB \mid fB$
$B \to cB \mid adB \mid \varepsilon$

Μετασχηματισμοί

I) Αντικατάσταση

$A \to a_1 | a_2 | \ldots | a_n$

$B \to b_1 A b_2$

$\Updownarrow$

$A \to a_1 | a_2 | \ldots | a_n$

$B \to b_1 a_1 b_2 | b_1 a_2 b_2 | \ldots | b_1 a_n b_2$

II) Αριστερή παραγοντοποίηση

$A \to a b_1 | a b_2 | \ldots | a b_n$

$\Updownarrow$

$A \to a B$

$B \to b_1 | b_2 | \ldots | b_n$

(III) Απαλοιφή Αριστερής Αναδρομής

$A \to A a_1 | \ldots | A a_n | b_1 | \ldots | b_m$

$\Updownarrow$

$A \to b_1 B | \ldots | b_m B$

$B \to a_1 B | \ldots | a_n B | \varepsilon$

-άσκηση

α) Παρουσιάστε μια BNF γραμματική, η οποία επιτρέπει την εκτέλεση αριθμητικών πράξεων μεταξύ αριθμών, θεωρώντας ως αριθμητικούς τελεστές το + και το *. Επίσης, μπορούν να χρησιμοποιηθούν και παρενθέσεις, οι οποίες αλλάζουν την προτεραιότητα των πράξεων.

β) Αν η γραμματική είναι διφορούμενη, τότε να γίνει μη διφορούμενη

γ) Να παρουσιάσετε το συντακτικό δέντρο για την παράσταση (A+B)*C

Λύση

Ορισμός: Μια ΓΧΣ ονομάζεται **διφορούμενη** όταν υπάρχουν δύο ή περισσότερα συντακτικά δέντρα (ισοδύναμα αριστερότερες ή δεξιότερες παραγωγές) για την ίδια παραγόμενη συμβολοσειρά.

α) Η γραμματική σε BNF συμβολισμό είναι η ακόλουθη:

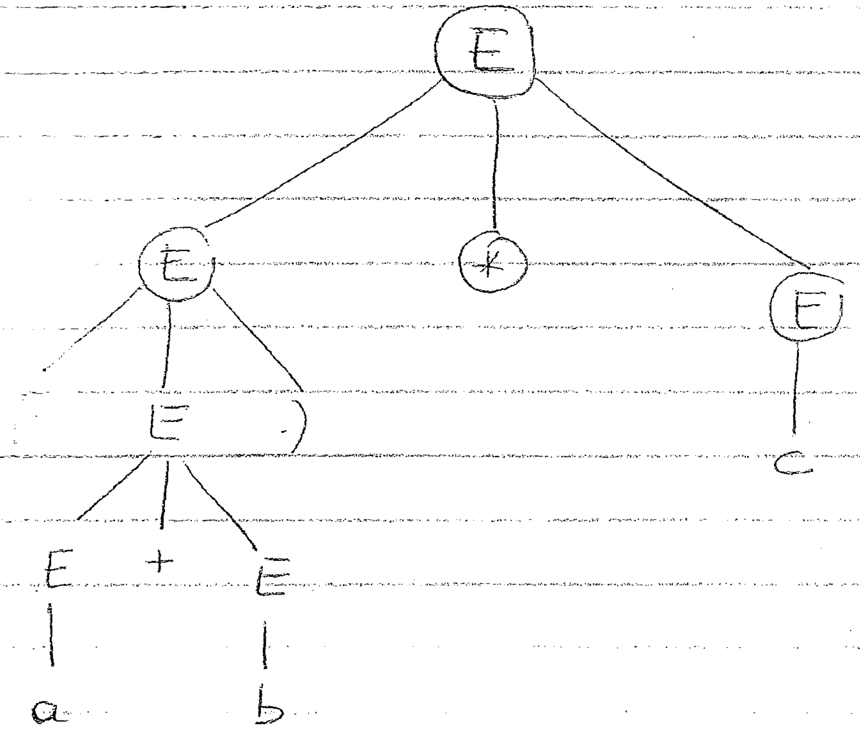
<E> ::= <E>+<E> | <E>*<E> | (<E>) | id

β) Η γραμματική είναι διφορούμενη διότι για την κατασκευή της συμβολοσειράς a+b*c υπάρχουν δύο διαφορετικές παραγωγές

$E \Rightarrow E*E \Rightarrow E+E*E \Rightarrow a+b*c$

$E \Rightarrow E+E \Rightarrow E+E*E \Rightarrow a+b*c$

$E \Rightarrow E*E \Rightarrow (E)*E \Rightarrow (E+E)*E \Rightarrow (a+b)*c$



Προϋποθέσεις LL(1)

1) Όχι αριστερή αναδρομή
   π.χ. όχι κανόνες της μορφής $A \to Aa$
2) Όχι δύο κανόνες που έχουν ίδιο πρώτο τερματικό σύμβολο μετά το $\to$.
   π.χ. όχι κανόνες της μορφής $A \to aB$ και $\Gamma \to a\Delta$
3) Όχι δυο κανόνες τα δεξιά μέλη των οποίων να παράγουν την κενή συμβολοσειρά
   π.χ. όχι $A \to aB$ και $\Gamma \to b\Delta$, όπου $B \to \varepsilon$ και $\Delta \to \varepsilon$

Άσκηση

Δίνονται οι τρεις παρακάτω απλές γραμματικές σε συμβολισμό BNF.

1) $\langle A \rangle ::= 0\langle A \rangle \mid 1\langle A \rangle \mid 0$

2) $\langle S \rangle ::= \langle S \rangle\langle S \rangle \mid 0 \mid 1$

3) $\langle \Lambda \rangle ::= 0\langle \Lambda \rangle 0 \mid 1\langle \Lambda \rangle 1 \mid 0 \mid 1$

α) Περιγράψτε τις ακολουθίες δυαδικών ψηφίων που παράγουν οι τρεις γραμματικές. Δώστε και μικρά παραδείγματα

β) Αντιστοιχίστε τις ακολουθίες 0111010, 1101101, 0100010, 110110111, 11111 στην (ή στις) γραμματική (ή γραμματικές) που τις παράγει (ή παράγουν) ή σε καμία αν δεν μπορούν να παραχθούν

γ) Ποια ή ποιες γραμματικές είναι διφορούμενη/ες. Αποδείξτε την απάντησή σας με ένα μικρό παράδειγμα.

δ) Η πρώτη γραμματική (Α) είναι κανονική και γιατί; Αν ναι, κατασκευάστε το πεπερασμένο αυτόματο (το διάγραμμα καταστάσεων-μεταβάσεων) ισοδύναμο με τη γραμματική, δηλαδή να αναγνωρίζει τις νόμιμες ακολουθίες δυαδικών ψηφίων που παράγει η γραμματική.

Άσκηση

Δίνονται οι τρεις παρακάτω απλές γραμματικές σε συμβολισμό BNF.

1) <A>::=0<A>|1<A>|0

2) <S>::=<S><S>|0|1

3) <Λ>::=0<Λ>0|1<Λ>1|0|1

α) Περιγράψτε τις ακολουθίες δυαδικών ψηφίων που παράγουν οι τρεις γραμματικές. Δώστε και μικρά παραδείγματα

β) Αντιστοιχίστε τις ακολουθίες 011010, 110110, 0100010, 11011011, 11111 στην (ή στις) γραμματική (ή γραμματικές) που τις παράγει (ή παράγουν) ή σε καμία αν δεν μπορούν να παραχθούν

γ) Ποια ή ποιες γραμματικές είναι διφορούμενη/ες. Αποδείξτε την απάντησή σας με ένα μικρό παράδειγμα.

δ) Η πρώτη γραμματική (A) είναι κανονική και γιατί; Αν ναι, κατασκευάστε το πεπερασμένο αυτόματο (το διάγραμμα καταστάσεων - μεταβάσεων) ισοδύναμο με τη γραμματική, δηλαδή να αναγνωρίζει τις νόμιμες ακολουθίες δυαδικών ψηφίων που παράγει η γραμματική.

α)

Η <Α> παράγει όλες τις ακολουθίες διαδικών ψηφίων που τελειώνουν σε 0

π.χ. 00, 10, 100

Η <Σ> παράγει όλες τις ακολουθίες δ.ψ. αρτίου μήκους

π.χ. 0000

Η <Λ> παράγει ακολουθίες δ.ψ. που ξεκινούν και τελειώνουν με το ίδιο σύμβολο

π.χ. 010, 101

b)

- 0111010 ⟶ Γραμματική <Α> γιατί τελειώνει σε 0
  ⟶ Γραμματική <Λ> γιατί αρχίζει και τελειώνει με το ίδιο σύμβολο

- 1101101 ⟶ Γραμματική <Λ> γιατί αρχίζει και τελειώνει με το ίδιο σύμβολο

- 0100010 ⟶ Γραμματική <Α> γιατί τελειώνει σε 0
  ⟶ Γραμματική <Λ> γιατί αρχίζει και τελειώνει με το ίδιο σύμβολο

- 110110111 ⟶ Γραμματική <Λ> γιατί αρχίζει και τελειώνει με το ίδιο σύμβολο

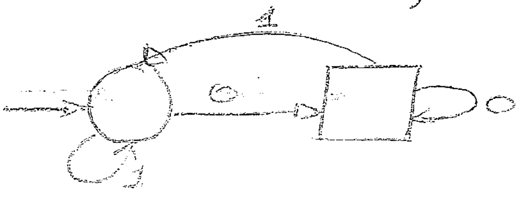
. 1111 1 → Γραμματική <Λ> γιατί ξεκινάει και τελειώνει με το ίδιο σύμβολο

ε)

ζ) Ορισμός Κανονικής Γραμματικής

Μια Κανονική Γραμματική είναι μία ΓΧΣ, η οποία στο δεξί μέρος κάθε κανόνα περιέχει το πολύ ένα μη τερματικό σύμβολο, που αν υπάρχει θα είναι το τελευταίο σύμβολο του κανόνα.
Με άλλα λόγια, έχει κανόνες της μορφής $A \to aB$, $A \to a$ ή $A \to \varepsilon$
ΓΧΣ ονομάζεται η γραμματική, οι κανόνες της οποίας είναι της μορφής : $A \to a$, όπου A μη τερματικό και a συμβολοσειρά (δηλαδή όλες οι γραμματικές που σε κάθε κανόνα τους έχουν αριστερά του → μόνο ένα μη τερματικό σύμβολο)

Η συγκεκριμένη γραμματική πληροί τους κανόνες που αναφέραμε, άρα είναι κανονική. Το DFA είναι:



Θέμα 4ο

Δίνεται ο παρακάτω κώδικας σε C:

```
int x,y,z;
main()
{
 x=15;
 y=10;
 z=A(x,y);
 print(z);
}
int A(int u,int v)
{
 if(v==0)
   return ;
 else
   return A(v,u%v);
}
```

a) Ποια είναι η λειτουργία της function A;

b) Παρουσιάστε τη στοίβα εκτέλεσης (run-time stack) των Εγγραφών Ενεργοποίησης (Activation Records-AR) κατά την εκτέλεση της τελευταίας κλήσης της A (δηλαδή όταν είναι σε ισχύ το μέγιστο μέγεθος μιας στοίβας εκτέλεσης). Ποια τιμή του z θα τυπωθεί τελικά στη main;

c) Στο συγκεκριμένο πρόγραμμα, ποια είναι η χρησιμότητα (αν υπάρχει) των στατικών δεικτών (Static Pointers) της στοίβας εκτέλεσης;

Λύση

α) Η συνάρτηση αυτή είναι αναδρομική και υπολογίζει το ΜΚΔ των ορισμάτων που λαμβάνει. Ο τρόπος με τον οποίο εκτελείται φαίνεται στο ακόλουθο σχήμα.

```
A(15, 10)            A(10, 5)             A(5, 0)
{                    {                    {
  return A(10,5)       return A(5,0)        5
}                    }                    }
5
```

(β)

Κενή Στοίβα	Είσοδος στο αρχείο	Εκτέλεση της main
	x ακέραιος	x ακέραιος
	y ακέραιος	y ακέραιος
	z ακέραιος	z ακέραιος
	main όνομα συνάρτ.	main όνομα συνάρτησης
	A όνομα συνάρτησης	A όνομα συνάρτησης
		A όνομα συνάρτησης } main

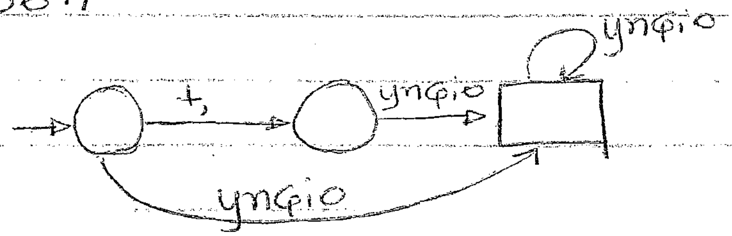
1η κλήση της A	2η κλήση της A	3η κλήση της A
x ακέραιος	x ακέραιος	x ακέραιος
y ακέραιος	y ακέραιος	y ακέραιος
z ακέραιος	z ακέραιος	z ακέραιος
main όνομα συν/σης	main ον. συν/σης	main ον. συν/σης
A όνομα συν/σης	A ον. συν/σης	A ον. συν/σης
A όνομα συν/σης } main	A ον. συν/σης } main	A ον. συν/σης } main
u ακέραιος v ακέραιος } A (1η εκτέλεση)	u ακέραιος v ακέραιος } A	u ακέραιος v ακέραιος } A
	u ακέραιος v ακέραιος	u ακέραιος v ακέραιος } A
		u ακέραιος v ακέραιος } A.

Στη main τυπώνεται τελικά 5 που είναι ο ΜΚΔ(15,10)
Η χρησιμότητα του στατικού δείκτη είναι να δείχνει τις καθολικές μεταβλητές του προγράμματος, κάτι που είναι απαραίτητο όταν τα υποπρογράμματα χρησιμοποιούν καθολικές μεταβλητές. Η συνάρτηση Α δε χρησιμοποιεί καθολικές μεταβλητές οπότε ο στατικός δείκτης δεν έχει κάποια χρησιμότητα στο συγκεκριμένο πρόγραμμα.

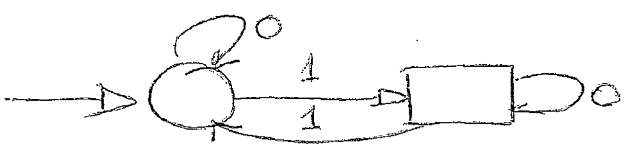
Ασκήσεις σε αυτόματα

1) Να φτιάξετε NFA που να αναγνωρίζει προσημασμένους ή μη προσημασμένους ακεραίους

Λύση

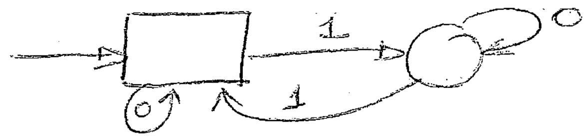


2) Να φτιάξετε DFA που να αναγνωρίζει δυαδικούς αριθμούς με περιττό αριθμό άσσων.

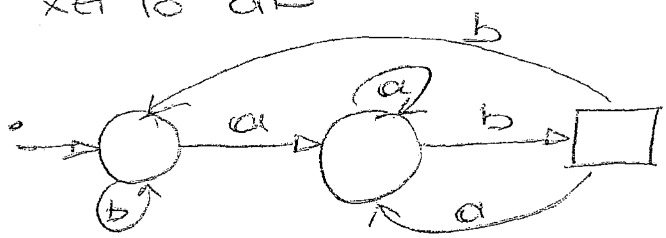


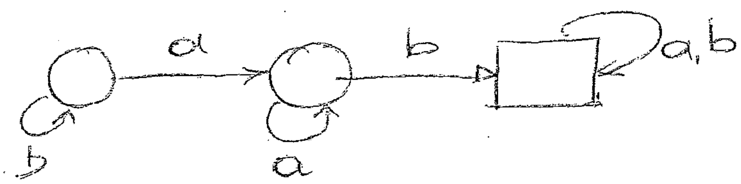
ΠΡΟΣΟΧΗ!
Στο DFA πρέπει για κάθε κατάσταση να γράφουμε ΟΛΕΣ τις πιθανές εξόδους

4) Να φτιάξετε DFA που να αναγνωρίζει δυαδικούς αριθμούς με άρτιο αριθμό άσων



5) Να φτιάξετε DFA που να αναγνωρίζει τα tokens που να τελειώνουν σε ab και DFA που να περιέχει το ab





Θέμα 2ο

Θέλουμε να ορίσουμε μια γλώσσα περιγραφής απλώ
αριθμητικών εκφράσεων μεταξύ μη αρνητικών
ακεραίων οποιουδήποτε μήκους. Η γλώσσα υποστηρίζει τους
εξής δυαδικούς τελεστές, οι οποίοι είναι ισοδύναμοι
μεταξύ τους, δηλαδή η προτεραιότητα εκτέλεσής τους
καθορίζεται μόνο από τη θέση τους στην έκφραση.
(φθίνουσα από αριστερά προς δεξιά)

^ ύψωση σε ακέραια δύναμη
* πολλαπλασιασμός
+ πρόσθεση

Η γλώσσα περιλαμβάνει εκτός από τα παραπάνω τρία
τερματικά σύμβολα, παρενθέσεις που χρησιμοποιούνται
για την αλλαγή προτεραιότητας και φυσικά τα ψηφία
0-9

α) Περιγράψτε το συντακτικό της γλώσσας με χρήση
γραμματικής BNF

β) Μεταφράστε τη γραμματική BNF σε συντακτικά
διαγράμματα που εμφανίζουν την ειδική δομή δεδομένων, η οποία μπορεί να χρησιμοποιηθεί από ένα
γενικό πρόγραμμα συντακτικής ανάλυσης

γ) Να προσπαθήσετε να κατασκευάσετε τα δέντρα
συντακτικής ανάλυσης για τις παρακάτω αριθμητικές εκφράσεις για να συμπεράνετε αν είναι
«νόμιμες» εκφράσεις της γλώσσας:

1> 35+(447*(26+7)^4)+2

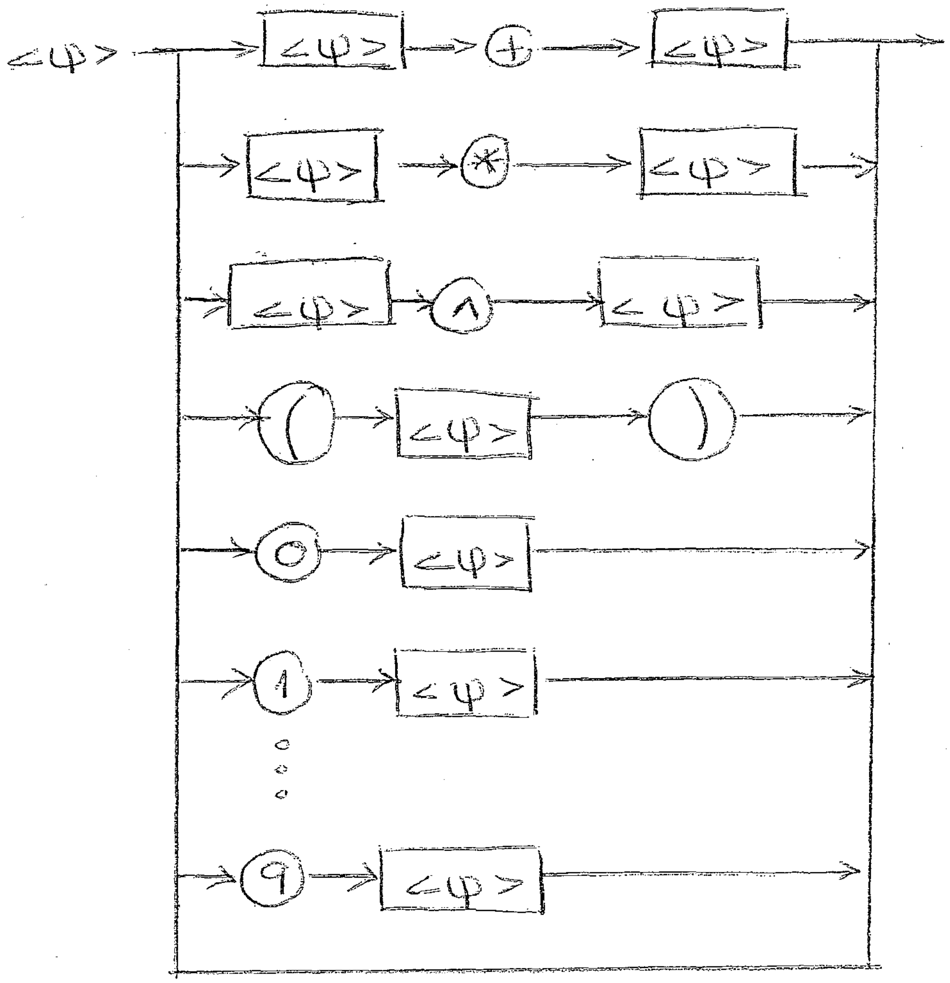
2> (27)+86*(32^)+73

Λύση

α) Οι κανόνες της γραμματικής είναι:

$$\langle\psi\rangle ::= \langle\psi\rangle + \langle\psi\rangle \mid \langle\psi\rangle * \langle\psi\rangle \mid \langle\psi\rangle \,\hat{}\, \langle\psi\rangle \mid (\langle\psi\rangle)$$
$$\langle\psi\rangle ::= 0\langle\psi\rangle \mid 1\langle\psi\rangle \mid 2\langle\psi\rangle \mid \ldots \mid 9\langle\psi\rangle \mid \varepsilon$$

β)



γ) $\langle\psi\rangle \Rightarrow \langle\psi\rangle + \langle\psi\rangle \Rightarrow \langle\psi\rangle + \langle\psi\rangle + \langle\psi\rangle \Rightarrow \langle\psi\rangle + (\langle\psi\rangle) + \langle\psi\rangle \Rightarrow$

$\Rightarrow \langle\psi\rangle + (\langle\psi\rangle \wedge \langle\psi\rangle) + \langle\psi\rangle \Rightarrow \langle\psi\rangle + (\langle\psi\rangle * \langle\psi\rangle \wedge \langle\psi\rangle) + \langle\psi\rangle \Rightarrow$

$\Rightarrow \langle\psi\rangle + (\langle\psi\rangle * (\langle\psi\rangle) \wedge \langle\psi\rangle) + \langle\psi\rangle \Rightarrow \langle\psi\rangle + (\langle\psi\rangle * (\langle\psi\rangle + \langle\psi\rangle) \wedge \langle\psi\rangle)$

$\langle\psi\rangle \Rightarrow 3\langle\psi\rangle + (4\langle\psi\rangle * (2\langle\psi\rangle + 7) \wedge 4) + 2 \Rightarrow$

$35 + (44\langle\psi\rangle * (26+7) \wedge 4) + 2 \Rightarrow 35 + (447 * (26+7) \wedge 4) + 2$

Η δευτερη εκφραση δεν ειναι νόμιμη διότι δεν παράγεται από τους κανόνες της γλώσσας

Άσκηση σε περιβάλλοντα αναφοράς

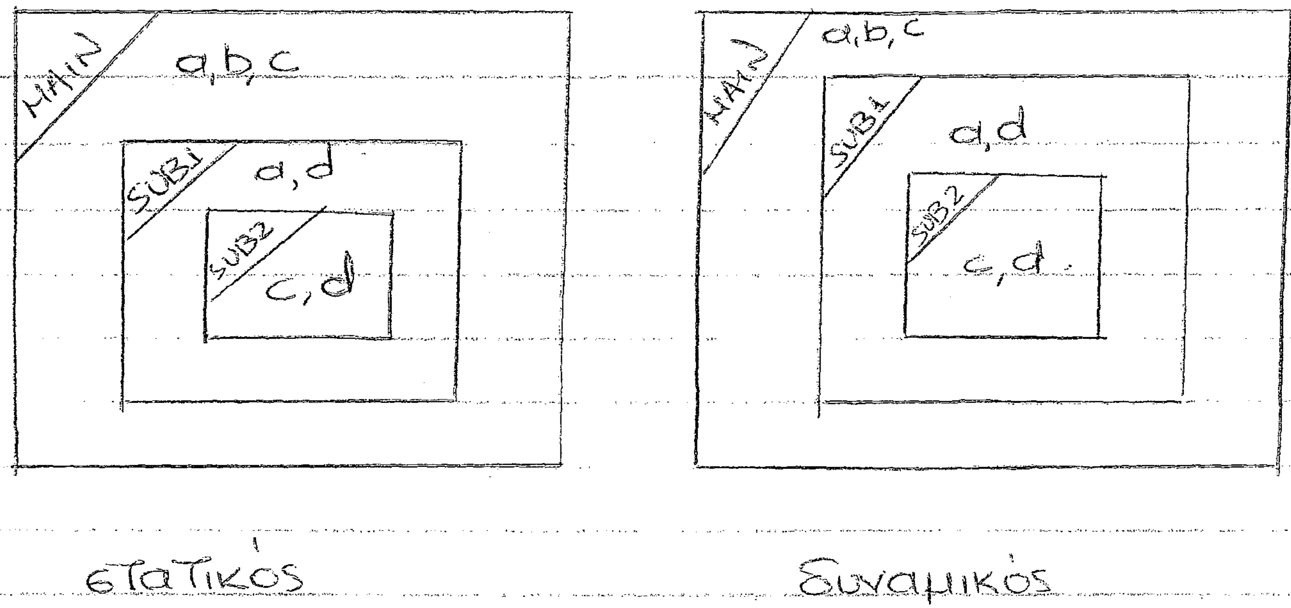
Δίνεται το ακόλουθο πρόγραμμα σε Pascal

```
Program MAIN
var a,b,c:real
 procedure SUB1 (a:real)
     var d:real
     procedure SUB2 (c:real)
     var d:real;
          begin
          c:=c+R;
          end;
     begin
          SUB2(b);
     end
 BEGIN
 SUB1(a);
 END;
```

(a) Να προσδιορίσετε τα περιβάλλοντα αναφοράς για το κάθε υποπρόγραμμα καθώς και το κύριο πρόγραμμα, χρησιμοποιώντας στατικό και δυναμικό κανόνα εμβέλειας.

Λύση

α)



Παρατηρούμε ότι η σειρά με την οποία γράφονται τα blocks είναι ΙΔΙΑ και στο στατικό και στο δυναμικό κανόνα εμβέλειας.

MAIN

Τοπικό : a, b, c, SUB1

SUB1

Τοπικό : a, d, SUB2

Μη Τοπικό και καθολικό : SUB1, b

SUB2

Τοπικό : c, d

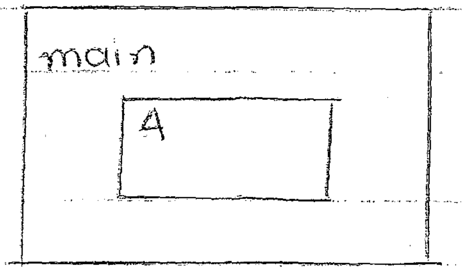
Μη Τοπικό : a, SUB1, SUB2

Καθολικό : b, SUB1

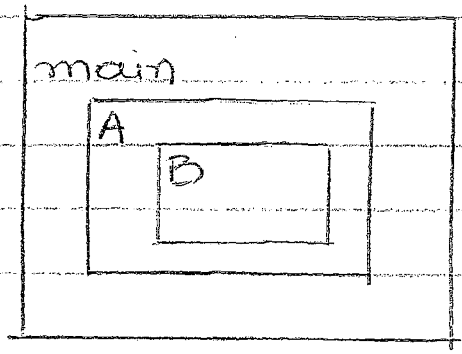
Παρατηρήσεις

1) Το main έχει μόνο τοπικό περιβάλλον. Μέσα στο τοπικό περιβάλλον του main γράφουμε τις μεταβλητές που δηλώνονται μέσα στο main, καθώς και τα υποπρογράμματα που δηλώνονται μέσα στο main.

(2) Ένα υποπρόγραμμα Α της μορφής:



έχει μη τοπικό περιβάλλον που ταυτίζεται με το καθολικό περιβάλλον του, έχω ένα υποπρόγραμμα Β της μορφής:



έχει μη τοπικό περιβάλλον που είναι διαφορετικό από το καθολικό περιβάλλον του

(3) Στο τοπικό περιβάλλον ενός υποπρογράμματος γράφουμε όλες τις τοπικές μεταβλητές που δηλώνονται σε αυτό καθώς και όλες τις τυπικές παραμέτρους που δηλώνονται σε αυτό. Επίσης γράφουμε και όλα τα υποπρογράμματα που δηλώνονται μέσα σε αυτό.

(4) Στο μη τοπικό περιβάλλον ενός υποπρογράμματος.

γράφουμε όλες τις διαφορετικές μεταβλητές που δηλώνονται σε όλα τα περιβάλλοντα υποπρογράμματα που βρίσκονται έξω από αυτό το υποπρόγραμμα και τα ονόματα των υποπρογραμμάτων που δηλώνονται στα περιβάλλοντα υποπρογράμματα.

5) Στο καθολικό περιβάλλον (εφόσον υπάρχει), γράφουμε όλες τις διαφορετικές μεταβλητές που δηλώνονται στη main, καθώς και τα υποπρογράμματα που δηλώνονται στη main.

Άσκηση

i) Παραστήστε μια BNF γραμματική η οποία παράγει όλες τις δυνατές ακολουθίες που αποτελούνται από παρενθέσεις ή/και αγκύλες σωστά φωλιασμένες.
ii) Είναι η γραμματική σας LL(1); Αν όχι παραστήστε μια ισοδύναμη γραμματική που είναι LL(1).
iii) Χρησιμοποιώντας τη γραμματική σας, κατά προτίμηση την LL(1), παραστήστε το δένδρο της συντακτικής ανάλυσης για την ακολουθία ([ ]([ ]))[ ](( ))

Λύση

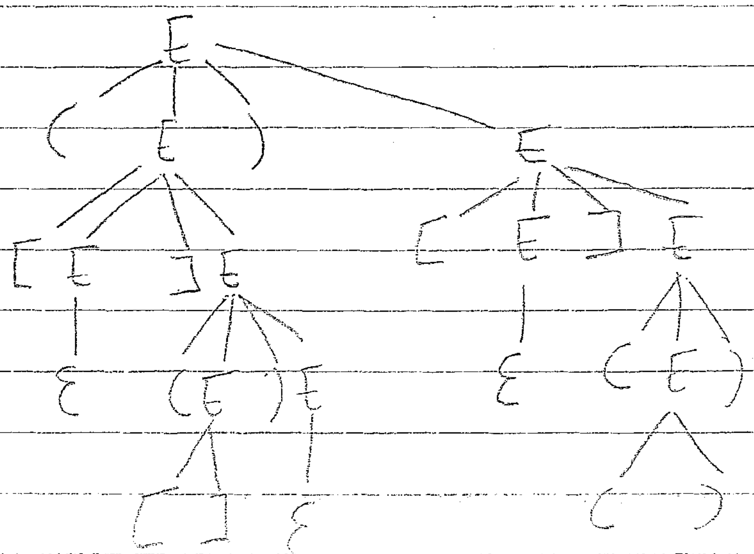
i) $E \rightarrow (E) \mid [E] \mid () \mid [\,] \mid (E)E \mid [E]E \mid \varepsilon$

ii) Η γραμματική είναι LL(1) διότι δεν παραβιάζεται καμία από τις 3 προϋποθέσεις

iii) $E \Rightarrow (E)E \Rightarrow (E)[E]E \Rightarrow (E)[E](E) \Rightarrow (E)[\,](()) \Rightarrow$

$([E]E)[\,](()) \Rightarrow ([\,](E)E)[\,](()) \Rightarrow$

$([\,]([\,]))[\,](())$



ΘΕΜΑ (Ιούνιος 1999)

Θέλουμε να ορίσουμε μια γλώσσα η οποία να αποτελείται από όλα τα strings που περιλαμβάνουν χαρακτήρες από τα 4 γράμματα α,β,γ,δ με την εξής ιδιομορφία. Τα α εμφανίζονται οπουδήποτε στο string, αλλά μόνο σε μια ακολουθία από n α ακολουθούμενη από β, όπου n ακέραιος με n>0 και χωρίς να παρεμβάλλεται μεταξύ των ακολουθιών των α και β άλλος χαρακτήρας. Για παράδειγμα, τα strings αβ, γαααβββδγ ανήκουν στην γλώσσα, αλλά όχι τα strings α, αββ, ββαα, αααβββδβ

a) Περιγράψτε το συντακτικό της γλώσσας με χρήση γραμματικής BNF

b) "Μεταφράστε" την γραμματική BNF σε συντακτικά διαγράμματα

c) Κατασκευάζοντας τα δέντρα συντακτικής ανάλυσης (parse trees) για τα strings ααβγγ και αααβββδ να αποφανθείτε αν τα δύο strings είναι μέλη της γλώσσας.

Λύση

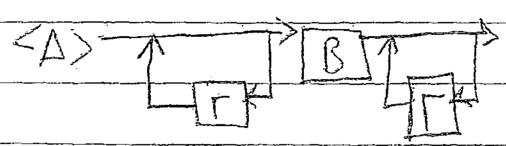
a) <Δ> ::= {<Γ>}<Β>{<Γ>}

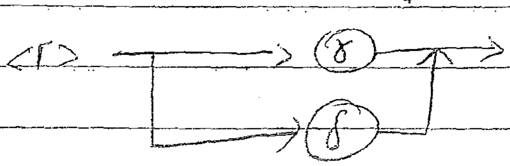
<Γ> ::= γ|δ

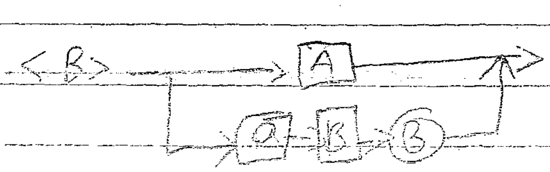
<Β> ::= <Α>|α<Β>β

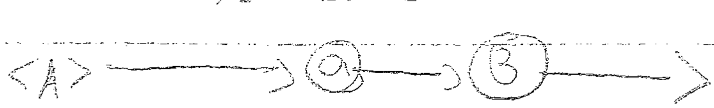
<Α> ::= αβ

b)

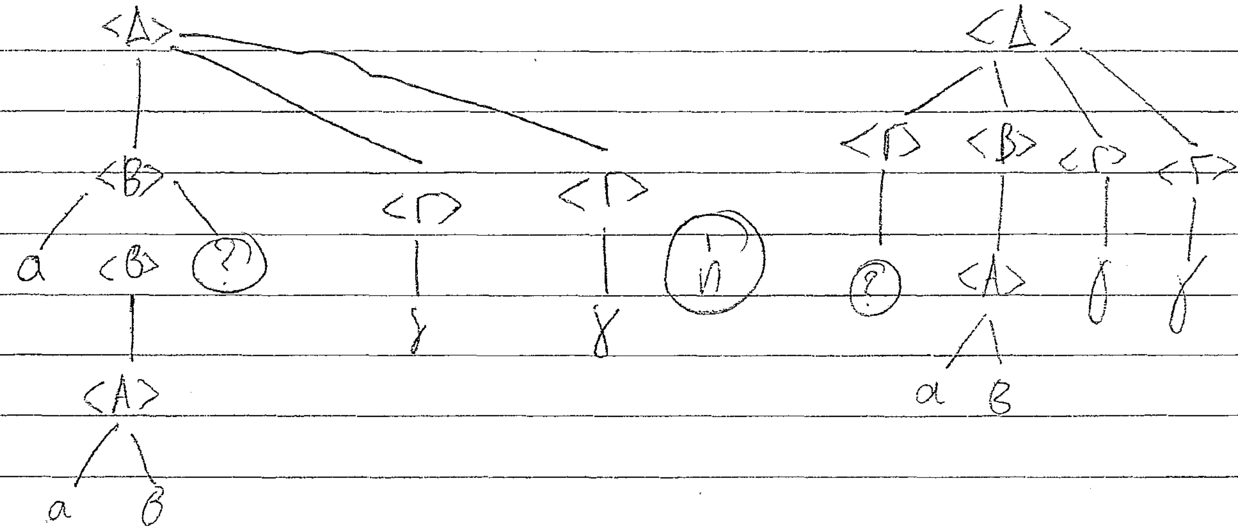








c) Parse tree για τα ααβγγγ

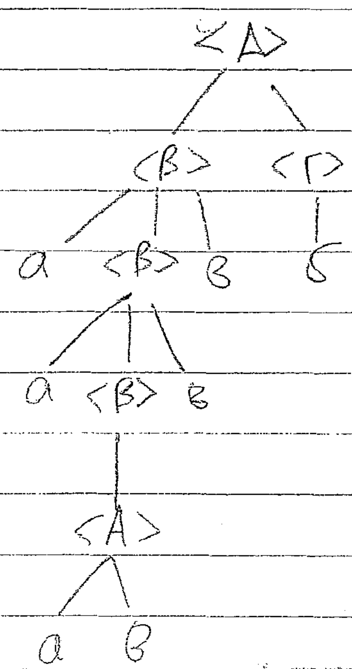


το ααβγγ δεν ανηκει στη γλωσσα

$\langle \Delta \rangle \Rightarrow \langle B \rangle \langle \Gamma \rangle \Rightarrow \langle B \rangle \delta \Rightarrow \alpha \langle B \rangle \beta\delta \Rightarrow \alpha\alpha \langle B \rangle \beta\beta\delta \Rightarrow$

$\alpha\alpha A \beta\beta\delta \Rightarrow \alpha\alpha\alpha\beta\beta\beta\delta$

Parse tree για το αααβββδ



το αααβββδ ανηκει στην γλωσσα