

---

# Δομές Ενδιάμεσου Κώδικα

Διαλέξεις στο μάθημα: Μεταφραστές  
Γεώργιος Μανής

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ  
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ  
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING  
UNIVERSITY OF IOANNINA



## *Αρχή και Τέλος Block*

---

|                 |     |  |
|-----------------|-----|--|
| <program>       | ::= | <b>program</b> name<br><program_block>         |
| <program_block> | ::= | <declarations><br><subprograms><br>< block>    |
| <subprograms>   | ::= | <b>function</b> id <formalpars><br>{ <block> } |

## Αρχή και Τέλος Block

---

**program** P

**function** a()

{ code for a }

**function** b()

{ code for b }

{ code for P }



begin\_block a \_ \_

... intermediate code for a

end\_block a \_ \_

begin\_block b \_ \_

... intermediate code for b

end\_block b \_ \_

begin\_block P \_ \_

... intermediate code for P

halt \_ \_ \_

end\_block P \_ \_

---

## *Αρχή και Τέλος Block*

---

|                 |     |  |
|-----------------|-----|--|
| <program>       | ::= | <b>program</b> name<br><program_block>         |
| <program_block> | ::= | <declarations><br><subprograms><br>< block>    |
| <subprograms>   | ::= | <b>function</b> id <formalpars><br>{ <block> } |

## *Αρχή και Τέλος Block*

---

|                 |     |   |
|-----------------|-----|---|
| <program>       | ::= | <b>program</b> name<br><program_block>  |
| <program_block> | ::= | <declarations><br><subprograms><br><b>genquad</b> ("begin_block",name,"_","_")<br>< block><br><b>genquad</b> ("end_block",name,"_","_") |
| <subprograms>   | ::= | <b>function</b> id <formalpars><br>{ <block> }  |

## *Αρχή και Τέλος Block*

---

|                 |     |   |
|-----------------|-----|---|
| <program>       | ::= | <b>program</b> name<br><program_block>  |
| <program_block> | ::= | <declarations><br><subprograms><br><b>genquad</b> ("begin_block",name,"_","_")<br>< block><br><b>genquad</b> ("end_block",name,"_","_")   |
| <subprograms>   | ::= | <b>function</b> id <formalpars><br>{ <b>genquad</b> ("begin_block",id,"_","_")<br><block><br><b>genquad</b> ("end_block",id,"_","_")<br>} |

## Αρχή και Τέλος Block

---

|                 |     |  |
|-----------------|-----|--|
| <program>       | ::= | <b>program</b> name<br><program_block>   |
| <program_block> | ::= | <declarations><br><subprograms><br>genquad("begin_block",name,"_","_")<br>< block><br>genquad("halt","_","_","_")<br>genquad("end_block",name,"_","_") |
| <subprograms>   | ::= | <b>function</b> id <formalpars><br>{ genquad("begin_block",id,"_","_")<br><block><br>genquad("end_block",id,"_","_")<br>}                              |

## Αρχή και Τέλος Block

---

|                               |     |  |
|-------------------------------|-----|--|
| <program>                     | ::= | <b>program</b> name<br><program_block> <b>(name)</b>   |
| <program_block> <b>(name)</b> | ::= | <declarations><br><subprograms><br><b>genquad</b> ("begin_block",name,"_","_")<br>< block><br><b>genquad</b> ("halt","_","_","_")<br><b>genquad</b> ("end_block",name,"_","_") |
| <subprograms>                 | ::= | <b>function</b> id <formalpars><br>{ <b>genquad</b> ("begin_block",id,"_","_")<br><block><br><b>genquad</b> ("end_block",id,"_","_")<br>}                                      |



## Αριθμητικές Παραστάσεις

---

Παράδειγμα:

$$x + (y + z) \times w$$

ενδιάμεσος κώδικας:

1: +, y, z, T\_1

2: ×, T\_1, w, T\_2

3: +, x, T\_2, T\_3

## *Αριθμητικές Παραστάσεις*

---

Έστω η γραμματική:

$$E \rightarrow T^1 ( + T^2 )^*$$

$$T \rightarrow F^1 ( \times F^2 )^*$$

$$F \rightarrow ( E )$$

$$F \rightarrow id$$

---

## *Αριθμητικές Παραστάσεις*

---

$$E \rightarrow T^1 ( + T^2 )^*$$

## Αριθμητικές Παραστάσεις

---

$E \rightarrow T^1 ( + T^2 \{P_1\} )^*$

$\{P_1\}$ :      $w = \text{newTemp}()$   
               $\text{genquad}("+", T^1.\text{place}, T^2.\text{place}, w)$   
               $T^1.\text{place} = w$

## Αριθμητικές Παραστάσεις

---

$E \rightarrow T^1 ( + T^2 \{P_1\})^* \{P_2\}$

$\{P_1\}$ :      $w = \text{newTemp}()$   
               $\text{genquad}("+", T^1.\text{place}, T^2.\text{place}, w)$   
               $T^1.\text{place} = w$

$\{P_2\}$ :      $E.\text{place} = T^1.\text{place}$

---

## Αριθμητικές Παραστάσεις

$E \rightarrow T^1 ( + T^2 \{P_1\} )^* \{P_2\}$

$\{P_1\}$ :  $w = \text{newTemp}()$

Νέα προσωρινή μεταβλητή που θα κρατήσει το μέχρι στιγμής αποτέλεσμα

$\text{genquad}("+", T^1.\text{place}, T^2.\text{place}, w)$

Παραγωγή τετράδας που προσθέτει το μέχρι στιγμής αποτέλεσμα στο νέο  $T^2$

$T^1.\text{place} = w$

Το μέχρι στιγμής αποτέλεσμα τοποθετείται στην  $T^1$  ώστε να χρησιμοποιηθεί αν υπάρξει επόμενο  $T^2$

$\{P_2\}$ :  $E.\text{place} = T^1.\text{place}$

Όταν δεν υπάρχει άλλο  $T^2$  το αποτέλεσμα είναι στο  $T^1$

## Αριθμητικές Παραστάσεις

---

$T \rightarrow F^1 (\times F^2 \{P_1\})^* \{P_2\}$

$\{P_1\}$ :      $w = \text{newTemp}()$

$\text{genquad}(\text{"\times"}, F^1.\text{place}, F^2.\text{place}, w)$

$F^1.\text{place} = w$

$\{P_2\}$ :      $T.\text{place} = F^1.\text{place}$

Ανάλογη λογική με τον κανόνα E

---

## Αριθμητικές Παραστάσεις

---

$F \rightarrow ( E ) \{P_1\}$

Απλή μεταφορά από το E.place στο F.place

$\{P_1\}: F.place = E.place$

$F \rightarrow id \{P_1\}$

Απλή μεταφορά από το id.place στο F.place

$\{P_1\}: F.place = id.place$

---



## *Αριθμητικές Παραστάσεις*

---

```
procedure E (E.place)  
  begin  
    T ( T1.place )  
    while token=plustk do begin  
      lex();  
      T (T2.place)  
      w:=newTemp()  
      genquad( "+", T1.place, T2.place, w)  
      T1.place :=w  
    end  
    E.place := T1.place  
  end
```

## Λογικές Παραστάσεις

---

Έστω η γραμματική:

$B \rightarrow Q \text{ ( or } Q \text{ )}^*$

$Q \rightarrow R \text{ ( and } R \text{ )}^*$

$R \rightarrow ( B )$

$R \rightarrow E \text{ relop } E$

---

## Λογικές Παραστάσεις - OR

---

Παράδειγμα:

$x > y$  or  $x < w$

100:  $>, x, y, \_$

101: jump,  $\_, \_, 102$

102:  $<, x, w, \_$

103: jump,  $\_, \_, \_$

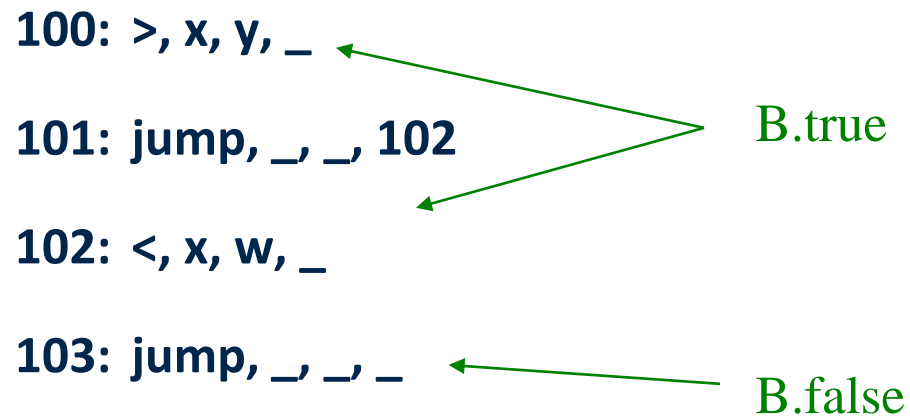
---

## Λογικές Παραστάσεις - OR

---

Παράδειγμα:

$B = x > y \text{ or } x < w$

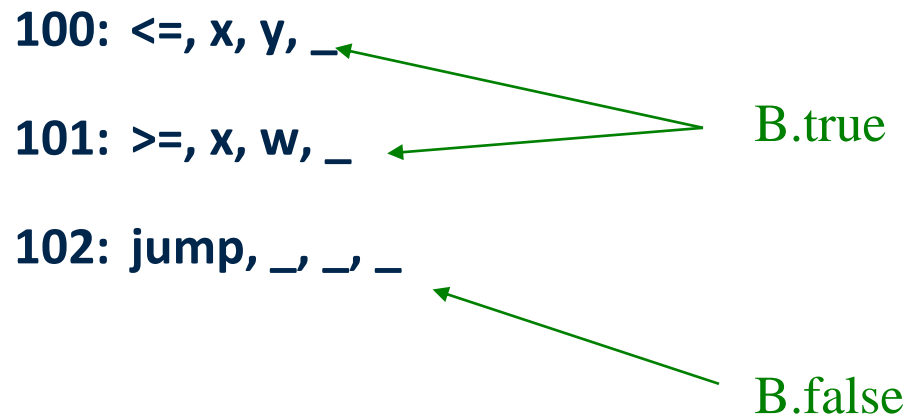


## Λογικές Παραστάσεις - OR

---

Παράδειγμα:

$B = x > y \text{ or } x < w$



## *Λογικές Παραστάσεις - OR*

---

$B \rightarrow Q^1 \text{ ( or } Q^2 \text{ )}^*$

## Λογικές Παραστάσεις - OR

---

$B \rightarrow Q^1 \{P_1\} ( \text{ or } Q^2 )^*$

$\{P_1\}$ :       $B.\text{true} = Q^1.\text{true}$

$B.\text{false} = Q^1.\text{false}$

## Λογικές Παραστάσεις - OR

---

$B \rightarrow Q^1 \{P_1\} ( \text{ or } \{P_2\} Q^2 )^*$

$\{P_1\}$ :       $B.\text{true} = Q^1.\text{true}$

$B.\text{false} = Q^1.\text{false}$

$\{P_2\}$ :       $\text{backpatch}(B.\text{false}, \text{nextquad}())$



## Λογικές Παραστάσεις - OR

---

$B \rightarrow Q^1 \{P_1\} ( \text{ or } \{P_2\} Q^2 \{P_3\} )^*$

$\{P_1\}$ :       $B.\text{true} = Q^1.\text{true}$

$B.\text{false} = Q^1.\text{false}$

$\{P_2\}$ :       $\text{backpatch}(B.\text{false}, \text{nextquad}())$

$\{P_3\}$ :       $B.\text{true} = \text{merge}(B.\text{true}, Q^2.\text{true})$

$B.\text{false} = Q^2.\text{false}$

---

## Λογικές Παραστάσεις - OR

$B \rightarrow Q^1 \{P_1\} ( \text{or } \{P_2\} Q^2 \{P_3\} )^*$

$\{P_1\}$ :  $B.\text{true} = Q^1.\text{true}$

$B.\text{false} = Q^1.\text{false}$

$\{P_2\}$ :  $\text{backpatch}(B.\text{false}, \text{nextquad}())$

$\{P_3\}$ :  $B.\text{true} = \text{merge}(B.\text{true}, Q^2.\text{true})$

$B.\text{false} = Q^2.\text{false}$

Μεταφορά των τετράδων  
από τη λίστα  $Q^1$  στη λίστα  $B$

Συμπλήρωση όσων τετράδων μπορούν  
να συμπληρωθούν μέσα στον κανόνα

Συσώρευση στη λίστα  $\text{true}$  των τετράδων  
που δεν μπορούν να συμπληρωθούν και  
αντιστοιχούν σε αληθή αποτίμηση  
λογικής παράστασης

Η λίστα  $\text{false}$  περιέχει την τετράδα η οποία  
αντιστοιχεί σε στη μη αληθή αποτίμηση της  
λογικής παράστασης

## Λογικές Παραστάσεις - AND

$Q \rightarrow R^1 \{P_1\} ( \text{and } \{P_2\} R^2 \{P_3\} )^*$

$\{P_1\}$ :  $Q.\text{true} = R^1.\text{true}$

$Q.\text{false} = R^1.\text{false}$

$\{P_2\}$ :  $\text{backpatch}(Q.\text{true}, \text{nextquad}())$

$\{P_3\}$ :  $Q.\text{false} = \text{merge}(Q.\text{false}, R^2.\text{false})$

$Q.\text{true} = R^2.\text{true}$

Μεταφορά των τετράδων  
από τη λίστα  $R^1$  στη λίστα  $Q$

Συμπλήρωση όσων τετράδων μπορούν  
να συμπληρωθούν μέσα στον κανόνα

Συσώρευση στη λίστα false των τετράδων  
που δεν μπορούν να συμπληρωθούν και  
αντιστοιχούν σε μη αληθή αποτίμηση  
λογικής παράστασης

Η λίστα true περιέχει την τετράδα η οποία  
αντιστοιχεί σε στην αληθή αποτίμηση της  
λογικής παράστασης

## *Λογικές Παραστάσεις*

---

**R -> ( B )**

## Λογικές Παραστάσεις

---

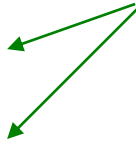
$R \rightarrow (B) \{P_1\}$

$\{P_1\}$ :

$R.true = B.true$

$R.false = B.false$

Μεταφορά των τετράδων  
από τη λίστα B στη λίστα R



## *Λογικές Παραστάσεις*

---

**R -> not ( B )**

## Λογικές Παραστάσεις

---

$R \rightarrow \text{not } ( B ) \{ P_1 \}$

$\{ P_1 \}$ :  $R.\text{true} = B.\text{false}$

$R.\text{false} = B.\text{true}$

Αντιστροφή και μεταφορά  
τετράδων από τη λίστα B στη λίστα R



## *Λογικές Παραστάσεις*

---

**R -> E<sup>1</sup> relop E<sup>2</sup>**



## Λογικές Παραστάσεις

$R \rightarrow E^1 \text{ relop } E^2 \{P_1\}$

$\{P_1\}$ :  $R.\text{true} = \text{makelist}(\text{nextquad}())$

$\text{genQuad}(\text{relop}, E^1.\text{place}, E^2.\text{place}, \text{"\_"} )$

$R.\text{false} = \text{makelist}(\text{nextquad}())$

$\text{genQuad}(\text{"jump"}, \text{"\_"}, \text{"\_"}, \text{"\_"} )$

Δημιουργία μη συμπληρωμένης  
τετράδας και εισαγωγή στη λίστα  
μη συμπληρωμένων τετράδων για  
την αληθή αποτίμηση της relop

Δημιουργία μη συμπληρωμένης  
τετράδας και εισαγωγή στη λίστα  
μη συμπληρωμένων τετράδων για  
τη μη αληθή αποτίμηση της relop

## *Κλήση Υποπρογραμμάτων*

---

Κλήση διαδικασίας:

call assign\_v (in a, inout b)

par, a, CV, \_

par, b, REF, \_

call, assign\_v , \_ , \_

---

## *Κλήση Υποπρογραμμάτων*

---

Κλήση συνάρτησης:

error = assign\_v (in a, inout b)

par, a, CV, \_

par, b, REF, \_

w = newTemp()

par, w, RET, \_

call, assign\_v , \_ , \_

---

## *Εντολή return*

---

**S -> return (E) {P1}**

**{P1}:      genquad("retv",E.place,"\_","\_")**

## Εκχώρηση

---

$S \rightarrow id := E \{P1\};$

$\{P1\} : \quad \text{genQuad}(":=", E.\text{place}, \_ , id)$

---

## *Δομή while*

---

**S -> while B do S<sup>1</sup>**

---

## *Δομή while*

---

$S \rightarrow \text{while } \{P1\} \text{ B do } S^1$

$\{P1\}: \quad Bquad := \text{nextquad}()$

---

## *Δομή while*

---

$S \rightarrow \text{while } \{P1\} B \text{ do } \{P2\} S^1$

$\{P1\}$ :       $Bquad := nextquad()$

$\{P2\}$ :       $backpatch(B.true, nextquad())$



## *Δομή while*

---

**S -> while {P1} B do {P2} S<sup>1</sup> {P3}**

**{P1}:      Bquad:=nextquad()**

**{P2}:      backpatch(B.true,nextquad())**

**{P3}:      genquad("jump","\_","\_",Bquad)  
            backpatch(B.false,nextquad())**

---

## Δομή while

---

$S \rightarrow \text{while } \{P1\} B \text{ do } \{P2\} S^1 \{P3\}$

**{P1}:      Bquad:=nextquad()**

**{P2}:      backpatch(B.true,nextquad())**

**{P3}:      genquad("jump","\_","\_",Bquad)  
            backpatch(B.false,nextquad())**

Συμπλήρωση των τετράδων που έχουν μείνει ασυμπλήρωτες και γνωρίζουμε τώρα ότι πρέπει να συμπληρωθούν με την επόμενη τετράδα, το true πάνω στην S και το false έξω από τη δομή

Μετάβαση στην αρχή της συνθήκης ώστε να ξαναγίνει έλεγχος

## *Δομή Repeat...Until*

---

**S -> repeat S<sup>1</sup> until (cond)**

---

## *Δομή Repeat...Until*

---

**S -> repeat {P1} S<sup>1</sup> until (cond)**

**{P1}:      sQuad:=nextquad()**

---

## *Δομή Repeat...Until*

---

**S -> repeat {P1} S<sup>1</sup> until (cond) {P2}**

**{P1}:      sQuad:=nextquad()**

**{P2}:      backpatch(cond.False,sQuad)  
            backpatch(cond.True,nextquad())**

## Δομή Repeat...Until


---

**S -> repeat {P1} S<sup>1</sup> until (cond) {P2}**


**{P1}:      sQuad:=nextquad()**

**{P2}:      backpatch(cond.False,sQuad)  
            backpatch(cond.True,nextquad())**

Οι τετράδες αυτές πρέπει να μεταβούν στην αρχή της συνθήκης για να επανελεγχθεί



Συμπλήρωση των τετράδων που έχουν μείνει ασυμπλήρωτες και και γνωρίζουμε τώρα ότι πρέπει να συμπληρωθούν με την επόμενη τετράδα, δηλαδή έξω από τη δομή



## *Δομή if*

---

$S \rightarrow \text{if } B \text{ then } S^1 \text{ TAIL}$

$\text{TAIL} \rightarrow \text{else } S^2 \mid \text{TAIL} \rightarrow \epsilon$

---

## *Δομή if*

---

$S \rightarrow \text{if } B \text{ then } \{P1\} S^1 \text{ TAIL}$

$\{P1\}: \quad \text{backpatch}(B.\text{true}, \text{nextquad}())$

$\text{TAIL} \rightarrow \text{else } S^2 \mid \text{TAIL} \rightarrow \epsilon$

---



## Δομή if

---

$S \rightarrow \text{if } B \text{ then } \{P1\} S^1 \{P2\} \text{TAIL}$

$\{P1\}$ :      `backpatch(B.true,nextquad())`

$\{P2\}$ :      `ifList=makelist(nextquad())`  
              `genquad("jump","_","_","_")`  
              `backpatch(B.false,nextquad())`

$\text{TAIL} \rightarrow \text{else } S^2 \mid \text{TAIL} \rightarrow \epsilon$

---

## *Δομή if*

---

$S \rightarrow \text{if } B \text{ then } \{P1\} S^1 \{P2\} \text{TAIL } \{P3\}$

$\{P1\}$ :      `backpatch(B.true,nextquad())`

$\{P2\}$ :      `ifList=makelist(nextquad())`  
              `genquad("jump","_","_","_")`  
              `backpatch(B.false,nextquad())`

$\{P3\}$ :      `backpatch(ifList,nextquad())`

$\text{TAIL} \rightarrow \text{else } S^2 \mid \text{TAIL} \rightarrow \epsilon$

---

## Δομή if

---

$S \rightarrow \text{if } B \text{ then } \{P1\} S^1 \{P2\} \text{TAIL } \{P3\}$

$\{P1\}: \quad \text{backpatch}(B.\text{true}, \text{nextquad}())$

$\{P2\}: \quad \text{ifList} = \text{makelist}(\text{nextquad}())$

$\quad \text{genquad}(\text{"jump"}, \text{"_"}, \text{"_"}, \text{"_"}, \text{"_"})$

$\quad \text{backpatch}(B.\text{false}, \text{nextquad}())$

$\{P3\}: \quad \text{backpatch}(\text{ifList}, \text{nextquad}())$

$\text{TAIL} \rightarrow \text{else } S^2 \mid \text{TAIL} \rightarrow \epsilon$

Συμπλήρωση των τετράδων που έχουν μείνει ασυμπλήρωτες και και γνωρίζουμε τώρα ότι πρέπει να συμπληρωθούν με την επόμενη τετράδα, στο if και else αντίστοιχα

Εξασφαλίζουμε ότι εάν εκτελεστούν οι εντολές του if δε θα εκτελεστούν στη συνέχεια οι εντολές του else

## *Δομή switch*

---

S -> switch

( (cond): S<sup>1</sup> break )\*

default: S<sup>2</sup>

---

## *Δομή switch*

---

S -> switch {P1}

( (cond): S<sup>1</sup> break )\*

default: S<sup>2</sup>

{P1} : exitlist = emptylist()

---

## *Δομή switch*

---

**S -> switch {P1}**

**( (cond): {P2} S<sup>1</sup> break )\***

**default: S<sup>2</sup>**

**{P1} : exitlist = emptylist()**

**{P2} : backpatch(cond.true,nextquad())**

## *Δομή switch*

---

S -> switch {P1}

( (cond): {P2} S<sup>1</sup> break {P3} )\*

default: S<sup>2</sup>

{P1}: exitlist = emptylist()

{P2}: backpatch(cond.true,nextquad())

{P3}: e = makelist(nextquad())

genquad('jump', '\_', '\_', '\_')

mergelist(exitlist,e)

backpatch(cond.false,nextquad())

---

## *Δομή switch*

---

S -> switch {P1}

( (cond): {P2} S<sup>1</sup> break {P3} )\*

default: S<sup>2</sup> {P4}

{P1} : exitlist = emptylist()

{P2} : backpatch(cond.true,nextquad())

{P3} : e = makelist(nextquad())

genquad('jump', '\_', '\_', '\_')

mergelist(exitlist,e)

backpatch(cond.false,nextquad())

{P4} : backpatch(exitlist,nextquad())

---



## *Είσοδος - Έξοδος*

---

**S -> input (id) {P1}**

**{P1}:      genquad("inp",id.place,"\_","\_")**

**S -> print (E) {P2}**

**{P2}:      genquad("out",E.place,"\_","\_")**

---

---

*Ευχαριστώ*

---