

Μεταγλωττιστές #5

Νίκος Παπασπύρου
nickie@softlab.ntua.gr



Εθνικό Μετσόβιο Πολυτεχνείο
Τμήμα Ηλεκτρολόγων Μηχ. και Μηχ. Υπολογιστών
Εργαστήριο Τεχνολογίας Λογισμικού
Πολυτεχνειούπολη, 15780 Ζωγράφου.

Πίνακας συμβόλων

- Συγκεκριώνει πληροφορίες για τα **ονόματα** που εμφανίζονται στο αρχικό πρόγραμμα
- Ονόματα είναι:
 - το **πρόγραμμα**
 - οι **μεταβλητές**
 - τα **υποπρογράμματα** (διαδικασίες, συναρτήσεις)
 - οι **παράμετροι** των υποπρογραμμάτων
 - οι **ετικέτες** εντολών
 - οι **σταθερές**
 - οι **τύποι δεδομένων**



Χαρακτηριστικά ονομάτων

- Κατηγορία αποθήκευσης (storage class)
 - Καθολικές μεταβλητές (global variables)
 - Μεταβλητές στοίβας (stack variables)
 - Στατικές μεταβλητές (static variables)
- Εμβέλεια (scope)
- Ορατότητα (visibility)
- Διάρκεια ζωής (lifetime)



Περιεχόμενα πίνακα συμβόλων

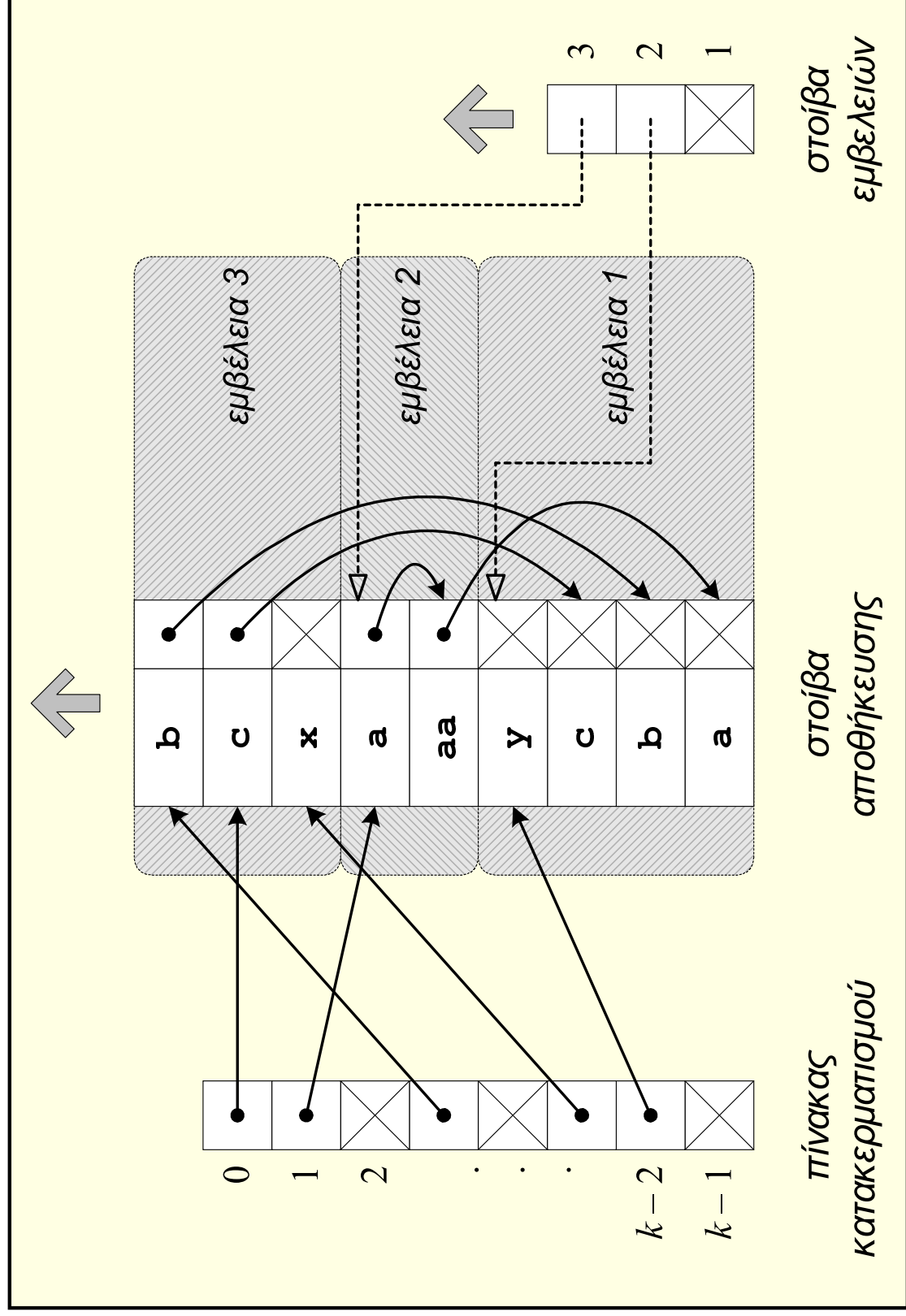
- Εμβέλεια (έμμεσα)
- Ορατότητα (έμμεσα)
- Διάρκεια ζωής
- Τύπος
- Θέση (διεύθυνση μνήμης, καταχωρητής, ...)
- Αριθμός παραμέτρων υποπρογράμματος
- Τύπος παραμέτρων υποπρογράμματος
- Τρόπος περάσματος παραμέτρων υποπρογράμματος
- Τύπος αποτελέσματος συνάρτησης

Οργάνωση πίνακα συμβόλων

- Βασικές λειτουργίες
 - Προσθήκη ονόματος
 - Αναζήτηση ονόματος
 - Διαγραφή ονόματος ή ομάδας ονομάτων
- Κόστος προσθήκης ή αναζήτησης ανάλογα με την υλοποίηση:

γραμμική λίστα	$O(n)$
δυσადικό δέντρο αναζήτησης	$O(\log n)$
πίνακας κατακερματισμού	$O(n/k)$

Υλοποίηση με ΠΚ





Σύνταξη και σημασιολογία

- **Σύνταξη**: μορφή και δομή των καλώς σχηματισμένων προγραμμάτων
- **Σημασιολογία**: ερμηνεία των καλώς σχηματισμένων προγραμμάτων
 - **Στατική** σημασιολογία: εντοπισμός σημασιολογικών σφαλμάτων κατά τη διάρκεια της μεταγλώττισης
 - **Δυναμική** σημασιολογία: απόδοση ερμηνείας στα προγράμματα κατά την εκτέλεσή τους



Στατική σημασιολογία (i)

- Περιβάλλοντα τύπων

$$\Gamma_1 = \{ i \mapsto integer, x \mapsto real \}$$

- Σχέση αντιστοίχισης τύπων

$$\Gamma \vdash E : \tau$$

- Κανόνες τύπων

$$\frac{\Gamma \vdash E_1 : integer \quad \Gamma \vdash E_2 : integer}{\Gamma \vdash E_1 + E_2 : integer}$$

Στατική σημασιολογία

(ii)

- Παραγωγές τύπων

$$\frac{\Gamma_1 \vdash i : integer \quad \Gamma_1 \vdash 1 : integer}{\Gamma_1 \vdash i+1 : integer} \quad \frac{\Gamma_1 \vdash i+1 : integer \quad \Gamma_1 \vdash x : real}{\Gamma_1 \vdash (i+1)*x : real}$$

- Η αντιστοίχιση τύπων με κανόνες τύπων επεκτείνεται σε όλα τα τμήματα προγράμματος

$$\frac{\Gamma \vdash E : boolean \quad \Gamma \vdash S : stmt}{\Gamma \vdash \text{while } E \text{ do } S : stmt}$$

Δυναμική σημασιολογία (i)

- Λειτουργική σημασιολογία (operational semantics)
⇒ ακολουθία υπολογιστικών βημάτων
- Δηλωτική σημασιολογία (denotational semantics)
⇒ μαθηματική συνάρτηση από το πεδίο των δεδομένων εισόδου στο πεδίο των αποτελεσμάτων
- Αξιωματική σημασιολογία (axiomatic semantics)
⇒ η ερμηνεία καθορίζεται έμμεσα μέσω λογικών προτάσεων που περιγράφουν ιδιότητες του προγράμματος

Δυναμική σημασιολογία (ii)

- Η εντολή ανάθεσης $I=E$
- Λειτουργική σημασιολογία

$$\frac{\langle E, \sigma \rangle \rightarrow v}{\langle I=E, \sigma \rangle \rightarrow \sigma[I \mapsto v]}$$

- Δηλωτική σημασιολογία
 - Αξιωματική σημασιολογία
- $$\{ P [I \mapsto E] \} \quad I=E \quad \{ P \}$$



Σημσιολογικός έλεγχος

- Έλεγχος τύπων
- Έλεγχος ροής
- Έλεγχος ύπαρξης ονομάτων
- Έλεγχος μοναδικότητας
- Έλεγχος συνέπειας



Σύστημα τύπων

(*i*)

- Βασικοί τύποι (integer, boolean, real, char, ...)
- Σύνθετοι τύποι
 - Πίνακες (arrays)
 - Ζεύγη (products) και *πλειάδες* (tuples)
 - Εγγραφές (records)
 - Δείκτες (pointers)
 - Συναρτήσεις (functions)
- Τύποι πρώτης τάξης (*first class*)

Σύστημα τύπων

(ii)

- Μετατροπές τύπων (type casting)
- Υπερφόρτωση τελεστών (operator overloading)
- Πολυμορφικοί τελεστές (polymorphic operators)
- Υποσύνολα τύπων και υπο-τύποι (subtypes)
- Πολυμορφικά συστήματα τύπων
(polymorphic type systems)
- Στατική και δυναμική αντιστοίχιση τύπων
(type binding)
- Εξαγωγή τύπων (type inference)

Δυναμικός έλεγχος τύπων

- Επιβάλλεται όταν υπάρχει **δυναμική αντιστοίχιση τύπων**
- Πολλές φορές όμως απαιτείται και σε **στατική αντιστοίχιση τύπων**, π.χ. έλεγχος ορίων σε arrays της Pascal:

`a[i] := 42`

αν $i \geq 0$ και $i \leq 100$ τότε

κώδικας για την ανάθεση του 42 στο `a[i]`
αλλιώς

σφάλμα εκτέλεσης
τέλος αν

Ενδιάμεσος κώδικας

(i)

- Λόγοι ύπαρξης
 - Διευκολύνει το έργο της μετάφρασης
 - Διευκολύνει τη βελτιστοποίηση
 - Διευκολύνει την κατάτμηση σε εμπρόσθιο και οπίσθιο τμήμα



Ενδιάμεσος κώδικας

(ii)

- Μετάφραση οδηγούμενη από τη σύνταξη (syntax-directed translation)
 - Για κάθε δομή της γλώσσας προσδιορίζεται ο αντίστοιχος ενδιάμεσος κώδικας
 - Διευρύνεται ο συντακτικός αναλυτής με σημασιολογικές ρουτίνες που παράγουν ενδιάμεσο κώδικα
- Σχέδιο παραγωγής ενδιάμεσου κώδικα
- Μεταβλητές ιδιοτήτων (attributes) για κάθε σύμβολο της γραμματικής

Ενδιάμεση γλώσσα

(i)

- Τετράδες (quadruples)

n : op, x, y, z

- Παράδειγμα:

$b * b - 4 * a * c$

- 1: $*, b, b, \$1$
- 2: $*, 4, a, \$2$
- 3: $*, \$2, c, \3
- 4: $-, \$1, \$3, \$4$

Ενδιάμεση γλώσσα

(ii)

- Τριάδες (triples)

n : op, x, y

- Παράδειγμα:

$b * b - 4 * a * c$

1: $*, b, b$

2: $*, 4, a$

3: $*, (2), c$

4: $-, (1), (3)$

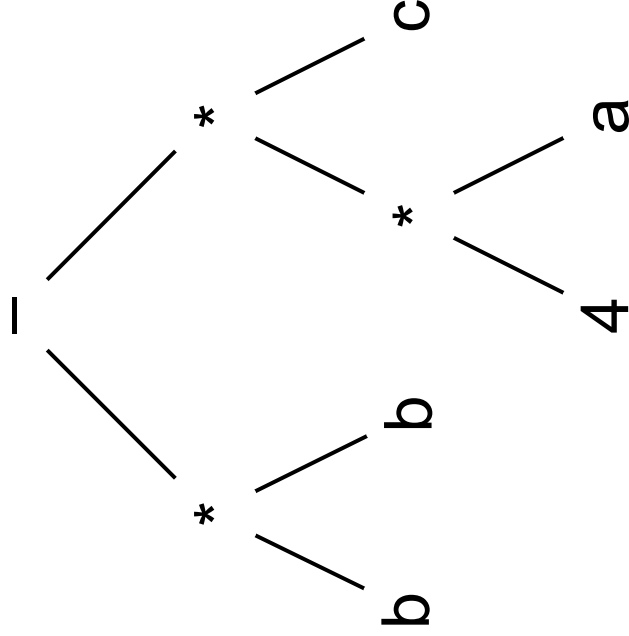


Ενδιάμεση γλώσσα

(iii)

- Αφηρημένα συντακτικά δέντρα
(abstract syntax trees)
- Παράδειγμα:

$b * b - 4 * a * c$



Ενδιάμεση γλώσσα

(iv)

- Προθεματικός και επιθεματικός κώδικας
(prefix/postfix code)

- Παράδειγμα:

$b * b - 4 * a * c$

— * b b * * 4 a c

προθεματικός

b b * 4 a * c * —

επιθεματικός

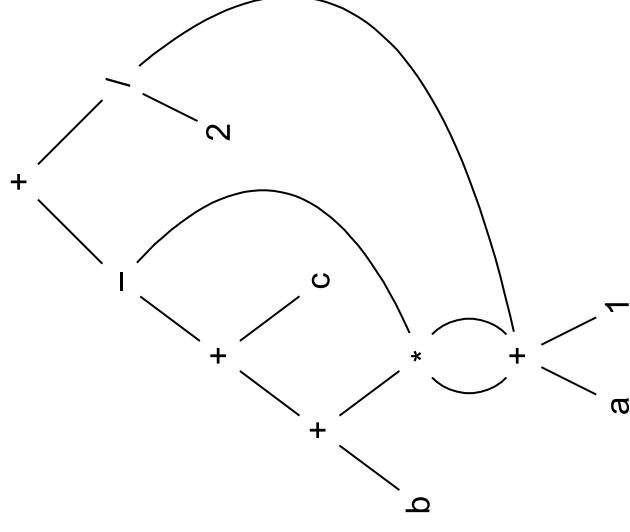
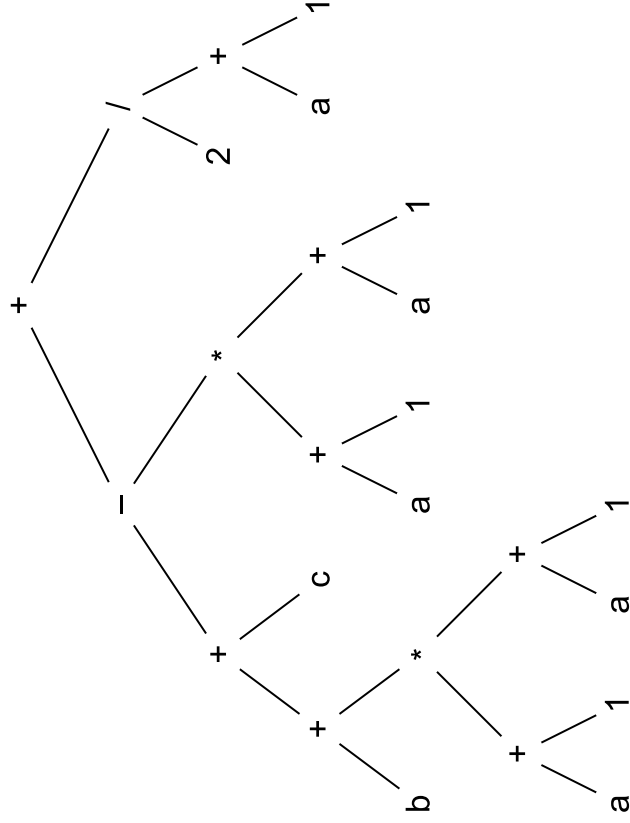
Ενδιάμεση γλώσσα

(v)

- Κατευθυνόμενοι ακυκλικοί γράφοι
(directed acyclic graphs)

- Παράδειγμα:

$$b+(a+1)*c-(a+1)+2/(a+1)$$



Γλώσσα τετράδων

- Μορφή τετράδας:

$n: op, x, y, z$

όπου:

- n : ετικέτα τετράδας (φυσικός αριθμός)
- op : τελεστής
- x, y, z : τελούμενα

- Ανάλογα με το είδος του τελεστή, κάποια τελούμενα ενδεχομένως παραλείπονται

Τελούμενα

(i)

- Σταθερά
 - ακέραια, πραγματική, λογική
 - χαρακτηρισ, συμβολοσειρά, nil
- Όνομα
 - μεταβλητή
 - παραμέτρος υποπρογράμματος
 - υποπρόγραμμα
- Προσωρινή μεταβλητή: \$n



Τελοούμενα

(ii)

- Ετικέτα
 - εντολής στο αρχικό πρόγραμμα
 - τετράδας
- Τρόπος περάσματος
 - V : κατ' αξία
 - R : κατ' αναφορά
 - RET : θέση αποτελέσματος συνάρτησης
- Κενό : —
- Προσωρινά κενό : * (για backpatching)

Τελεστές

(i)

- **unit**, I , $-$, $-$

- **endu**, I , $-$, $-$

αρχή και τέλος δομικής μονάδας

- **op**, x , y , z $op \in \{+, -, *, /, \%\}$

$z := x \text{ op } y$

- **:=**, x , $-$, z

$z := x$

- **op**, x , y , z $op \in \{=, <, >, <=, <=\}$

αν $x \text{ op } y$ τότε πήγαινε στην τετράδα z

Τελεστές

(ii)

- **ifb**, $x, -, z$
αν η λογική τιμή x είναι αληθής τότε πήγαινε στην τετράδα z
- **jump**, $-, -, z$
πήγαινε στην τετράδα z
- **call**, $-, -, I$
κάλεσε τη δομική μονάδα I

Τελεστές

(iii)

- `par`, x , m , —

πέρασε την πραγματική παράμετρο x με τρόπο περάσματος m

- `retv`, x , —, —

κάνε το αποτέλεσμα της τρέχουσας συνάρτησης ίσο με x

- `ret`, —, —, —

επιστροφή από την τρέχουσα δομική μονάδα



Μεταβλητές ιδιοτήτων

- *PLACE*: θέση όπου βρίσκεται αποθηκευμένη η τιμή μιας l-value ή μιας r-value
- *TYPE*: τύπος μιας l-value ή μιας r-value
- *NEXT*: λίστα από ετικέτες τετράδων που περιέχουν άλματα στην επόμενη εντολή
- *TRUE, FALSE*: λίστες από ετικέτες τετράδων που περιέχουν άλματα στον κώδικα που πρέπει να εκτελεστεί αν μια συνθήκη είναι αληθής ή ψευδής

Βοηθητικές υπορουτίνες (i)

- NEXTQUAD()

Επιστρέφει τον αριθμό της επόμενης τετράδας

- GENQUAD(op, x, y, z)

Γεννά την επόμενη τετράδα op, x, y, z

- NEWTEMP(t)

Δημιουργεί μια νέα προσωρινή μεταβλητή τύπου t

- EMPTYLIST()

Δημιουργεί μια κενή λίστα ετικετών τετράδων

Βοηθητικές υπορουτίνες (ii)

- MAKELIST(x)

Δημιουργεί μια λίστα ετικετών τετράδων που περιέχει **μόνο το στοιχείο** x

- MERGE(l_1, \dots, l_n)

Συνένωση των λιστών ετικετών τετράδων $l_1 \dots l_n$

- BACKPATCH(l, z)

Αντικαθιστά σε όλες τις τετράδες που περιέχονται στην l την άγνωστη ετικέτα τετράδας με τη z
(**backpatching**)

Αριθμητικές εκφράσεις

- Ακέραιες σταθερές

$\langle \text{r-value} \rangle ::= \langle \text{integer-const} \rangle \{ P_1 \}$

$P_1 : \{ \langle \text{r-value} \rangle.PLACE = \langle \text{integer-const} \rangle; \}$

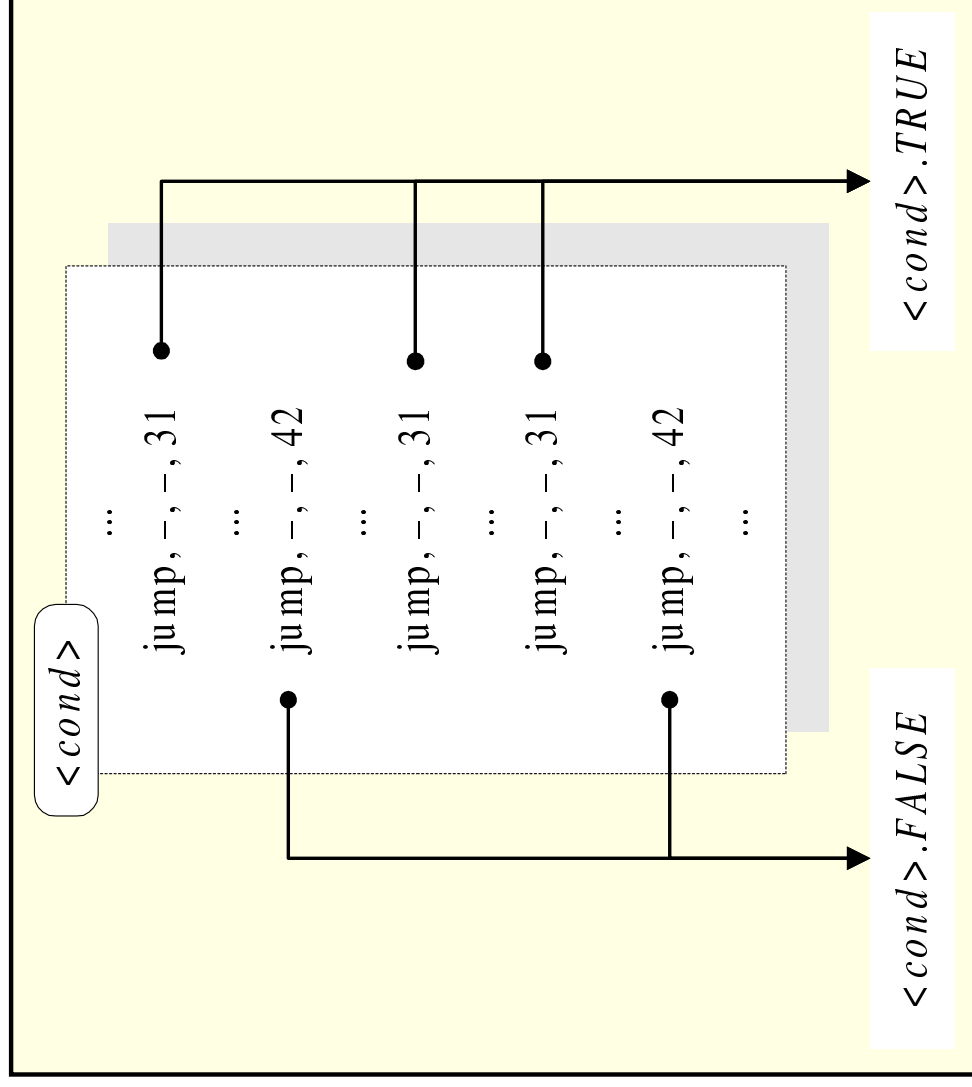
- Τελεστές με δύο τελούμενα

$\langle \text{r-value} \rangle ::= \langle \text{expr} \rangle \langle \text{binop} \rangle \langle \text{expr} \rangle \{ P_{14} \}$

$P_{14} : \{ W = \text{NEWTEMP}(\langle \text{r-value} \rangle.TYPE);$
GENQUAD($\langle \text{binop} \rangle.NAME,$
 $\langle \text{expr} \rangle^1.PLACE,$
 $\langle \text{expr} \rangle^2.PLACE, W);$
 $\langle \text{r-value} \rangle.PLACE = W; \}$

Λογικές εκφράσεις

(i)

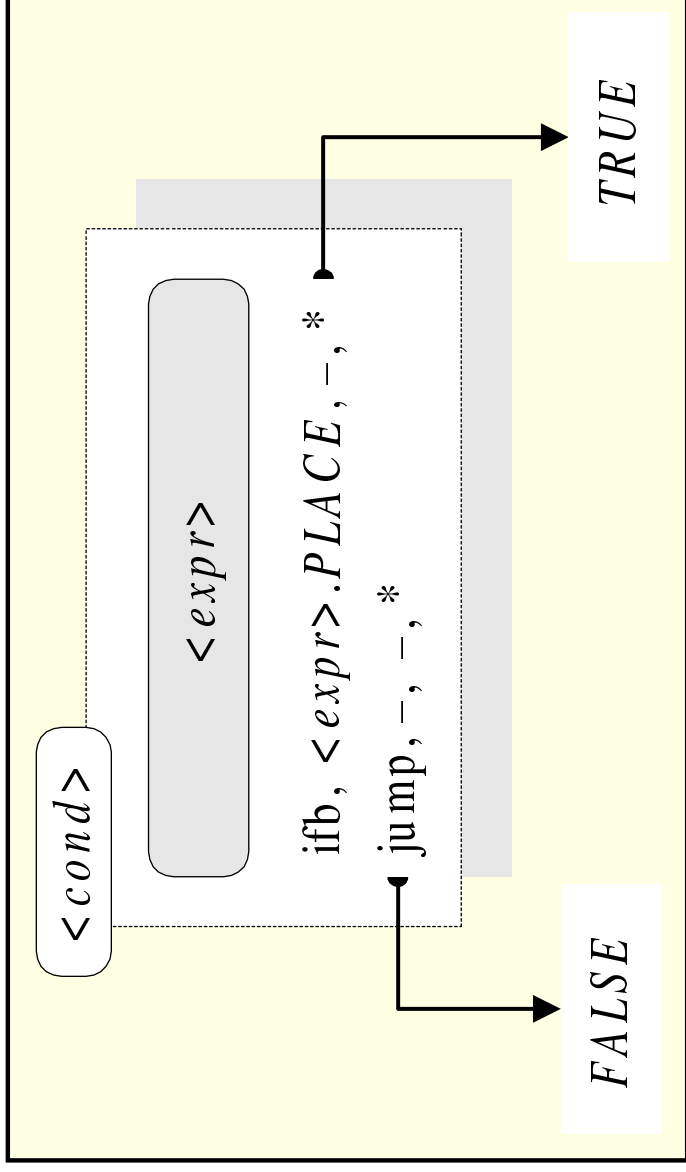


Λογικές εκφράσεις

(ii)

- Λογικές εκφράσεις σε συμβολισμό 0/1

$\langle \text{cond} \rangle ::= \langle \text{expr} \rangle$



Λογικές εκφράσεις

(iii)

- Λογικές εκφράσεις σε συμβολισμό 0/1

$\langle \text{cond} \rangle ::= \langle \text{expr} \rangle \{ P_{21} \}$

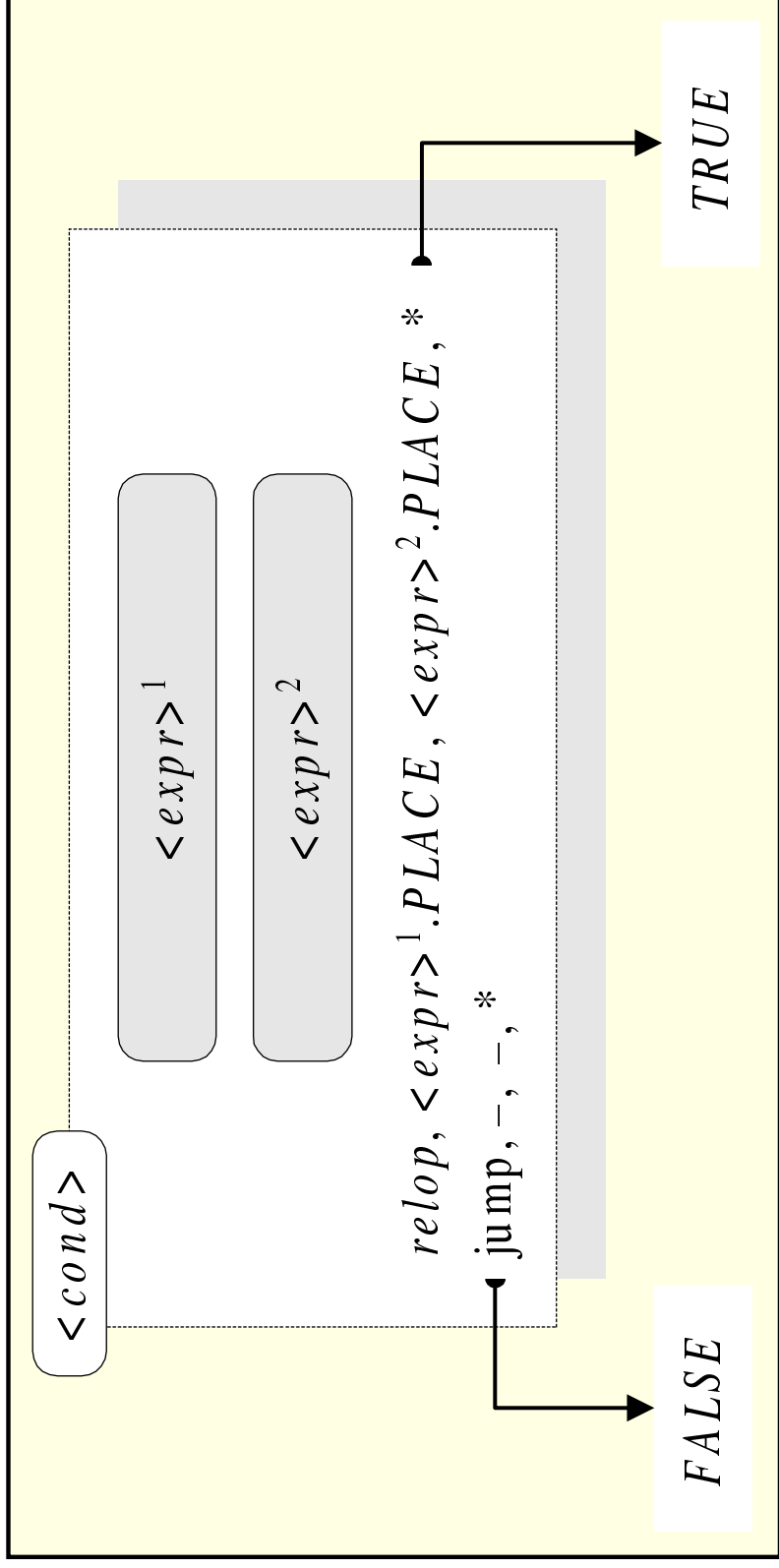
$P_{21} : \{ \langle \text{cond} \rangle . \text{TRUE} = \text{MAKELIST}(\text{NEXTQUAD}());$
 $\text{GENQUAD}(\text{ifb}, \langle \text{expr} \rangle . \text{PLACE}, -, *);$
 $\langle \text{cond} \rangle . \text{FALSE} = \text{MAKELIST}(\text{NEXTQUAD}());$
 $\text{GENQUAD}(\text{jump}, -, -, *); \}$

Λογικές εκφράσεις

(iv)

- Τελεστές σύγκρισης

$\langle \text{cond} \rangle ::= \langle \text{expr} \rangle^1 \langle \text{relop} \rangle \langle \text{expr} \rangle^2$



Λογικές εκφράσεις

(v)

■ Τελεστές σύγκρισης

$\langle \text{cond} \rangle ::= \langle \text{expr} \rangle^1 \langle \text{relop} \rangle \langle \text{expr} \rangle^2 \{ P_{23} \}$

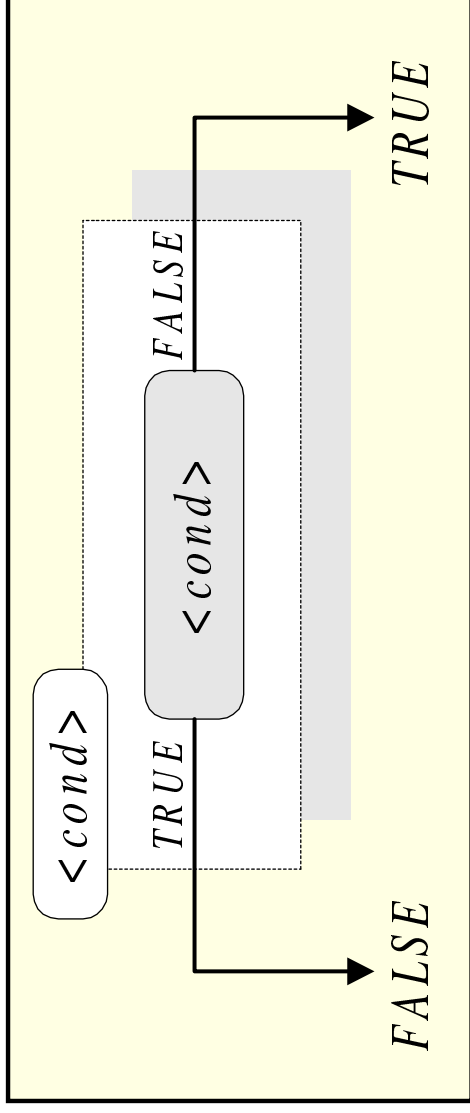
$P_{23} : \{ \langle \text{cond} \rangle . TRUE = \text{MAKELIST}(\text{NEXTQUAD}());$
 $\text{GENQUAD}(\langle \text{relop} \rangle . NAME,$
 $\langle \text{expr} \rangle^1 . PLACE,$
 $\langle \text{expr} \rangle^2 . PLACE, *);$
 $\langle \text{cond} \rangle . FALSE = \text{MAKELIST}(\text{NEXTQUAD}());$
 $\text{GENQUAD}(\text{jump}, -, -, *); \}$

Λογικές εκφράσεις

(vi)

■ Άρνηση

$\langle \text{cond} \rangle ::= \text{“not” } \langle \text{cond} \rangle$

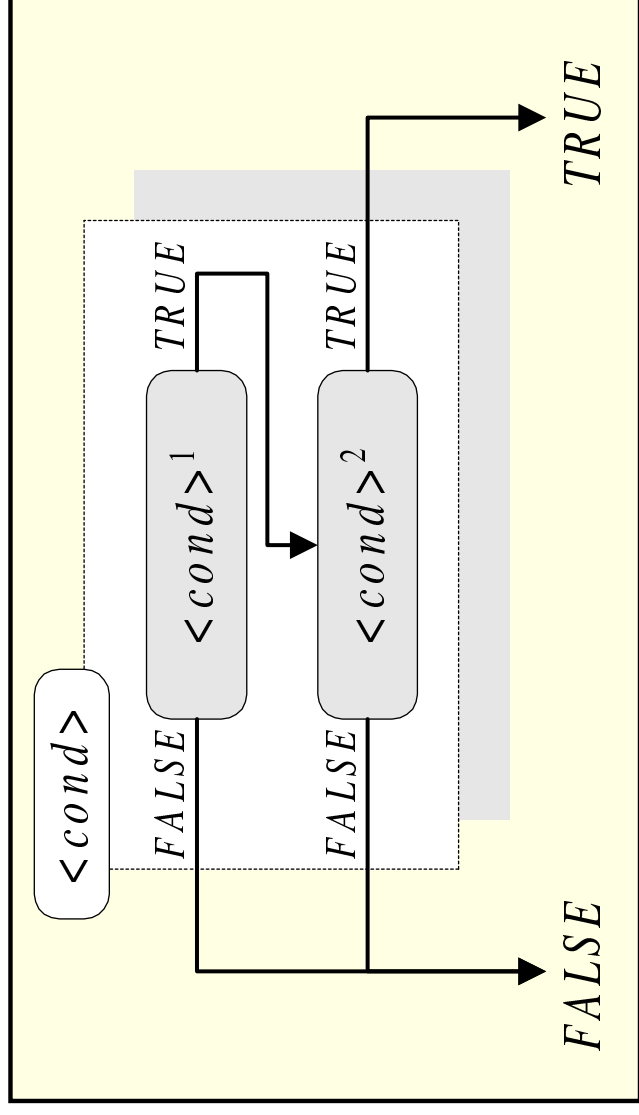


Λογικές εκφράσεις

(vii)

■ Σύζευξη

$\langle \text{cond} \rangle ::= \langle \text{cond} \rangle^1 \text{ “and” } \langle \text{cond} \rangle^2$



Λογικές εκφράσεις

(viii)

■ Σύζευξη

$\langle \text{cond} \rangle ::= \langle \text{cond} \rangle_1$ “and” $\{ P_{25} \} \langle \text{cond} \rangle_2 \{ P_{26} \}$

$P_{25} : \{ \text{BACKPATCH}(\langle \text{cond} \rangle^1. \text{TRUE}, \text{NEXTQUAD}()); \}$

$P_{26} : \{ \langle \text{cond} \rangle. \text{FALSE} = \text{MERGE}(\langle \text{cond} \rangle^1. \text{FALSE},$
 $\qquad \qquad \qquad \langle \text{cond} \rangle^2. \text{FALSE});$

$\langle \text{cond} \rangle. \text{TRUE} = \langle \text{cond} \rangle^2. \text{TRUE}; \}$