

FORT200

Γενική Περιγραφή

Η γλώσσα FORT200 μοιάζει με τη γλώσσα υψηλού επιπέδου FORTRAN, και ορίζεται στη συνέχεια. Η FORT200 είναι δομημένη με σύνθετες εντολές, σε αντίθεση με την κλασική FORTRAN, η οποία δε διαθέτει τέτοιες εντολές. Εκτός από τους συνήθεις τύπους της FORTRAN, η FORT200 υποστηρίζει και δομές εγγραφών, όπως η Pascal ή η C, καθώς και έναν τύπο λίστας που θυμίζει την αντίστοιχη δομή της LISP. Όμως δεν υποστηρίζει κοινές περιοχές, ούτε ισοδυναμίες μεταβλητών. Η FORT200 δεν έχει αυστηρότητα στη μορφή των προγραμμάτων όπως η κλασική FORTRAN, και το κενό παίζει τον ίδιο ρόλο στη μορφή όπως στις σύγχρονες γλώσσες προγραμματισμού. Τέλος, η FORT200 επιτρέπει τον ορισμό υποπρογραμμάτων σε ένα μόνο – το εξωτερικό – επίπεδο, όπως και η FORTRAN, αλλά χρησιμοποιεί τη στοιβά για κλήση υποπρογραμμάτων, επιτρέποντας έτσι αναδρομή.

Ειδική Περιγραφή

A. Λεκτικές Μονάδες

Οι λεκτικές μονάδες που αποτελούν και τα τερματικά σύμβολα της γραμματικής της FORT200 περιγράφονται στη συνέχεια. Σε παρένθεση – όπου χρειάζεται – δίνονται τα αντίστοιχα συμβολικά ονόματα που εμφανίζονται στη γραμματική της FORT200.

Μαζί με τις λεκτικές μονάδες δίνεται και η περιγραφή των σχολίων της FORT200, τα οποία όμως δεν εμφανίζονται στη γραμματική της γλώσσας.

Σημειώνεται ότι στη γλώσσα FORT200 δεν υπάρχει διάκριση μεταξύ κεφαλαίων και πεζών αλφαβητικών χαρακτήρων, εκτός αν αυτοί αποτελούν μέρος της λέξης μιας λεκτικής μονάδας CCONST ή STRING.

Λέξεις-κλειδιά

Οι παρακάτω λέξεις που αποτελούν ανεξάρτητες λεκτικές μονάδες της FORT200:

FUNCTION SUBROUTINE END INTEGER REAL LOGICAL CHARACTER COMPLEX RECORD
ENDREC LIST DATA CONTINUE GOTO CALL READ WRITE NEW LENGTH IF THEN ELSE
ENDIF DO ENDDO STOP RETURN

Αναγνωριστικά (ID)

Συμβολοσειρές που αρχίζουν με προαιρετικό χαρακτήρα ‘_’, ακολουθούμενο από αλφαβητικό χαρακτήρα, ακολουθούμενο από μηδέν ή περισσότερους αλφαριθμητικούς χαρακτήρες, και δεν αποτελούν λέξεις-κλειδιά. Αν το αναγνωριστικό αρχίζει με ‘_’, τότε μπορεί να περιέχει και άλλους χαρακτήρες ‘_’ μετά τον υποχρεωτικό αλφαβητικό χαρακτήρα, οι οποίοι όμως θα πρέπει να μην είναι διαδοχικοί, ενώ θα πρέπει και να τελειώνει σε ‘_’.

Αποδεκτά παραδείγματα:

a100version2

_a100_version2_

Μη αποδεκτά παραδείγματα¹:

100version2

a100_version2

_a100__version2_

a100_version2_

a100--version2

¹ Σε όλα τα μη αποδεκτά παραδείγματα που δίνονται, η συμβολοσειρά δεν αναγνωρίζεται συνολικά, είναι όμως δυνατό να αναγνωρίζονται μέρη αυτής ως ανεξάρτητες λεκτικές μονάδες.

Απλές σταθερές

Μη προσημασμένες ακέραιες (ICONST):

Ο μοναδικός χαρακτήρας '0', που παριστάνει τη σταθερά με τιμή 0. Επίσης, ένας ή περισσότεροι αριθμητικοί χαρακτήρες, που ο πρώτος δεν είναι ο '0', οπότε η τιμή που παριστάνεται είναι ο αντίστοιχος αριθμός σε δεκαδική βάση. Επίσης, η συμβολοσειρά "0H" ακολουθούμενη από έναν ή περισσότερους αριθμητικούς χαρακτήρες που ο πρώτος δεν είναι ο '0', ή από έναν ή περισσότερους από τους αλφαβητικούς χαρακτήρες 'A', 'B', 'C', 'D', 'E' και 'F'. Στην περίπτωση αυτή, η τιμή που παριστάνεται είναι ο αντίστοιχος αριθμός – μετά το πρόθεμα "0H" – σε δεκαεξαδική βάση. Τέλος, η συμβολοσειρά "0B" ακολουθούμενη από έναν ή περισσότερους από τους αριθμητικούς χαρακτήρες '0' και '1' που ο πρώτος δεν είναι ο '0', οπότε η τιμή που παριστάνεται είναι ο αντίστοιχος αριθμός – μετά το πρόθεμα "0B" – σε δυαδική βάση.

Αποδεκτά παραδείγματα:

0
180
0H9F0
0B1001

Μη αποδεκτά παραδείγματα:

0180
HB7
0H0
00B10
0HG8A
0B301

Μη προσημασμένες πραγματικές (RCONST):

Μηδέν ή περισσότεροι αριθμητικοί χαρακτήρες που ακολουθούνται από το χαρακτήρα '.' και τουλάχιστον έναν αριθμητικό χαρακτήρα, ή ένας τουλάχιστον αριθμητικός χαρακτήρας που ακολουθείται προαιρετικά από το χαρακτήρα '.' και μηδέν ή περισσότερους αριθμητικούς χαρακτήρες. Ακολουθεί προαιρετικά πεδίο εκθέτη, που ξεκινάει με τον χαρακτήρα 'E', ακολουθούμενο από προαιρετικό πρόσημο και τουλάχιστον έναν αριθμητικό χαρακτήρα. Αν δεν υπάρχει πεδίο εκθέτη, ο αριθμός μπορεί να είναι σε δεκαεξαδική ή σε δυαδική βάση, με αντίστοιχο πρόθεμα "0H" ή "0B". Το ακέραιο μέρος της σταθεράς, όπως και το αριθμητικό μέρος του εκθέτη, δε μπορεί να ξεκινά με '0' αν δεν είναι "0". Σε κάθε περίπτωση που υπάρχει κλασματικό μέρος, αυτό πρέπει να περιλαμβάνει τουλάχιστον ένα χαρακτήρα διαφορετικό από τον '0', εκτός αν είναι "0". Τέλος, εξαιρούνται σταθερές που έχουν ήδη οριστεί ως ακέραιες.

Αποδεκτά παραδείγματα:

180E-2
.5
180.100
7.0
0HA.
0B1.0010
0H0.00B9CF

Μη αποδεκτά παραδείγματα:

180E-2.2
.E-2
.5.
7.00
.5G-2
05.2E-05
0HBE-2
0B01.1
1001

Λογικές (LCONST):

Οι συμβολοσειρές ". TRUE ." και ". FALSE .".

Σημειώστε ότι το σύμβολο ‘.’ στην αρχή και στο τέλος της λέξης αποτελεί μέρος αυτής.

Χαρακτήρες (CCONST):

Οποιοσδήποτε εκτυπώσιμος ASCII χαρακτήρας (κωδικοί 32-126) μεταξύ δύο εμφανίσεων του ειδικού χαρακτήρα ‘\’. Επιπλέον, ειδικοί χαρακτήρες ASCII παριστάνονται με τη βοήθεια του χαρακτήρα ‘\’. Πιο συγκεκριμένα, ο χαρακτήρας LF (Line Feed) παριστάνεται ως ‘\n’, ο χαρακτήρας FF (Form Feed) ως ‘\f’, ο χαρακτήρας HT (Horizontal Tab) ως ‘\t’, ο χαρακτήρας CR (Carriage Return) ως ‘\r’, ο χαρακτήρας BS (BackSpace) ως ‘\b’ και ο χαρακτήρας VT (Vertical Tab) ως ‘\v’.

Αποδεκτά παραδείγματα:

```
'a'
'$'
','
'''
'\n'
'\'
```

Μη αποδεκτά παραδείγματα:

```
'ac'
'\p'
'\\'
```

Τελεστές

Λογικό Η (OROP): .OR.

Λογικό ΚΑΙ (ANDOP): .AND.

Λογικό ΟΧΙ (NOTOP): .NOT.

Τελεστές σύγκρισης (RELOP): .GT. .GE. .LT. .LE. .EQ. .NE.

Προσθετικοί τελεστές (ADDOP): + -

Πολλαπλασιασμός (MULOP): *

Διαίρεση (DIVOP): /

Δύναμη (POWEROP): **

Σημειώστε ότι στις τέσσερις πρώτες κατηγορίες το σύμβολο ‘.’ στην αρχή και στο τέλος της λέξης αποτελεί μέρος αυτής.

Στοιχεία λίστας (LISTFUNC)

Ένα ‘A’ και οσαδήποτε ‘D’, ή κανένα ‘A’ και τουλάχιστον ένα ‘D’, μεταξύ ‘C’ και ‘R’.

Αποδεκτά παραδείγματα:

```
CAR
CDR
CADDR
```

Μη αποδεκτά παραδείγματα:

```
CARD
CDAR
```

Ορμαθοί χαρακτήρων (STRING)

Οποιαδήποτε συμβολοσειρά μεταξύ δύο εμφανίσεων του ειδικού χαρακτήρα ‘”’, συμπεριλαμβανομένης της κενής συμβολοσειράς. Ο χαρακτήρας ‘”’ και οι πιο πάνω ειδικοί χαρακτήρες ASCII παριστάνονται σε έναν ορμαθό χαρακτήρων με τη βοήθεια του χαρακτήρα ‘\’. Με κάθε άλλη χρήση του χαρακτήρα ‘\’ παριστάνεται ο χαρακτήρας που ακολουθεί. Έτσι, ο χαρακτήρας ‘\’ καθαυτός παριστάνεται ως \\. Ειδικά όταν ο χαρακτήρας ‘\’ βρίσκεται στο τέλος της γραμμής, ο ορμαθός συνεχίζεται στην επόμενη γραμμή, χωρίς οι χαρακτήρες ‘\’ και αλλαγής γραμμής να αποτελούν μέρος αυτού.

Αποδεκτά παραδείγματα:

```
"CHARACTER +"
""
"STRINGS START AND END WITH \""
"CHARACTER \\ AT THE END OF THE LINE \
EXTENDS STRING IN THE NEXT LINE\n"
```

Άλλες λεκτικές μονάδες

Άλλοι χαρακτήρες που αποτελούν ανεξάρτητες λεκτικές μονάδες είναι:

‘(’ (LPAREN), ‘)’ (RPAREN), ‘,’ (COMMA), ‘=’ (ASSIGN), ‘.’ (DOT), ‘:’ (COLON), ‘[’ (LBRACK), ‘]’ (RBRACK), (EOF)

Σε ορισμένες περιπτώσεις κάποιες από τις παραπάνω λεκτικές μονάδες ενεργούν ως τελεστές, όπως αυτό θα φανεί στη γραμματική και σημασιολογία της FORT200.

Η λεκτική μονάδα EOF – που ως χαρακτήρας ορίζεται με ειδικό τρόπο από το κάθε σύστημα – δεν εμφανίζεται στη γραμματική της FORT200, αλλά πρέπει να παράγεται από το λεκτικό αναλυτή με τιμή 0 για τον τερματισμό της συντακτικής ανάλυσης.

Σχόλια

Τα σχόλια στην FORT200 είναι συμβολοσειρές που ξεκινάνε με το χαρακτήρα ‘\$’ με εύρος μέχρι το τέλος της τρέχουσας γραμμής.

B. Συντακτικοί Κανόνες

Η γραμματική της FORT200 περιγράφεται από τους πιο κάτω κανόνες:

program → body END subprograms

body → declarations statements

declarations → declarations type vars
 | declarations RECORD fields ENDREC vars
 | declarations DATA vals
 | ε

type → INTEGER | REAL | LOGICAL | CHARACTER | COMPLEX

vars → vars COMMA undef_variable
 | undef_variable

undef_variable → LIST undef_variable
 | ID LPAREN dims RPAREN
 | ID

dims → dims COMMA dim
 | dim

dim → ICONST | ID

fields → fields field
 | field

field → type vars
 | RECORD fields ENDREC vars

vals → vals COMMA ID value_list
 | ID value_list

```

value_list → DIVOP values DIVOP

values → values COMMA value
        | value

value → sign constant
       | STRING

sign → ADDOP | ε

constant → simple_constant
          | complex_constant

simple_constant → ICONST | RCONST | LCONST | CCONST

complex_constant → LPAREN RCONST COLON sign RCONST RPAREN

statements → statements labeled_statement
            | labeled_statement

labeled_statement → label statement
                  | statement

label → ICONST

statement → simple_statement
           | compound_statement

simple_statement → assignment
                 | goto_statement
                 | if_statement
                 | subroutine_call
                 | io_statement
                 | CONTINUE
                 | RETURN
                 | STOP

assignment → variable ASSIGN expression
            | variable ASSIGN STRING

variable → variable DOT ID
          | variable LPAREN expressions RPAREN
          | LISTFUNC LPAREN expression RPAREN
          | ID

expressions → expressions COMMA expression
            | expression

expression → expression OROP expression
            | expression ANDOP expression
            | expression RELOP expression
            | expression ADDOP expression
            | expression MULOP expression
            | expression DIVOP expression
            | expression POWEROP expression
            | NOTOP expression
            | ADDOP expression
            | variable
            | simple_constant

```

```

    | LENGTH LPAREN expression RPAREN
    | NEW LPAREN expression RPAREN
    | LPAREN expression RPAREN
    | LPAREN expression COLON expression RPAREN
    | listexpression

listexpression → LBRACK expressions RBRACK
    | LBRACK RBRACK

goto_statement → GOTO label
    | GOTO ID COMMA LPAREN labels RPAREN

labels → labels COMMA label
    | label

if_statement → IF LPAREN expression RPAREN label COMMA label COMMA label
    | IF LPAREN expression RPAREN simple_statement

subroutine_call → CALL variable

io_statement → READ read_list
    | WRITE write_list

read_list → read_list COMMA read_item
    | read_item

read_item → variable
    | LPAREN read_list COMMA ID ASSIGN iter_space RPAREN

iter_space → expression COMMA expression step

step → COMMA expression
    | ε

write_list → write_list COMMA write_item
    | write_item

write_item → expression
    | LPAREN write_list COMMA ID ASSIGN iter_space RPAREN
    | STRING

compound_statement → branch_statement
    | loop_statement

branch_statement → IF LPAREN expression RPAREN THEN body tail

tail → ELSE body ENDIF
    | ENDIF

loop_statement → DO ID ASSIGN iter_space body ENDDO

subprograms → subprograms subprogram
    | ε

subprogram → header body END

header → type FUNCTION ID LPAREN formal_parameters RPAREN
    | LIST FUNCTION ID LPAREN formal_parameters RPAREN
    | SUBROUTINE ID LPAREN formal_parameters RPAREN
    | SUBROUTINE ID

```

```
formal_parameters → type vars COMMA formal_parameters
                    | type vars
```

όπου το σύμβολο ‘|’ διαχωρίζει τα εναλλακτικά δεξιά μέλη των κανόνων και ε είναι η κενή συμβολοσειρά.

Οι παραπάνω κανόνες ορίζουν διφορούμενη γραμματική, που με κατάλληλους μετασχηματισμούς ή βοηθητικές περιγραφές προτεραιότητας και προσεταιριστικότητας τελεστών μπορεί να γίνει μη διφορούμενη.

Αρχικό σύμβολο της γραμματικής της FORT200 αποτελεί το “program”.

Γ. Σημασιολογία

Η σημασιολογία της FORT200 καθορίζεται από μια σειρά κανόνων που αφορούν τη δομή ενός *ορθού* προγράμματος. Κάποιοι από τους κανόνες αυτούς είναι πιθανό να καλύπτονται από τη σύνταξη της γλώσσας, ενώ κάποιοι άλλοι μπορούν να χρησιμοποιηθούν για τη μετατροπή της γραμματικής της FORT200 σε μη διφορούμενη. Όλοι οι υπόλοιποι κανόνες κωδικοποιούνται στο μεταγλωττιστή της FORT200 με τη βοήθεια σημασιολογικών ρουτινών που εκτελούνται κατά τη μετατροπή ενός αρχικού προγράμματος σε ενδιάμεσο κώδικα.

Τύποι δεδομένων

Η FORT200 υποστηρίζει τέσσερις βασικούς τύπους δεδομένων:

- `integer` : ο αριθμητικός τύπος των ακέραιων αριθμών,
- `real` : ο αριθμητικός τύπος των πραγματικών αριθμών,
- `logical` : ο τύπος των λογικών τιμών «Αληθής» και «Ψευδής», και
- `character` : ο τύπος των ASCII χαρακτήρων.

Το μέγεθος και η αναπαράσταση των δεδομένων καθενός από τους βασικούς αριθμητικούς τύπους της FORT200 καθορίζονται από την τελική γλώσσα μεταγλώττισης. Αυτή επιτρέπει μεγέθη τύπων και αναπαράστασεις ανάλογα με την υποστήριξη που δίνει η αρχιτεκτονική για τον καθένα. Η ευθυγράμμιση που απαιτείται για την προσπέλαση δεδομένων καθενός από τους τύπους αυτούς καθορίζεται επίσης από την τελική γλώσσα.

Όσο αφορά τον τύπο `logical`, το μέγεθος των δεδομένων του είναι το ελάχιστο που επιτρέπει η τελική γλώσσα, συνήθως ένα byte, ενώ η αναπαράστασή τους γίνεται με τη σταθερά 0 για την τιμή «Ψευδής» και τη σταθερά 1 για την τιμή «Αληθής».

Παρόμοια, το μέγεθος των δεδομένων του τύπου `character` είναι επίσης το ελάχιστο που επιτρέπει η τελική γλώσσα. Η αναπαράστασή τους γίνεται με τον κώδικα ASCII.

Εκτός από τους πιο πάνω βασικούς τύπους, η FORT200 υποστηρίζει και τον τύπο `complex`, καθώς και τρεις σύνθετους τύπους, τον τύπο πίνακα, τον τύπο εγγραφής και τον τύπο λίστας.

Ο τύπος `complex` είναι ο αριθμητικός τύπος των μιγαδικών αριθμών. Συμμετέχει στη γραμματική της FORT200 ως βασικός, γι’ αυτό και στη συνέχεια θα συμπεριλαμβάνεται σε αυτούς. Σε αντίθεση όμως με τους πιο πάνω τύπους, ο τύπος `complex` παράγεται από τον τύπο `real`, καθώς οι τιμές του προκύπτουν από δύο τιμές αριθμών τύπου `real`, που αντιστοιχούν στο πραγματικό και στο φανταστικό μέρος ενός μιγαδικού αριθμού. Μια τιμή τύπου `complex` μπορεί να παρασταθεί με παράθεση των δύο μερών του, διαχωρισμένων με το σύμβολο ‘:’, και τον παραγόμενο μιγαδικό αριθμό να τοποθετείται σε παρενθέσεις. Τα δεδομένα τύπου `complex` έχουν μέγεθος δύο δεδομένων τύπου `real`, και αποθηκεύονται στη μνήμη συνεχόμενα, ανάλογα με την ευθυγράμμιση του τύπου `real`.

Ο τύπος πίνακα είναι σύνθετος τύπος με στοιχεία δεδομένα του ίδιου βασικού τύπου ή τύπου εγγραφής. Ένα στοιχείο πίνακα αναπαριστάται με το όνομα του πίνακα και μέσα σε παρενθέσεις μια λίστα από θετικούς ακέραιους δείκτες, που αντιστοιχούν στις διαστάσεις του πίνακα. Δείκτες διαδοχικών διαστάσεων διαχωρίζονται με το σύμβολο ‘,’.

Παράδειγμα αναφοράς σε στοιχείο πίνακα είναι το εξής:

$C(103, 5)$

Η δεικτοδότηση ενός πίνακα N στοιχείων σε οποιαδήποτε διάσταση γίνεται με ακέραιες τιμές δείκτη από 1 έως N . Έτσι, οι παρακάτω αναφορές σε στοιχεία πίνακα δεν είναι αποδεκτές:

$C(0, 5, 8)$

$C(3.5)$

$C(-3, 1, 10)$

Δεικτοδότηση με τη βοήθεια μεταβλητής επιτρέπεται μόνο αν ο τύπος του δείκτη είναι ακέραιος. Έτσι, η αναφορά:

$C(i, k+2)$

είναι αποδεκτή, μόνο αν τα αναγνωριστικά i και k είναι ακέραιου τύπου.

Ο τύπος εγγραφής είναι ο δεύτερος σύνθετος τύπος της FORT200, με στοιχεία πιθανά διαφορετικών τύπων που ονομάζουμε *πεδία* της εγγραφής. Ένα πεδίο εγγραφής αναπαριστάται με το όνομα της εγγραφής και το όνομα του πεδίου, διαχωρισμένα με το ειδικό σύμβολο ‘.’.

Παράδειγμα αναφοράς σε στοιχείο εγγραφής είναι το εξής:

$r.a1$

όπου r είναι το όνομα της εγγραφής και $a1$ είναι το όνομα κάποιου πεδίου της.

Όσο αφορά τον τρίτο σύνθετο τύπο της FORT200, τον τύπο λίστας, τα στοιχεία του μπορεί να είναι βασικού ή σύνθετου τύπου, και αναφορά σε αυτά γίνεται με τη βοήθεια ειδικών συναρτήσεων στοιχείων λίστας. Οι συναρτήσεις αυτές ορίζονται παρακάτω. Μια ολόκληρη λίστα στοιχείων βασικού τύπου μπορεί να παρασταθεί και ως έκφραση λίστας με τη βοήθεια αγκυλών. Για παράδειγμα, μια λίστα ακεραίων μπορεί να είναι η:

$[1,3,-5,0,2]$

Η FORT200 υποστηρίζει φωλιασμένες δηλώσεις σύνθετων τύπων, μέχρι ένα μέγιστο βάθος φωλιάσματος ίσο με 5, με τον περιορισμό ότι δεν επιτρέπεται άμεσο φώλιασμα πινάκων, αφού όπως αναφέρθηκε παραπάνω, ένα στοιχείο πίνακα δε μπορεί να είναι άλλου τύπου πίνακα. Έτσι, η αναφορά:

$r(10).a1.b(2,3)$

είναι αποδεκτή όταν r είναι πίνακας εγγραφών, $a1$ είναι πεδίο του r τύπου εγγραφής, και b είναι πεδίο του $a1$ τύπου πίνακα. Λίστες πινάκων δεν ορίζονται, ορίζονται όμως πίνακες λιστών.

Η αποθήκευση των στοιχείων ενός πίνακα της FORT200 στη μνήμη γίνεται σε συνεχόμενες θέσεις, ανάλογα με την ευθυγράμμιση του τύπου αυτών, με αύξουσα σειρά μεταβολής των δεικτών από την πρώτη διάσταση προς την τελευταία. Για παράδειγμα, ένας τρισδιάστατος πίνακας A $3 \times 3 \times 2$ στοιχείων αποθηκεύει τα στοιχεία του με τη σειρά: $A(1,1,1)$, $A(2,1,1)$, $A(3,1,1)$, $A(1,2,1)$, $A(2,2,1)$, $A(3,2,1)$, $A(1,3,1)$, $A(2,3,1)$, $A(3,3,1)$, $A(1,1,2)$, $A(2,1,2)$, $A(3,1,2)$, $A(1,2,2)$, $A(2,2,2)$, $A(3,2,2)$, $A(1,3,2)$, $A(2,3,2)$ και $A(3,3,2)$. Με τον τρόπο αυτό, ένας δισδιάστατος πίνακας αποθηκεύεται στήλη-στήλη.

Η αποθήκευση των πεδίων μιας εγγραφής γίνεται επίσης σε συνεχόμενες θέσεις μνήμης, ανάλογα με την ευθυγράμμιση του τύπου του κάθε πεδίου, και με τη σειρά που τα πεδία έχουν δηλωθεί.

Ένας τύπος πίνακα καταλαμβάνει τόσες θέσεις αποθήκευσης, όσος είναι ο αριθμός των στοιχείων του επί το μέγεθος του ευθυγραμμισμένου στοιχείου του. Αντίστοιχα, ένας τύπος εγγραφής καταλαμβάνει τόσες θέσεις αποθήκευσης, όσο είναι το άθροισμα των θέσεων των ευθυγραμμισμένων πεδίων του.

Η αποθήκευση μιας λίστας της FORT200 στη μνήμη γίνεται με δυναμικό τρόπο. Ένα στοιχείο λίστας αποθηκεύεται ως ένα διατεταγμένο ζεύγος (περιεχόμενο, διεύθυνση επόμενου στοιχείου). Μια μεταβλητή τύπου λίστας έχει μέγεθος όσο το μέγεθος μιας διεύθυνσης μνήμης και τιμή τη διεύθυνση του πρώτου στοιχείου της λίστας. Το τελευταίο στοιχείο μιας λίστας έχει τιμή 0 ως διεύθυνση επόμενου στοιχείου. Μια έκφραση λίστας δημιουργεί μια ακολουθία τέτοιων ζευγών. Οι διευθύνσεις στοιχείων παράγονται από κάποιο μηχανισμό δυναμικής εκχώρησης μνήμης, και ως εκ τούτου διαδοχικά στοιχεία δεν αποθηκεύονται κατ' ανάγκη στη σειρά. Κάθε στοιχείο λίστας καταλαμβάνει τόσες θέσεις μνήμης, όσο είναι το άθροισμα των θέσεων του ευθυγραμμισμένου περιεχομένου του και της ευθυγραμμισμένης διεύθυνσης επόμενου στοιχείου.

Δομή ενός προγράμματος FORT200

Ένα πρόγραμμα σε γλώσσα FORT200 αποτελείται από την κύρια μονάδα και μονάδες υποπρογραμμάτων. Κάθε υποπρόγραμμα έχει την ίδια δομή με την κύρια μονάδα, και μια επικεφαλίδα που καθορίζει τις παραμέτρους του.

Η FORT200 – όπως η κλασική FORTRAN – δε διαθέτει καθολικές μεταβλητές, σε αντίθεση με τις περισσότερες σύγχρονες γλώσσες προγραμματισμού. Έτσι η επικοινωνία μεταξύ των μονάδων του προγράμματος γίνεται μέσω των παραμέτρων των υποπρογραμμάτων του.

Κάθε μονάδα περιέχει:

- Δηλώσεις μεταβλητών (προαιρετικά): Οι μεταβλητές αυτές έχουν εμβέλεια τη μονάδα. Εσωτερικές εμβέλειες ορίζονται με κάθε δομημένη εντολή της FORT200, κι επομένως μια μονάδα μπορεί να έχει φωλιασμένες εμβέλειες.
- Αποδόσεις αρχικών τιμών (προαιρετικά): Οι μεταβλητές που δηλώθηκαν νωρίτερα μπορούν να αρχικοποιούνται στην αρχή της εκτέλεσης του προγράμματος.
- Εντολές: Τουλάχιστον μια εντολή πρέπει να υπάρχει σε κάθε μονάδα, όπως προκύπτει κι από τους συντακτικούς κανόνες της γλώσσας. Οι εντολές χωρίζονται σε απλές και δομημένες, με τις τελευταίες να μπορούν να περιέχουν δηλώσεις μεταβλητών και τουλάχιστον μια άλλη απλή ή δομημένη εντολή.
- Ετικέτες: Μιας εντολής μπορεί να προηγείται μια ετικέτα, που είναι μια ακέραια σταθερά με ορατότητα την τρέχουσα εμβέλεια. Η ετικέτα αναπαριστά τη διεύθυνση της εντολής στον κώδικα του προγράμματος και μπορεί να χρησιμοποιηθεί σε συνδυασμό με κάποιες εντολές της FORT200.
- Δήλωση τερματισμού μονάδας end.

Οι δηλώσεις των αναγνωριστικών της FORT200 πρέπει να είναι μοναδικές σε κάθε εμβέλεια, είτε αυτά αντιστοιχούν σε υποπρογράμματα, είτε σε μεταβλητές. Στην εξωτερική εμβέλεια του προγράμματος ανήκουν τα ονόματα των υποπρογραμμάτων, ενώ στις εσωτερικές εμβέλειες ανήκουν τα ονόματα των μεταβλητών. Η επίλυση αναφοράς σε μη τοπικά ονόματα γίνεται με στατικό δέσιμο.

Η εκτέλεση του κώδικα μιας μονάδας ξεκινάει με την πρώτη εντολή της μονάδας, ενώ τερματίζεται με ειδικές εντολές της FORT200.

Δηλώσεις μεταβλητών

Οι μεταβλητές δηλώνονται στην αρχή των εμβελειών μιας μονάδας. Κάθε μεταβλητή έχει ορατότητα την εμβέλεια στην οποία δηλώνεται, εκτός από όσες φωλιασμένες εμβέλειες την επισκιάζουν.

Μια δήλωση μεταβλητών – πλην εγγραφών – αποτελείται από:

- (α) το όνομα ενός βασικού τύπου,
- (β) ένα ή περισσότερα αναγνωριστικά που αποδίδονται σε μεταβλητές αυτού του τύπου,

- (γ) πριν από ένα αναγνωριστικό την προαιρετική λέξη-κλειδί list που μετατρέπει τη δήλωση γι' αυτό το αναγνωριστικό σε δήλωση λίστας, και
- (δ) μετά από ένα αναγνωριστικό δηλώσεις διαστάσεων για μετατροπή της δήλωσης αυτού του αναγνωριστικού σε δήλωση πίνακα.

Μια δήλωση εγγραφής αποτελείται από:

- (α) τη λέξη-κλειδί record,
- (β) δηλώσεις πεδίων,
- (γ) τη λέξη-κλειδί endrec,
- (δ) ένα ή περισσότερα αναγνωριστικά που αποδίδονται σε μεταβλητές αυτού του τύπου εγγραφής, και
- (ε) όπως παραπάνω, για κάθε αναγνωριστικό δυνατότητα μετατροπής σε λίστα ή/και πίνακα. Οι δηλώσεις πεδίων γίνονται όπως οι δηλώσεις μεταβλητών.

Παραδείγματα δηλώσεων είναι τα παρακάτω:

```
integer i, j, k logical l real dev complex c(100,10)
real a, list list b, z(5), p(100,5), q
record integer i, j, k real list x(10) endrec r1, r2(5)
record real a record integer m, n endrec list b endrec r3
```

Ειδικότερα:

1. Κάθε δήλωση γίνεται για ένα μόνο βασικό τύπο ή τύπο εγγραφής. Περισσότερες δηλώσεις για τον ίδιο τύπο μπορούν να ακολουθούν παρακάτω.
2. Για έναν τύπο πίνακα, η δήλωσή του γίνεται με βάση τον τύπο των στοιχείων του. Ο αριθμός και το μέγεθος των διαστάσεων του πίνακα καθορίζονται σε παρένθεση δίπλα στο όνομα του πίνακα. Έτσι, με:


```
c(100,10)
```

 δηλώνεται ένας δισδιάστατος πίνακας 100 γραμμών και 10 στηλών.
3. Η λέξη-κλειδί list για μετατροπή σε λίστα μπορεί να επαναλαμβάνεται, ώστε να επιτρέπεται η δημιουργία φωλιασμένων λιστών.
4. Μετά από μετατροπή σε λίστα μπορεί να ακολουθήσει επέκταση σε πίνακα. Το αποτέλεσμα είναι ένας πίνακας λιστών, και όχι το ανάποδο.
5. Σε κάθε δήλωση μπορούν να συνυπάρχουν αναγνωριστικά βαθμωτών μεταβλητών, μεταβλητών εγγραφής, μεταβλητών τύπου πίνακα και μεταβλητών λίστας.

Όλες οι μεταβλητές της FORT200 – εκτός από τις τυπικές παραμέτρους ενός υποπρογράμματος – είναι στατικές. Δεν τοποθετούνται στη στοίβα, αλλά έχουν σταθερές διευθύνσεις έξω από αυτή, έχουν δε διάρκεια ζωής όλη τη διάρκεια εκτέλεσης του προγράμματος.

Αποδόσεις αρχικών τιμών

Οι μεταβλητές μιας μονάδας μπορούν να αρχικοποιηθούν με ειδική δήλωση στην αρχή της μονάδας.

Μια δήλωση απόδοσης αρχικών τιμών αποτελείται από:

- (α) τη λέξη-κλειδί data,
- (β) το όνομα μιας μεταβλητής που θέλουμε να αρχικοποιήσουμε,
- (γ) μια λίστα αρχικών τιμών που αποδίδονται στη μεταβλητή, και
- (δ) αν θέλουμε, αποδόσεις αρχικών τιμών για άλλες μεταβλητές.

Παραδείγματα δηλώσεων απόδοσης αρχικών τιμών είναι τα πιο κάτω:

```
data i/1/, x/.true./, c/(3.2:1.8), -(3.1:0.3), (0.: -3.4) /
data a/8.e-2/, p/-1.0, -1.0, -1.0/
data s/"string", "another string"/, v/'g','o' /
```

Ειδικότερα:

1. Οι δηλώσεις απόδοσης αρχικών τιμών γίνονται με τη βοήθεια του διαχωριστικού συμβόλου '/', που τοποθετείται πριν και μετά από μια λίστα αρχικών τιμών.
2. Όλες οι μεταβλητές που εμφανίζονται σε δήλωση απόδοσης αρχικών τιμών μιας μονάδας πρέπει να έχουν δηλωθεί νωρίτερα στη μονάδα. Ο μεταγλωττιστής πρέπει να ελέγχει, αν οι αρχικές τιμές που αποδίδονται είναι τύπου συμβατού για ανάθεση με τον τύπο της αντίστοιχης μεταβλητής, ή, για τύπο εγγραφής, με τους τύπους των πεδίων της εγγραφής, ή, για τύπο πίνακα, με τον τύπο των στοιχείων του πίνακα. Η συμβατότητα για ανάθεση ορίζεται πιο κάτω.
3. Μια απόδοση αρχικών τιμών σε μεταβλητή τύπου εγγραφής ή πίνακα γίνεται με απόδοση των τιμών της λίστας σε διαδοχικά πεδία της εγγραφής ή στοιχεία του πίνακα, με τη σειρά αποθήκευσης που αναφέρθηκε νωρίτερα. Αν ο αριθμός των αρχικών τιμών που αποδίδονται δεν είναι ο ίδιος με τον αριθμό των πεδίων ή στοιχείων, τότε: αν είναι μικρότερος από το δεύτερο, τα υπόλοιπα πεδία ή στοιχεία θα έχουν μηδενική αρχική τιμή – ανάλογα με τον τύπο τους, διαφορετικά οι υπόλοιπες αρχικές τιμές αγνοούνται. Παρόμοια, αν σε μια βαθμωτή μεταβλητή αποδοθούν περισσότερες από μία τιμή, μόνο η πρώτη αρχικοποιεί τη μεταβλητή, ενώ οι υπόλοιπες αγνοούνται.
4. Για πίνακα στοιχείων τύπου character, η αρχικοποίηση μπορεί να γίνει με ορμαθό χαρακτήρων, που ισοδυναμεί με τόσες τιμές τύπου character, όσο είναι το μήκος του ορμαθού συν ένας χαρακτήρας με τιμή 0 που τερματίζει τον ορμαθό. Ειδικά για πολυδιάστατους πίνακες στοιχείων τύπου character, απόδοση αρχικών τιμών μπορεί να γίνει με διαδοχικούς ορμαθούς χαρακτήρων. Σε τέτοια περίπτωση, η απόδοση ισοδυναμεί με πολλαπλές διαδοχικές αποδόσεις ορμαθών σε μονοδιάστατους πίνακες character, αντί για διαδοχικές αποδόσεις μεμονωμένων τιμών τύπου character.
5. Οι αρχικές τιμές επιτρέπεται να είναι προσημασμένες, αν είναι αριθμητικού τύπου. Η χρήση προσήμου για τιμές τύπου logical ή character αποτελεί σημασιολογικό σφάλμα.
6. Δηλώσεις απόδοσης αρχικών τιμών επιτρέπονται μόνο στην πιο εξωτερική εμβέλεια των μονάδων.
7. Απόδοση αρχικής τιμής σε τύπο λίστας επιτρέπεται μόνο για τιμή 0. Απόδοση μη μηδενικής αρχικής τιμής αποτελεί σημασιολογικό σφάλμα.

Επειδή οι μεταβλητές της FORT200 είναι στατικές, οι αρχικές τιμές που τους αποδίδονται με τον παραπάνω τρόπο αποθηκεύονται κατ' ευθείαν σε κατάλληλο χώρο στον κώδικα που παράγει ο μεταγλωττιστής και δεν αποδίδονται κατά την εκτέλεση του προγράμματος.

Δηλώσεις υποπρογραμμάτων

Τα υποπρογράμματα της FORT200 έχουν καθολική εμβέλεια και δηλώνονται στο εξωτερικό περιβάλλον, αμέσως μετά την κύρια μονάδα του προγράμματος.

Ένα υποπρόγραμμα δε χρειάζεται να έχει δηλωθεί πριν την κλήση του, γεγονός που έχει σαν αποτέλεσμα ο έλεγχος ορθότητας της κλήσης να πρέπει να γίνεται εκ των υστέρων – όταν δηλαδή συναντηθεί η δήλωσή του – με την τεχνική του μπαλώματος (backpatching), δηλαδή με επιστροφή της ανάλυσης στην εντολή κλήσης.

Τα υποπρογράμματα της FORT200 διακρίνονται σε υπορουτίνες (subroutines) και συναρτήσεις (functions). Οι υπορουτίνες καλούνται μέσα από ειδική εντολή της FORT200, ενώ οι συναρτήσεις μέσα από εκφράσεις. Οι συναρτήσεις επιστρέφουν αποτελέσματα μέσα από τα ονόματά τους που για το σκοπό αυτό λειτουργούν και σαν καθολικές μεταβλητές του προγράμματος.

Κάθε υποπρόγραμμα είναι μια μονάδα, που εκτός από τη δομή της μονάδας, διαθέτει και επικεφαλίδα. Αυτή περιέχει με τη σειρά:

- (α) τον τύπο του αποτελέσματος, εάν το υποπρόγραμμα είναι συνάρτηση,
- (β) το είδος του υποπρογράμματος,
- (γ) το όνομά του, και

(δ) τις τυπικές παραμέτρους του.

Παραδείγματα επικεφαλίδων υποπρογραμμάτων είναι τα εξής:

```
real function LL (integer y, list w(10))
list function F (integer n)
subroutine A(integer n, x(n), logical z(100))
subroutine S
```

Ειδικότερα:

1. Η δήλωση των τυπικών παραμέτρων είναι πλήρης, περιλαμβάνει δηλαδή τον τύπο και τις διαστάσεις τους. Έτσι, πριν από μια λίστα παραμέτρων πρέπει να δηλώνεται ο τύπος τους, και εάν μια παράμετρος είναι τύπου πίνακα, η δήλωση των διαστάσεών της είναι υποχρεωτική, ενώ αν είναι τύπου λίστας, η δήλωση μετατροπής σε λίστα είναι επίσης υποχρεωτική.
2. Μία παράμετρος τύπου πίνακα μπορεί να δηλωθεί με εικονικό μέγεθος των διαστάσεών της, με τη βοήθεια ενός ή περισσότερων αναγνωριστικών. Τα αναγνωριστικά αυτά πρέπει να συγκαταλέγονται στις παραμέτρους του υποπρογράμματος, να προηγούνται του ονόματος του πίνακα, και βέβαια να δηλώνονται ως τύπου integer.
3. Το πέρασμα μιας παραμέτρου βασικού τύπου γίνεται κατ' αναφορά, εάν υπάρχει ανάθεση σε αυτήν μέσα στο υποπρόγραμμα, διαφορετικά γίνεται κατ' αξία. Οι παράμετροι σύνθετου τύπου περνούν κατ' αναφορά.
4. Μια υπορουτίνα μπορεί να μην έχει παραμέτρους, μια συνάρτηση όμως πρέπει να έχει τουλάχιστον μία παράμετρο.
5. Ο τύπος του αποτελέσματος μιας συνάρτησης πρέπει να είναι βασικός ή λίστας. Στην τελευταία περίπτωση, ως τύπος αποτελέσματος αναφέρεται μόνο η λέξη-κλειδί list και δεν ορίζεται ο τύπος στοιχείου της λίστας, ενώ τιμή αποτελέσματος θα είναι η διεύθυνση του πρώτου στοιχείου της λίστας.

Η κλασική FORTRAN δεν υποστηρίζει αναδρομή. Έτσι, κάθε υποπρόγραμμα έχει ακριβώς ένα ενεργό αντίγραφο του χώρου δεδομένων του, το οποίο καθορίζεται στατικά από το μεταγλωττιστή. Η FORT200 υποστηρίζει αναδρομή μειωμένων δυνατοτήτων, χρησιμοποιώντας τη στοίβα για τη μετάδοση των παραμέτρων του υποπρογράμματος και για την επιστροφή του αποτελέσματος μιας συνάρτησης. Κατά τα άλλα, ακολουθεί την κλασική FORTRAN για τον υπόλοιπο χώρο δεδομένων, κι επομένως οι μεταβλητές μιας αναδρομικής συνάρτησης είναι κοινές για όλες τις αναδρομικές κλήσεις της συνάρτησης.

Εκφράσεις

Η FORT200 υποστηρίζει αριθμητικές εκφράσεις, λογικές εκφράσεις και εκφράσεις λίστας. Οι τιμές των πρώτων μπορούν να είναι τύπου integer, real ή complex, οι τιμές των δεύτερων μπορούν να είναι μόνο τύπου logical, ενώ οι τιμές των τρίτων είναι σύνθετου τύπου λίστας. Οι εκφράσεις αποτιμώνται με βάση συγκεκριμένους κανόνες και με τη βοήθεια των τελεστών της FORT200. Οι αριθμητικές εκφράσεις μπορούν να περιέχονται σε λογικές εκφράσεις, αλλά όχι το αντίστροφο.

Εκτός από τις πιο πάνω, ορίζονται και απλές εκφράσεις τύπου character, οι οποίες είτε εμφανίζονται μεμονωμένα σε κλήσεις συναρτήσεων ή εντολές της FORT200, είτε περιέχονται σε λογικές εκφράσεις. Τέλος, ορίζονται και απλές εκφράσεις σύνθετου τύπου που εμφανίζονται μόνο σε κλήσεις υποπρογραμμάτων.

Μια έκφραση της FORT200 περιλαμβάνει:

- Τιμές αριστερής προσπέλασης (L-values), δηλαδή διευθύνσεις του χώρου δεδομένων του προγράμματος που αντιστοιχούν σε μεταβλητές, πεδία εγγραφών, στοιχεία πινάκων ή στοιχεία λίστας.
- Σταθερές, δηλαδή τιμές που υπάρχουν στον αρχικό κώδικα της μονάδας.
- Τελεστές, που επιτρέπουν πράξεις μεταξύ υποεκφράσεων.

- Κλήσεις συναρτήσεων, που έχουν σαν αποτέλεσμα την αντικατάσταση της συνάρτησης στην έκφραση με την τιμή που αυτή επιστρέφει.
Τα τρία τελευταία αντικείμενα ορίζουν τιμές δεξιάς προσπέλασης (R-values).

Παραδείγματα εκφράσεων της FORT200 είναι τα εξής:

```
i
a+1
a(3,i+1,j-1)**k**2 + (tu(b,z(2)) / (i-1)) * i
a+b .gt. 0.0 .and. .not. x .eq. 'g'
z+w.a(i)/w.b(i)
[3, x+3, f(4*y)]
```

Τιμές αριστερής προσπέλασης

Κάθε διεύθυνση στο χώρο δεδομένων του προγράμματος ονομάζεται τιμή αριστερής προσπέλασης, επειδή μπορεί να βρεθεί στο αριστερό μέρος μιας ανάθεσης. Τέτοιες τιμές στην FORT200 αποτελούν οι μεταβλητές, τα πεδία εγγραφών, τα στοιχεία πινάκων και τα στοιχεία λίστας.

Όταν μια τιμή αριστερής προσπέλασης βρεθεί σε μια έκφραση, αποτιμάται, και η τιμή που αποδίδεται γι' αυτήν είναι η τιμή που περιέχεται στη διεύθυνση που αυτή παριστάνει, δηλαδή η τιμή της μεταβλητής, του πεδίου εγγραφής ή του στοιχείου πίνακα ή λίστας. Απαραίτητη προϋπόθεση για αποτίμηση είναι η προηγούμενη δήλωση της αντίστοιχης μεταβλητής, της αντίστοιχης εγγραφής, του αντίστοιχου πίνακα ή της αντίστοιχης λίστας, στην ίδια εμβέλεια ή σε κάποια εξωτερική της. Σε περίπτωση απουσίας τέτοιας δήλωσης υπάρχει σημασιολογικό σφάλμα. Μια μεταβλητή τύπου εγγραφής, πίνακα ή λίστας μπορεί να εμφανιστεί σε μια έκφραση χωρίς αναφορά σε πεδίο ή στοιχείο της, μόνο εάν αποτελεί πραγματική παράμετρο στην κλήση ενός υποπρογράμματος.

Η αποτίμηση μεταβλητής που έχει δηλωθεί κανονικά σε μια μονάδα γίνεται με ανάγνωση της τιμής της μεταβλητής από το χώρο δεδομένων.

Η αποτίμηση πεδίου εγγραφής που έχει δηλωθεί κανονικά μπορεί να γίνει, εφ' όσον το αναγνωριστικό του πεδίου στο οποίο γίνεται αναφορά αντιστοιχεί σε δηλωμένο όνομα πεδίου της εγγραφής. Τότε, η διεύθυνση του πεδίου υπολογίζεται με τη διεύθυνση βάσης της εγγραφής και τη μετατόπιση του πεδίου σχετικά με αυτήν. Η μετατόπιση κάθε πεδίου είναι στατικά γνωστή στο μεταγλωττιστή και δεν απαιτεί υπολογισμό κατά την εκτέλεση του προγράμματος. Η αποτίμηση του πεδίου μπορεί να γίνει με ανάγνωσή του από το χώρο δεδομένων.

Η αποτίμηση στοιχείου πίνακα που επίσης έχει δηλωθεί κανονικά μπορεί να γίνει, εφ' όσον οι τιμές των εκφράσεων που αποδίδονται στους δείκτες του στοιχείου αυτού είναι μέσα στα επιτρεπόμενα όρια για τις αντίστοιχες διαστάσεις του πίνακα. Μ' άλλα λόγια, στην αποτίμηση στοιχείου πίνακα προηγείται η αποτίμηση των δεικτών του. Εάν οι τιμές των δεικτών είναι αποδεκτές, υπολογίζεται η διεύθυνση του ζητούμενου στοιχείου, λαμβάνοντας υπ' όψη την αρχική διεύθυνση του πίνακα στο χώρο δεδομένων της μονάδας, τον τρόπο αποθήκευσης των στοιχείων του πίνακα, τις διαστάσεις και το μέγεθός του σε κάθε διάσταση, καθώς και το μέγεθος του στοιχείου του. Στη συνέχεια μπορεί να αποτιμηθεί το στοιχείο αυτό, με ανάγνωσή του από το χώρο δεδομένων.

Για σύνθετους τύπους που συνδυάζουν πίνακες με εγγραφές, η αποτίμηση του τελικού βασικού στοιχείου γίνεται έπειτα από συνδυασμό των παραπάνω ενεργειών.

Η FORT200 έχει ισχυρό σύστημα τύπων, κι επομένως ο πιο πάνω έλεγχος τιμών των δεικτών για στοιχεία πίνακα πρέπει να γίνεται από τον κώδικα του προγράμματος. Ο μεταγλωττιστής, δηλαδή, πρέπει για κάθε προσπέλαση πίνακα να παράγει κώδικα που να κάνει αυτόν τον έλεγχο.

Για μια λίστα που έχει δηλωθεί κανονικά, η αποτίμηση ενός στοιχείου της γίνεται μόνο μέσω των ειδικών συναρτήσεων στοιχείων λίστας, οι οποίες με παράμετρο μια λίστα αναφέρονται στο περιεχόμενο και στη διεύθυνση επόμενου στοιχείου ως εξής:

1. Η συνάρτηση CAR() αναφέρεται στο περιεχόμενο του πρώτου στοιχείου της λίστας.
2. Η συνάρτηση CDR() αναφέρεται στη διεύθυνση του επόμενου στοιχείου της λίστας.
3. Η συνάρτηση CADR() αναφέρεται στο περιεχόμενο του δεύτερου στοιχείου της λίστας.
4. Η συνάρτηση CDDR() αναφέρεται στη διεύθυνση του μεθεπόμενου στοιχείου της λίστας.
5. Όμοια ορίζονται και άλλες συναρτήσεις που ακολουθούν την περιγραφή της λεκτικής μονάδας LISTFUNC που δόθηκε νωρίτερα.

Οι συναρτήσεις αυτές είναι τιμές αριστερής προσπέλασης που μέσα σε εκφράσεις αποτιμώνται και επιστρέφουν τις παραπάνω τιμές.

Οι τυπικές παράμετροι ενός υποπρογράμματος είναι τιμές αριστερής προσπέλασης στο δυναμικό χώρο δεδομένων του υποπρογράμματος στη στοίβα, αν μεταδίδονται κατ' αξία, και στον υπόλοιπο χώρο δεδομένων, αν μεταδίδονται κατ' αναφορά. Στη δεύτερη περίπτωση, και σε κάθε κλήση του υποπρογράμματος, οι αντίστοιχες πραγματικές παράμετροι πρέπει να είναι τιμές αριστερής προσπέλασης, που δεν αποτιμώνται, αλλά περνάνε στο υποπρόγραμμα με τη διεύθυνσή τους. Η αποτίμηση μιας τυπικής παραμέτρου που μεταδίδεται κατ' αναφορά γίνεται σε εκφράσεις που αυτή εμφανίζεται, με ανάγνωση από τη διεύθυνσή της. Ειδικά αν η τυπική παράμετρος είναι σύνθετου τύπου, οπότε και μεταδίδεται κατ' αναφορά, η προσπέλαση κάποιου στοιχείου της γίνεται όπως αναφέρθηκε παραπάνω.

Σταθερές

Οι σταθερές της FORT200 είναι αυτές που αναγνωρίζονται άμεσα από το λεκτικό αναλυτή και περιγράφηκαν στην αντίστοιχη ενότητα, καθώς και σταθερές τύπου complex. Οι τελευταίες προκύπτουν από δύο σταθερές τύπου real, με βάση αντίστοιχο συντακτικό κανόνα της γλώσσας, σε συμφωνία με την προηγούμενη περιγραφή τιμών τύπου complex.

Οι τιμές των αριθμητικών σταθερών προκύπτουν άμεσα με μετατροπή των αντίστοιχων λέξεων σε αριθμητικές τιμές. Οι τιμές των λογικών σταθερών, που συμβολίζονται με τις λέξεις “true.” και “false.” για «Αληθής» και «Ψευδής» αντίστοιχα, αποδίδονται με την κωδικοποίηση που αναφέρθηκε παραπάνω. Οι τιμές των σταθερών χαρακτήρων αποδίδονται με την κωδικοποίηση ASCII.

Οι αριθμητικές σταθερές δεν έχουν πρόσημο, και προσημασμένοι αριθμοί προκύπτουν από εκφράσεις με τη βοήθεια του τελεστή προσήμου ADDOP. Εξάιρεση αποτελεί το φανταστικό μέρος των μιγαδικών σταθερών, όπως φαίνεται και από τη σύνταξη αυτών.

Τελεστές

Τελεστές είναι ειδικά σύμβολα, που εφαρμοζόμενα σε έναν αριθμό εκφράσεων – που ονομάζονται *τελούμενα εισόδου* ή *τελεστέοι*, παράγουν μια νέα έκφραση.

Οι τελεστές της FORT200 δίνονται στο σχετικό πίνακα και διακρίνονται σε τελεστές με ένα τελούμενο και τελεστές με δύο τελούμενα εισόδου. Στην πρώτη κατηγορία ανήκουν οι τελεστές του προσήμου και της λογικής άρνησης. Οι τελεστές αυτής της κατηγορίας αναγράφονται πριν το τελούμενό τους. Στη δεύτερη κατηγορία ανήκουν οι υπόλοιποι τελεστές αριθμητικών και λογικών πράξεων. Οι τελεστές αυτής της κατηγορίας αναγράφονται ανάμεσα στα τελούμενά τους.

Τελεστές μπορούν να θεωρηθούν και τα διαχωριστικά σύμβολα ‘(’, ‘)’, ‘[’, ‘]’, ‘.’ και ‘:’, σε περιπτώσεις που θα αναλυθούν στη συνέχεια.

Πιο συγκεκριμένα:

- Ο τελεστής αναφοράς σε στοιχείο πίνακα / κλήσης συνάρτησης ‘()’ έχει σύνταξη:
<αναγνωριστικό> ‘(’ <λίστα εκφράσεων> ‘)’

| Τελεστής | Περιγραφή | Αριθμός τε- λούμενων | Προσεταιριστικότητα |
|--------------|--|-------------------------|---------------------|
| '()' , '.' | Αναφορά σε στοιχείο πίνακα / Κλήση συνάρτησης, Αναφορά σε πεδίο εγγραφής | 2 | αριστερή |
| POWEROP | Ύψωση σε δύναμη | 2 | δεξιά |
| MULOP, DIVOP | Πολλαπλασιασμός, διαίρεση | 2 | αριστερή |
| ADDOP | Πρόσθημο, πρόσθεση, αφαι- ρεση, ένωση λιστών | 1, 2, 2, 2 | αριστερή |
| RELOP | Σχεσιακοί τελεστές | 2 | καμία |
| NOTOP | Λογική άρνηση | 1 | - |
| ANDOP | Λογικό γινόμενο | 2 | αριστερή |
| OROP | Λογικό άθροισμα | 2 | αριστερή |
| '(:)' | Δημιουργία μιγαδικού αριθμού | 2 | - |
| '[' | Σύνθεση λίστας | 1 | - |

Τελεστές της FORT200 σε φθίνουσα σειρά προτεραιότητας

Ο τελεστής αυτός έχει δύο τελούμενα εισόδου, από τα οποία το πρώτο είναι το όνομα του πίνακα ή της συνάρτησης, και το δεύτερο μια λίστα εκφράσεων, οι τιμές των οποίων αποδίδονται στους δείκτες του στοιχείου πίνακα ή στις πραγματικές παραμέτρους, αντίστοιχα. Η αποτίμηση των εκφράσεων στη λίστα γίνεται από τα αριστερά προς τα δεξιά. Η αναφορά σε στοιχείο πίνακα έχει περιγραφεί νωρίτερα, ενώ η κλήση συνάρτησης θα επεξηγηθεί παρακάτω. Στη FORT200 η σύνταξη της κλήσης συναρτήσεων ταυτίζεται με την σύνταξη της αναφοράς σε στοιχείο πίνακα. Η σημασιολογική ανάλυση διαχωρίζει τη μια λειτουργία από την άλλη.

Ως πίνακας μπορεί να είναι και κάποια έκφραση αριστερής προσπέλασης τύπου complex, οπότε επιτρέπεται μόνο μία έκφραση δείκτη που αναφέρεται στο ένα από τα δύο μέρη του αριθμού. Έτσι, ο δείκτης 1 αναφέρεται στο πραγματικό και ο δείκτης 2 αναφέρεται στο φανταστικό μέρος του μιγαδικού.

- Ο τελεστής αναφοράς σε πεδίο εγγραφής '.', με σύνταξη:

<μεταβλητή> '.' <αναγνωριστικό>

όπου <μεταβλητή> μπορεί να είναι αναγνωριστικό, στοιχείο πίνακα ή άλλο πεδίο εγγραφής. Από τα δύο τελούμενα εισόδου του τελεστή αυτού, το πρώτο πρέπει να είναι τύπου εγγραφής και το δεύτερο όνομα πεδίου της, και η αναφορά γίνεται σύμφωνα με όσα αναφέρθηκαν πιο πάνω.

- Ο τελεστής σύνθεσης λίστας '[' έχει σύνταξη:

[' <ακολουθία εκφράσεων> ']

Ο τελεστής αυτός έχει σα μοναδικό τελούμενο εισόδου μια ακολουθία εκφράσεων, οι τιμές των οποίων αποδίδονται στα διαδοχικά στοιχεία μιας λίστας. Η αποτίμηση των εκφράσεων στην ακολουθία γίνεται από τα αριστερά προς τα δεξιά. Η ακολουθία εκφράσεων μπορεί να είναι κενή, οπότε δημιουργείται μια κενή λίστα, κάτι που ισοδυναμεί με μηδενική διεύθυνση πρώτου στοιχείου.

Οι επιμέρους εκφράσεις της λίστας πρέπει να είναι του ίδιου βασικού τύπου. Το αποτέλεσμα της σύνθεσης είναι τύπου λίστας, που ο μεταγλωττιστής χρησιμοποιεί είτε σε κάποια πράξη, είτε σε κάποια ανάθεση σε τιμή αριστερής προσπέλασης τύπου λίστας. Σε κάθε περίπτωση, ο τύπος των στοιχείων της λίστας πρέπει να είναι συμβατός με τον άλλο τελεστέο αν πρόκειται για πράξη, ή με την τιμή αριστερής προσπέλασης αν πρόκειται για ανάθεση.

Για τη σύνθεση της λίστας, ο μεταγλωττιστής παράγει κώδικα, ο οποίος να καλεί κάποια συνάρτηση δυναμικής εκχώρησης μνήμης για καθένα από τα στοιχεία της ακολουθίας, ώστε να κατασκευαστούν τα ζεύγη (περιεχόμενο, διεύθυνση επόμενου στοιχείου) που αποτελούν τη λίστα. Το τελευταίο στοιχείο λαμβάνει το 0 ως διεύθυνση επόμενου στοιχείου. Η τιμή του αποτελέσματος της σύνθεσης θα είναι έτσι η διεύθυνση του πρώτου στοιχείου της λίστας.

- Οι τελεστές προσήμου ADDOP. Ο μοναδικός τελεστής των τελεστών αυτών πρέπει να είναι αριθμητικού τύπου και το αποτέλεσμα της εφαρμογής τους είναι του ίδιου τύπου.
- Οι αριθμητικοί τελεστές POWEROP, MULOP, DIVOP και ADDOP. Η συμβατότητα τύπων των τελεστών αυτών και ο τύπος του αποτελέσματος της εφαρμογής τους καθορίζονται στον επόμενο πίνακα:

| A \ B | integer | real | complex |
|---------|---------|------------------------|------------------------|
| integer | integer | real | complex όχι POWEROP |
| real | real | real | complex όχι POWEROP |
| complex | complex | complex όχι POWEROP | complex όχι POWEROP |

Τύπος αποτελέσματος αριθμητικής έκφρασης A op B

Όπως δείχνει ο πίνακας, η μόνη περίπτωση μη συμβατότητας τύπων αφορά τον τελεστή POWEROP, για τον οποίο το δεξί τελούμενο εισόδου δε μπορεί να είναι τύπου complex και, εάν το αριστερό τελούμενο είναι τύπου complex, το δεξί μπορεί να είναι μόνο τύπου integer. Σε οποιαδήποτε περίπτωση που τα τελούμενα εισόδου είναι διαφορετικού τύπου, είναι απαραίτητο ο κώδικας να περιέχει μετατροπή του ενός τύπου στον τύπο του αποτελέσματος, η δε πράξη πρέπει να γίνεται για τον τύπο του αποτελέσματος.

Ο τελεστής ADDOP '+' εφαρμόζεται και σε τύπους λίστας, οπότε υλοποιεί την ένωση δύο λιστών. Η ένωση είναι η μόνη πράξη που επιτρέπεται σε λίστες, και μόνο σε λίστες με στοιχεία του ίδιου τύπου. Το αποτέλεσμα της ένωσης είναι του ίδιου τύπου λίστας, όπου η διεύθυνση επόμενου στοιχείου του τελευταίου στοιχείου της πρώτης λίστας έχει λάβει τιμή τη διεύθυνση πρώτου στοιχείου της δεύτερης λίστας.

- Οι σχεσιακοί τελεστές RELOP. Τα τελούμενα εισόδου μπορεί να είναι: (α) αριθμητικού τύπου εκτός από complex, οπότε, εάν είναι διαφορετικού τύπου, ο κώδικας πρέπει να μετατρέψει το τελούμενο τύπου integer σε real, και μετά την πιθανή μετατροπή, να κάνει τη σύγκριση για τον τύπο που προέκυψε, ή (β) τύπου character, οπότε η σύγκριση γίνεται με βάση την κωδικοποίηση αυτών. Ειδικά για τους τελεστές '.eq.' και '.ne.' τα τελούμενα εισόδου μπορεί να είναι και τύπου complex. Σε κάθε περίπτωση το αποτέλεσμα είναι τύπου logical.
- Ο τελεστής λογικής άρνησης NOTOP. Ο μοναδικός τελεστής του τελεστή αυτού πρέπει να είναι τύπου logical και το αποτέλεσμα είναι του ίδιου τύπου.
- Οι τελεστές λογικών πράξεων ANDOP και OROP. Τα τελούμενα εισόδου αυτών πρέπει να είναι τύπου logical. Το αποτέλεσμα της εφαρμογής τους είναι επίσης τύπου logical.
- Ο τελεστής δημιουργίας μιγαδικού αριθμού '(:)', με σύνταξη:
'(<έκφραση> ':' <έκφραση> ')

Τα δύο τελούμενα εισόδου του τελεστή αυτού πρέπει να είναι αριθμητικού τύπου εκτός από complex, και αν είναι τύπου integer, πρέπει να μετατραπούν σε real. Το αποτέλεσμα είναι ένας μιγαδικός αριθμός, στο πραγματικό μέρος του οποίου αποδίδεται η τιμή της αριστερής, και στο φανταστικό η τιμή της δεξιάς έκφρασης.

Παρά την αναπαράσταση των τύπων logical και character με ακέραιες τιμές, στη FORT200 δεν υπάρχει συμβατότητα μεταξύ αριθμητικών τιμών, λογικών τιμών και τιμών χαρακτήρων. Έτσι, η συμμετοχή αριθμητικών εκφράσεων σε λογικές μπορεί να γίνει μόνο μέσω των τελεστών RELOP. Αυτοί είναι οι μόνοι τελεστές που κατασκευάζουν λογικές εκφράσεις από αριθμητικές, καθώς μπορούν να δέχονται αριθμητικά τελούμενα και παράγουν αποτέλεσμα τύπου logical. Το αντίστροφο δεν είναι εφικτό, γι' αυτό και δε μπορεί μια λογική έκφραση να συμμετέχει σε μια αριθμητική. Παρόμοια, τιμές τύπου character μπορούν να συμμετάσχουν σε λογικές εκφράσεις μόνο μέσω των τελεστών RELOP. Τιμές τύπου character δε μπορούν να συμμετάσχουν σε αριθμητικές εκφράσεις.

Εξαίρεση στα παραπάνω αποτελεί ο τελεστής ‘()’ που από τη μία προσφέρει τη δυνατότητα έμμεσης συμμετοχής μιας έκφρασης σε μια μεγαλύτερη, αλλά και από την άλλη μπορεί να δώσει αποτέλεσμα τύπου που δε μπορούν να δώσουν άλλοι τελεστές.

Σε κάθε εφαρμογή τελεστή, και όταν συμμετέχουν περισσότερα από ένα τελούμενο εισόδου, η αποτίμηση αυτών γίνεται από αριστερά προς τα δεξιά. Σε κάθε περίπτωση απαιτείται αποτίμηση όλων των τελούμενων, μια που η FORT200 δεν υποστηρίζει βραχυκύκλωση.

Μέσα σε μια έκφραση μπορούν να υπάρχουν πολλοί τελεστές. Η σειρά εφαρμογής αυτών για την αποτίμηση της έκφρασης καθορίζεται από παρενθέσεις ή από κανόνες προτεραιότητας και προσεταιριστικότητας. Η σειρά προτεραιότητας των τελεστών καθορίζεται από τη σειρά που αυτοί αναγράφονται στον πρώτο από τους δύο παραπάνω πίνακες, από μεγαλύτερη προς μικρότερη προτεραιότητα. Η προσεταιριστικότητα των τελεστών, δίνεται στον ίδιο πίνακα. Αν δεν έχει νόημα, η προσεταιριστικότητα δεν είναι ορισμένη και αναγράφεται ως ‘-’.

Αξίζει να σημειωθεί ότι οι τελεστές RELOP δε μπορούν να έχουν καμία προσεταιριστικότητα, εφ’ όσον η εφαρμογή τους αλλάζει τον τύπο της έκφρασης. Έτσι, η έκφραση:

```
x+2 .gt. y .gt. 0
```

είναι λανθασμένη. Εφαρμογή οποιουδήποτε από τους δύο τελεστές RELOP ‘.gt.’ δίνει αποτέλεσμα τύπου logical, που δεν επιτρέπει την εφαρμογή του άλλου.

Σε περίπτωση τελεστών με ένα τελούμενο εισόδου, δεν επιτρέπεται διαδοχική εφαρμογή αυτών, εάν έχουν την ίδια προτεραιότητα. Για παράδειγμα, η έκφραση:

```
x.gt. +3
```

δεν είναι αποδεκτή, λόγω διαδοχικής εφαρμογής δύο τελεστών προσήμου ADDOP. Αντίθετα η έκφραση:

```
.not. -i.lt. -1
```

είναι αποδεκτή.

Ας θεωρήσουμε για παράδειγμα την έκφραση:

```
i*x-y-j*z .lt. -a/k/2 .and. .not. x(i+2,m) .a**r**2 .ge. 0.
```

στην οποία κατ’ αρχήν υποθέτουμε ότι οι τύποι των τιμών που συμμετέχουν είναι οι προβλεπόμενοι, καθώς και ότι x είναι διδιάστατος πίνακας εγγραφών, με a ένα πεδίο του.

Στην έκφραση αυτή ο τελεστής ANDOP ‘.and.’ έχει τη μικρότερη προτεραιότητα και δε μπορεί να εφαρμοστεί πριν την αποτίμηση και των δύο τελεστών του. Επίσης ο σχεσιακός τελεστής RELOP ‘.lt.’ στο αριστερό τελούμενο του προηγούμενου – το οποίο και αποτιμάται πριν από το δεξί – εφαρμόζεται μόνο αφού έχουν αποτιμηθεί και τα δύο τελούμενά του. Στο αριστερό τελούμενο του τελευταίου, οι δύο διαδοχικές αφαιρέσεις που ορίζονται από τον τελεστή ADDOP ‘-’ εφαρμόζονται από αριστερά προς τα δεξιά, λόγω αριστερής προσεταιριστικότητας του τελεστή αυτού.

Ο πρώτος τελεστής που θα εφαρμοστεί στην πιο πάνω έκφραση θα είναι ο πρώτος από αριστερά τελεστής MULOP ‘*’ που έχει μεγαλύτερη προτεραιότητα από τον ADDOP. Στη συνέχεια θα εφαρμοστεί ο πρώτος από αριστερά τελεστής ADDOP ‘-’, ενώ πριν εφαρμοστεί ο δεύτερος από τους δύο διαδοχικούς ADDOP, θα πρέπει να εφαρμοστεί ο δεύτερος τελεστής MULOP ‘*’ που υπάρχει στο δεξί τελούμενο του προηγούμενου. Η εφαρμογή του πρώτου από αριστερά τελεστή DIVOP ‘/’ προηγείται του δεύτερου, ενώ ο τελεστής προσήμου ADDOP ‘-’ εφαρμόζεται μετά τους δύο διαδοχικούς DIVOP, επιτρέποντας στη συνέχεια την εφαρμογή του τελεστή RELOP ‘.lt.’.

Έτσι ολοκληρώνεται η αποτίμηση του αριστερού τελεστέου του τελεστή ANDOP ‘.and.’ και μπορούμε να συνεχίσουμε με την αποτίμηση του δεξιού.

Στο τελούμενο του τελεστή NOTOP ‘.not.’ αποτιμάται πρώτα το αριστερό τελούμενο του τελεστή RELOP ‘.ge.’. Αυτό έχει δύο διαδοχικές εφαρμογές του τελεστή POWEROP ‘**’, όπου η δεξιά προσεταιριστικότητα επιβάλλει πρώτα την εφαρμογή του πρώτου από τα δεξιά. Για την εφαρμογή του δεύτερου τελεστή POWEROP απαιτείται η αποτίμηση του πεδίου εγγραφής ενός στοιχείου πίνακα. Λόγω αριστερής προσεταιριστικότητας, προηγείται η αποτίμηση των εκφράσεων των δεικτών του πίνακα, από αριστερά προς τα δεξιά, και ο έλεγχος ορθότητας των τιμών

τους. Στη συνέχεια υπολογίζεται η διεύθυνση βάσης της εγγραφής που μας ενδιαφέρει, αποτιμάται το πεδίο με όνομα *a*, και εφαρμόζεται ο δεύτερος τελεστής POWEROP. Τελειώνοντας, εφαρμόζεται ο τελεστής RELOP '.ge.', ακολουθούμενος από τον NOTOP. Η εφαρμογή του ANDOP μπορεί τώρα να πραγματοποιηθεί.

Σε περίπτωση που επιθυμούμε παρέμβαση στους πιο πάνω κανόνες πρέπει να χρησιμοποιήσουμε παρενθέσεις, όπως προβλέπει η σύνταξη των εκφράσεων της γλώσσας. Για παράδειγμα, η έκφραση:

$$- (-i+1) * (a - (b-c))$$

(α) επιτρέπει διαδοχική εφαρμογή του ίδιου τελεστή προσήμου, (β) επιβάλλει την αποτίμηση του δεξιού τελούμενου του ADDOP '-' πριν από το αριστερό, και (γ) επιβάλλει την εφαρμογή των τελεστών ADDOP '+' και '-' πριν από την εφαρμογή του MULOP '*'.

Ας σημειωθεί ότι παρενθέσεις μπορούν να χρησιμοποιηθούν και για βελτίωση της εμφάνισης μιας έκφρασης, χωρίς αναγκαστικά να επηρεάζουν τους κανόνες εφαρμογής των τελεστών που συμμετέχουν σε αυτήν.

Κλήση συναρτήσεων

Αν 'func' είναι το όνομα μιας συνάρτησης με αποτέλεσμα τύπου 'type', τότε η έκφραση

$$\text{func}(e_1, e_2, \dots, e_n)$$

είναι μια τιμή δεξιάς προσπέλασης τύπου 'type'. Ειδικά αν το αποτέλεσμα της συνάρτησης είναι τύπου λίστας, τότε η πιο πάνω έκφραση είναι αριστερής προσπέλασης αυτού του τύπου. Η αποτίμηση της έκφρασης αυτής γίνεται με την εκτέλεση του κώδικα της μονάδας της συνάρτησης, με τις εξής προϋποθέσεις:

- Ο αριθμός *n* των εκφράσεων στις παρενθέσεις – οι πραγματικές παράμετροι – πρέπει να είναι ίσος με τον αριθμό των τυπικών παραμέτρων.
- Ο τύπος κάθε πραγματικής παραμέτρου με πέρασμα κατ' αξία πρέπει να είναι συμβατός με τον τύπο της αντίστοιχης τυπικής παραμέτρου, σύμφωνα με τους κανόνες συμβατότητας για ανάθεση που περιγράφονται πιο κάτω.
- Ο τύπος κάθε πραγματικής παραμέτρου με πέρασμα κατ' αναφορά πρέπει να ταυτίζεται με τον τύπο της αντίστοιχης τυπικής παραμέτρου.

Κατά την κλήση μιας συνάρτησης οι πραγματικές παράμετροι αποτιμώνται από αριστερά προς τα δεξιά και τοποθετούνται στη στοίβα, απ' όπου θα μπορεί να τους χρησιμοποιήσει η μονάδα της συνάρτησης. Με την έξοδο από τη συνάρτηση, θα πρέπει η τιμή του αποτελέσματος να βρίσκεται σε προκαθορισμένο σημείο αποθήκευσης, επίσης στη στοίβα. Με κάθε κλήση μιας συνάρτησης, η στοίβα μεταβάλλεται, κι έτσι ο χώρος δεδομένων που η συνάρτηση διατηρεί τις παραμέτρους και το αποτέλεσμά της, δηλαδή το *εγγράφημα δραστηριοποίησης* της συνάρτησης, είναι δυναμικός.

Εξαίρεση στα παραπάνω αποτελούν οι προκαθορισμένες συναρτήσεις στοιχείων λίστας, οι οποίες δε λειτουργούν ως συναρτήσεις, αλλά απλά απομονώνουν και επιστρέφουν ένα μέρος της λίστας που δίνεται ως παράμετρος. Το μέρος αυτό μάλιστα επιστρέφεται ως τιμή αριστερής προσπέλασης, όπως ακριβώς ένα στοιχείο πίνακα ή ένα πεδίο εγγραφής.

Εκτός από τις προκαθορισμένες συναρτήσεις στοιχείων λίστας, ορίζεται και η προκαθορισμένη συνάρτηση LENGTH(), η οποία δέχεται ως παράμετρο μια λίστα και επιστρέφει το πλήθος των στοιχείων της λίστας, καθώς και η προκαθορισμένη συνάρτηση NEW(), η οποία δέχεται ως παράμετρο μια τιμή και επιστρέφει μια λίστα ενός στοιχείου, με στοιχείο την τιμή-παράμετρο, και με διεύθυνση επόμενου στοιχείου ίση με 0.

Σε περίπτωση που η συνάρτηση που καλείται δεν έχει ακόμα οριστεί, η θέση της εντολής κλήσης πρέπει να σημειωθεί για μετέπειτα επάνοδο της ανάλυσης σε αυτή, σύμφωνα με την τεχνική του μπαλώματος.

Εντολές

Η FORT200 υποστηρίζει απλές και δομημένες εντολές. Μια δομημένη εντολή της FORT200 περιέχει ένα ή δύο σύνολα άλλων εντολών που το καθένα έχει τη δική του εμβέλεια, στην οποία μπορούν να οριστούν μεταβλητές. Μέσα σε κάθε ένα από αυτά τα σύνολα πρέπει να υπάρχει τουλάχιστον μία άλλη εντολή, απλή ή δομημένη.

Οι απλές εντολές της FORT200 είναι:

- Η εντολή ανάθεσης
- Οι εντολές άμεσου άλματος
- Οι απλές εντολές ελέγχου ροής
- Η εντολή κλήσης υπορουτίνας
- Οι εντολές εισόδου/εξόδου
- Η εντολή continue
- Η εντολή return
- Η εντολή stop

Οι δομημένες εντολές της FORT200 είναι:

- Η εντολή διακλάδωσης
- Η εντολή βρόχου

Η εντολή ανάθεσης

Η εντολή αυτή αποδίδει την τιμή μιας έκφρασης σε μια τιμή αριστερής προσπέλασης, δηλαδή σε μια μεταβλητή, ένα πεδίο εγγραφής ή ένα στοιχείο πίνακα ή λίστας του χώρου δεδομένων του προγράμματος. Στην περίπτωση πίνακα, της ανάθεσης πρέπει να προηγηθεί η αποτίμηση των εκφράσεων των δεικτών, ενώ για όλες τις περιπτώσεις σύνθετου τύπου πρέπει να προηγηθεί και ο υπολογισμός της τιμής αριστερής προσπέλασης του πεδίου ή στοιχείου.

Η ανάθεση γίνεται με αποθήκευση της τιμής της έκφρασης στη διεύθυνση που παριστάνει η τιμή αριστερής προσπέλασης.

Παραδείγματα αναθέσεων είναι τα εξής:

```
a = 0.01
c(i-1, mm.k(j)) = x**2+1
s = NEW(mm)
t = [2] + CDR([5,3,-4])
```

όπου mm είναι εγγραφή, c και mm.k είναι πίνακες, και s και t είναι λίστες.

Για να μπορεί να γίνει ανάθεση, πρέπει η τιμή της έκφρασης να είναι συμβατού τύπου με τον τύπο της αριστερής προσπέλασης.

Δύο τύποι είναι συμβατοί για ανάθεση, εάν:

- (α) ταυτίζονται, ή αλλιώς
- (β) είναι αριθμητικοί, εκτός από τύπο complex, οπότε συμβαίνει μετατροπή σε πραγματικό για ανάθεση από τύπο integer σε τύπο real, ή αποκοπή του κλασματικού μέρους για ανάθεση από τύπο real σε τύπο integer.

Ανάθεση μεταξύ σύνθετων τύπων επιτρέπεται μόνο για λίστες στοιχείων συμβατού τύπου. Τότε, η διεύθυνση πρώτου στοιχείου της λίστας που παριστάνεται με το δεξί μέλος αποδίδεται στην έκφραση λίστας του αριστερού μέλους.

Μια ανάθεση λίστας ελλοχεύει τον κίνδυνο κάποια στοιχεία λίστας να μείνουν “ξεκρέμαστα”, δηλαδή να μην είναι πλέον προσπελάσιμα, κάτι που επιβαρύνει το σύστημα δυναμικής εκχώρησης μνήμης με κίνδυνο ασφυξίας. Για το σκοπό αυτό ο μεταγλωττιστής θα πρέπει να παράγει εδώ επιπλέον κώδικα ανίχνευσης και συλλογής ξεκρέμαστων στοιχείων.

Εκτός από την παραπάνω ανάθεση, η FORT200 υποστηρίζει και δύο ακόμα ειδικές μορφές ανάθεσης:

(α) Την ανάθεση στη μεταβλητή που παριστάνεται από το όνομα μιας συνάρτησης. Η τιμή αριστερής προσπέλασης αυτής είναι διεύθυνση στη στοίβα. Κάθε συνάρτηση πρέπει να έχει τουλάχιστον μία τέτοια ανάθεση. Η ανάθεση αυτή κατά τα άλλα ακολουθεί τους πιο πάνω κανόνες.

(β) Την ανάθεση ενός ορμαθού χαρακτήρων σε μια μεταβλητή πίνακα στοιχείων τύπου `character`. Σε μια τέτοια ανάθεση, αντιγράφονται στον πίνακα διαδοχικοί χαρακτήρες από τον ορμαθό, μέχρι να τελειώσουν οι χαρακτήρες του ορμαθού, ή μέχρι να γεμίσει ο πίνακας. Αν υπάρχει χώρος στον πίνακα, μετά τους χαρακτήρες του ορμαθού αποθηκεύεται ο χαρακτήρας με τιμή 0.

Οι εντολές άμεσου άλματος

Αυτές είναι οι εντολές `goto`, που είναι δύο:

- Το αδέσμευτο `goto`, με παράμετρο μια ετικέτα. Η εντολή αυτή εκτελεί άλμα και μεταφέρει τη ροή του προγράμματος στην εντολή που ακολουθεί την ετικέτα. Παράδειγμα αδέσμευτου `goto` είναι το εξής:

```
goto 100
```

- Το υπολογιζόμενο `goto`, με παραμέτρους μια βαθμωτή μεταβλητή τύπου `integer` και μια λίστα ετικετών σε παρενθέσεις. Η εντολή αυτή εκτελεί άλμα και μεταφέρει τη ροή του προγράμματος στην εντολή που ακολουθεί μια από τις ετικέτες της λίστας, με βάση την τιμή της μεταβλητής, ή συνεχίζει τη ροή στην επόμενη εντολή. Αν η λίστα περιέχει n ετικέτες και η τιμή της μεταβλητής είναι μεταξύ 1 και n , τότε με την τιμή αυτή να χρησιμοποιείται σα δείκτης στη λίστα, εκτελείται άλμα στην αντίστοιχη εντολή. Σε κάθε περίπτωση που η μεταβλητή έχει τιμή εκτός της παραπάνω περιοχής δεν εκτελείται άλμα. Παράδειγμα υπολογιζόμενου `goto` είναι το εξής:

```
goto i, (100, 200, 400, 100)
```

όπου η μεταβλητή i πρέπει να έχει τιμή 1, 2, 3 ή 4, για να εκτελέσει το άλμα. Παρατηρήστε ότι οι ετικέτες της λίστας δεν είναι αναγκαστικά διαφορετικές μεταξύ τους.

Μια ετικέτα που συμμετέχει σε μία από τις παραπάνω εντολές πρέπει να είναι μοναδικά ορισμένη σε μια εμβέλεια και η ορατότητά της να καλύπτει την εντολή. Σαν αποτέλεσμα αυτού του περιορισμού, δε μπορεί να βρίσκεται σε εμβέλεια εσωτερική αυτής στην οποία υπάρχει η εντολή.

Οι απλές εντολές ελέγχου ροής

Αυτές είναι οι εντολές `if`, που είναι δύο:

- Το αριθμητικό `if`, με παραμέτρους μια αριθμητική έκφραση σε παρενθέσεις και μια λίστα τριών ετικετών. Η εντολή αυτή αποτιμά την έκφραση και εκτελεί άλμα στην εντολή που δείχνει η πρώτη ετικέτα, στην εντολή που δείχνει η δεύτερη ετικέτα, ή στην εντολή που δείχνει η τρίτη ετικέτα, αν η τιμή της έκφρασης είναι μικρότερη από, ίση με ή μεγαλύτερη από μηδέν, αντίστοιχα. Παράδειγμα αριθμητικού `if` είναι:

```
if (a(i)-x) 100, 200, 300
```

Οι ετικέτες της λίστας – που δεν είναι αναγκαστικά διαφορετικές μεταξύ τους – ακολουθούν τον ίδιο περιορισμό με τις ετικέτες στις εντολές άμεσου άλματος.

- Το λογικό `if`, με παραμέτρους μια λογική έκφραση σε παρενθέσεις και μια απλή εντολή χωρίς ετικέτα. Η εντολή αυτή αποτιμά τη λογική έκφραση και εκτελεί την απλή εντολή, μόνο αν η τιμή της έκφρασης είναι «Αληθής». Παράδειγμα λογικού `if` είναι:

```
if (x.gt.0.0.and.x.lt.a(i)) y(i) = i*x
```

Παρατηρήστε ότι και στις δύο παραπάνω εντολές ο τύπος της έκφρασης σε παρενθέσεις δεν καθορίζεται από τη σύνταξη, αλλά πρέπει να ελέγχεται κατά τη σημασιολογική ανάλυση.

Η εντολή κλήσης υπορουτίνας

Η εντολή αυτή μεταφέρει τη ροή του κώδικα σε κάποια υπορουτίνα. Παράδειγμα κλήσης υπορουτίνας είναι το παρακάτω:

```
call subA(x,n+1,matrix,s(i)**2)
```

Στην εντολή κλήσης υπορουτίνας οι περιορισμοί στις πραγματικές παραμέτρους είναι ταυτόσημοι με τους αντίστοιχους περιορισμούς στην κλήση συνάρτησης που περιγράφηκαν νωρίτερα. Ο μηχανισμός κλήσης είναι επίσης ο ίδιος με το μηχανισμό κλήσης μιας συνάρτησης.

Όπως και με τις συναρτήσεις, σε περίπτωση που η υπορουτίνα που καλείται δεν έχει ακόμα οριστεί, η θέση της εντολής κλήσης πρέπει να σημειωθεί για μετέπειτα επάνοδο της ανάλυσης σε αυτή.

Οι εντολές εισόδου/εξόδου

Αυτές είναι οι εντολές ανάγνωσης (read) και εγγραφής (write) δεδομένων. Συντάσσονται με το όνομα της εντολής και μια λίστα στοιχείων, που στην περίπτωση ανάγνωσης είναι τιμές αριστερής προσπέλασης, ενώ στην περίπτωση εγγραφής είναι εκφράσεις ή ορμαθοί χαρακτήρων. Και στις δύο περιπτώσεις, η λίστα μπορεί να περιλαμβάνει τη μορφή του *υπονοούμενου βρόχου* που περιγράφεται πιο κάτω. Παραδείγματα εντολών εισόδου/εξόδου είναι:

```
read n, (y(i).a, y(i).b, i=1, n), x
write "Temperature: ", f, "F, or ", 5/9*(f-32), "C."
write "Squares: ", (x(i)**2, ", ", i=1, 99), x(100)**2
```

Η είσοδος και η έξοδος γίνονται στα συνήθη αρχεία εισόδου/εξόδου χωρίς προδιαγραφές, δηλαδή προηγούμενο καθορισμό του τύπου και της μορφής των στοιχείων που διαβάζονται ή γράφονται. Με κάθε εντολή εξόδου γράφεται μια γραμμή κειμένου στην έξοδο του προγράμματος.

Ο υπονοούμενος βρόχος είναι μια μορφή στοιχείου που *υπονοεί* την επανάληψη της εντολής για τη λίστα στοιχείων που είναι μέσα στις παρενθέσεις.

Μετά τη λίστα στοιχείων και πριν τη δεξιά παρένθεση πρέπει να υπάρχει το πεδίο επανάληψης, το οποίο συντάσσεται με τον τρόπο που συντάσσεται το πεδίο επανάληψης στην εντολή βρόχου που θα δούμε σε λίγο. Η λίστα στοιχείων του υπονοούμενου βρόχου μπορεί να χρησιμοποιεί τη μεταβλητή ελέγχου του βρόχου, η οποία ορίζεται στο πεδίο επανάληψης.

Ο υπονοούμενος βρόχος επιτρέπει φωλιάσματα.

Η εντολή continue

Αυτή είναι μια εικονική εντολή, η εκτέλεση της οποίας δεν έχει κανένα αποτέλεσμα. Μ' άλλα λόγια ο μεταγλωττιστής δεν παράγει κώδικα γι' αυτήν.

Η εντολή continue χρησιμοποιείται για να συνοδεύει μια ετικέτα, όταν στο σημείο που θέλουμε να τοποθετήσουμε την ετικέτα δεν υπάρχει άλλη εντολή. Έτσι για παράδειγμα μπορούμε να υλοποιήσουμε άμεσο άλμα στο τέλος μιας δομημένης εντολής.

Η εντολή return

Αυτή η εντολή επιτρέπεται να εμφανίζεται μόνο μέσα σε υποπρογράμματα, και είναι η μόνη εντολή που επιστρέφει τη ροή του προγράμματος από ένα υποπρόγραμμα στη μονάδα που το κάλεσε, και στο σημείο της κλήσης αυτού.

Η εντολή stop

Αυτή είναι η εντολή τερματισμού της εκτέλεσης ενός προγράμματος. Με την εκτέλεση αυτής σηματοδοτείται το τέλος του προγράμματος, και καμία άλλη εντολή δεν εκτελείται στη συνέχεια.

Η εντολή διακλάδωσης

Η εντολή διακλάδωσης είναι μια δομημένη παραλλαγή εντολής if με παραμέτρους μια λογική έκφραση σε παρενθέσεις και ένα ή δύο σύνολα εντολών με μία τουλάχιστον εντολή το κάθε σύνολο. Κάθε σύνολο ορίζει μια εσωτερική εμβέλεια και μπορεί να έχει δηλώσεις μεταβλητών.

Η εκτέλεση της εντολής διακλάδωσης ξεκινά με την αποτίμηση της λογικής έκφρασης. Αν αυτή έχει τιμή «Αληθής», εκτελούνται οι εντολές του συνόλου που ακολουθεί τη λέξη-κλειδί “then”, και τελειώνει με την αντίστοιχη λέξη-κλειδί “else” ή, αν αυτή δεν υπάρχει στη συγκεκριμένη εντολή, με την αντίστοιχη λέξη-κλειδί “endif”. Εάν η λογική έκφραση έχει τιμή «Ψευδής» και υπάρχει η αντίστοιχη λέξη-κλειδί “else”, τότε εκτελούνται οι εντολές του συνόλου που ακολουθεί αυτή τη λέξη, και τελειώνει με την αντίστοιχη λέξη-κλειδί “endif”, ενώ αν δε υπάρχει η λέξη-κλειδί “else”, δεν εκτελείται καμία εντολή.

Ένα παράδειγμα εντολής διακλάδωσης είναι το παρακάτω:

```

if (a(i+1).gt.13.5.and.z(j)) then
    a(i-1) = f(z(j+1),i-1)*2
    if (x.eq.0) return
else
    integer k,n
    read k,n
    if (n-i) 100,101,102
100  a(i-1) = a(n+i)
    return
102  a(i-1) = f(z(n-i),i-1)*k+1
101  continue
endif

```

όπου φαίνεται και ένα παράδειγμα χρήσης της εντολής continue.

Οι λέξεις-κλειδιά “then”, “else” και “endif” αποτελούν τα διαχωριστικά μεταξύ των εσωτερικών εμβελειών της εντολής διακλάδωσης. Η λογική έκφραση αποτιμάται στην εμβέλεια στην οποία βρίσκεται η εντολή.

Η σύνταξη της γλώσσας επιτρέπει φωλιάσματα στις εντολές διακλάδωσης, κι επομένως ο σημασιολογικός αναλυτής πρέπει να ακολουθεί τις φωλιασμένες εμβέλειες από τους συντακτικούς κανόνες της FORT200.

Η εντολή βρόχου

Η εντολή βρόχου ή εντολή do είναι μια δομημένη εντολή με παραμέτρους ένα πεδίο επανάληψης και ένα σύνολο εντολών με τουλάχιστον μία εντολή. Το σύνολο αυτό ορίζει μια εσωτερική εμβέλεια και μπορεί να έχει δηλώσεις μεταβλητών.

Το πεδίο επανάληψης περιέχει το όνομα μιας βαθμωτής μεταβλητής, που ονομάζεται μεταβλητή ελέγχου του βρόχου, και δύο ή τρεις αριθμητικές εκφράσεις, που αποτιμώμενες δίνουν: η πρώτη την αρχική τιμή της μεταβλητής ελέγχου, η δεύτερη την τελική τιμή αυτής, και η τρίτη το βήμα αύξησης της μεταβλητής ελέγχου μεταξύ διαδοχικών επαναλήψεων του βρόχου. Εάν η τρίτη έκφραση απουσιάζει, το βήμα θεωρείται ότι είναι 1.

Η εκτέλεση της εντολής do ξεκινά με αποτίμηση των εκφράσεων του πεδίου επανάληψης, και ανάθεση της αρχικής τιμής στη μεταβλητή ελέγχου. Στη συνέχεια εκτελούνται οι εντολές του συνόλου που ακολουθεί το πεδίο επανάληψης και τελειώνει με τη λέξη-κλειδί “enddo”. Αν δεν υπάρχει παρέμβαση στη ροή του προγράμματος, αυξάνεται κατάλληλα η τιμή της μεταβλητής ελέγχου του βρόχου, και η νέα τιμή συγκρίνεται με την τελική. Εάν είναι μικρότερη από ή ίση με αυτή, η διαδικασία επαναλαμβάνεται από την εκτέλεση των εντολών του συνόλου και κάτω. Διαφορετικά, η εκτέλεση του βρόχου τερματίζεται.

Ας σημειωθεί ότι οι εκφράσεις του πεδίου επανάληψης αποτιμώνται *μόνο* μία φορά, και ακόμα ότι οι εντολές του συνόλου εκτελούνται *τουλάχιστον* μία φορά. Κάθε παρέμβαση στη ροή του προγράμματος που τη μεταφέρει έξω από το εσωτερικό σύνολο εντολών του βρόχου τερματίζει την εκτέλεση αυτού.

Ένα παράδειγμα εντολής βρόχου δίνεται παρακάτω:

```
do i=1,n
    integer j,k
    a(i*m) = y/2
    k = f(n*m,a,x)
    do j=1,m
        b(j,i+k) = (m-k)*1.
        if (j .eq. k) goto 100
    enddo
100 continue
enddo
```

όπου για άλλη μια φορά δίνεται κι ένα παράδειγμα χρήσης της εντολής continue.

Η σημασιολογία της εντολής βρόχου συμπληρώνεται από τους ακόλουθους κανόνες:

1. Η μεταβλητή ελέγχου του βρόχου πρέπει να είναι τύπου integer.
2. Ο τύπος των εκφράσεων του πεδίου επανάληψης πρέπει να είναι integer.
3. Η τρίτη έκφραση του πεδίου επανάληψης, αν υπάρχει, πρέπει να έχει τιμή μεγαλύτερη από 0.
4. Η μεταβλητή ελέγχου δεν επιτρέπεται να λαμβάνει τιμή μέσα στο εσωτερικό σύνολο εντολών του βρόχου – άρα ούτε να μεταδίδεται κατ’ αναφορά σε κάποιο υποπρόγραμμα που καλείται μέσα από το σύνολο αυτό.

Η λέξη-κλειδί “enddo” αποτελεί το κάτω όριο της εσωτερικής εμβέλειας του βρόχου. Οι αριθμητικές εκφράσεις του πεδίου επανάληψης αποτιμώνται στην εμβέλεια στην οποία βρίσκεται η εντολή, ενώ η μεταβλητή ελέγχου πρέπει να είναι ορισμένη σε αυτήν ή εξωτερική της εμβέλειας.

Η σύνταξη της γλώσσας επιτρέπει φωλιάσματα στις εντολές βρόχου, κι επομένως ο σημασιολογικός αναλυτής πρέπει να ακολουθεί τις φωλιασμένες εμβέλειες από τους συντακτικούς κανόνες της FORT200.

Ετικέτες

Μια ετικέτα δείχνει τη θέση μιας εντολής στον κώδικα του προγράμματος. Δεν αντιπροσωπεύει κάτι το εκτελέσιμο, δηλαδή ο μεταγλωττιστής δεν παράγει κώδικα για αυτήν, αλλά χρησιμοποιείται σε συνδυασμό με άλλες εντολές για μεταφορά της ροής του προγράμματος σε αυτήν.

Μια ετικέτα δε δηλώνεται – όπως μία μεταβλητή – στην αρχή μιας εμβέλειας, αλλά ορίζεται τη στιγμή που θα συναντηθεί, δίπλα δηλαδή στην αντίστοιχη εντολή. Είναι μάλιστα πολύ πιθανό να ορίζεται, αφού έχει ήδη χρησιμοποιηθεί σε κάποια προηγούμενη εντολή του προγράμματος. Η ορατότητα μιας ετικέτας είναι ολόκληρη η εμβέλεια στην οποία ορίζεται.

Για τον παραπάνω λόγο, η ολοκλήρωση της ανάλυσης των εντολών που χρησιμοποιούν μια ετικέτα, όπως οι goto και if, σε περίπτωση που αυτή δεν είναι ακόμα ορισμένη, γίνεται μόλις αυτή οριστεί, με την τεχνική του μπαλώματος. Επομένως, οι θέσεις των εντολών αυτών πρέπει να σημειώνονται κατά τη σημασιολογική ανάλυσή τους, για μετέπειτα επάνοδο σε αυτές.

Δήλωση τερματισμού μονάδας

Η λεκτική μονάδα end δεν παριστάνει εκτελέσιμη εντολή. Αν ο μεταγλωττιστής συναντήσει μια δήλωση τερματισμού μονάδας, ολοκληρώνει την ανάλυση αυτής. Ύπαρξη άλλης εντολής της μονάδας μετά τη δήλωση τερματισμού αποτελεί συντακτικό σφάλμα.

Η εκτέλεση μιας μονάδας πρέπει να τερματίζεται με κάποια εντολή `return` ή εντολή `stop`. Σε περίπτωση που συναντηθεί δήλωση `end`, αλλά η εντολή που προηγείται αυτής δεν είναι μια από τις δύο αυτές εντολές, ο μεταγλωττιστής θα προσθέσει μια τέτοια εντολή ως εξής:
Αν η μονάδα είναι η κύρια μονάδα του προγράμματος, προστίθεται η εντολή `stop`, ενώ αν είναι μονάδα υποπρογράμματος, προστίθεται η εντολή `return`.