ΚΕΦΑΛΑΙΟ 1
ΤΑ ΒΑΣΙΚΑ ΓΙΑ ΤΟΝ CLIPPER

Λίγα Λόγια για τις Γλώσσες Προγραμματισμού

Ο Clipper είναι μια καθαρά επαγγελματική γλώσσα προγραμματισμού, με την οποία μπορούμε να κάνουμε μεταγλώττιση (compilation) και ερμηνεία (interpretation) των προγραμμάτων μας.

Γλώσσα προγραμματισμού είναι ένα σύνολο από εντολές που τις γράφουμε εμείς, έτσι ώστε να λειτουργούν για να εκτελούν κάποια συγκεκριμένη εργασία. Ο υπολογιστής ερμηνεύει κάθε εντολή του προγράμματος και ενεργεί σύμφωνα μ' αυτήν. Η γλώσσα προγραμματισμού διευκολύνει την επικοινωνία του χρήστη με τον υπολογιστή και περιλαμβάνει πολλά στοιχεία της αγγλικής γλώσσας για να μας διευκολύνει στη σύνταξη και στην κατανόηση των προγραμμάτων.

Πρέπει να έχουμε υπόψη μας ότι ο υπολογιστής δεν καταλαβαίνει την αγγλική ή οποιαδήποτε άλλη γλώσσα στην οποία είναι γραμμένα τα προγράμματά μας, αλλά μόνο τη δική του γλώσσα, τη γλώσσα μηχανής. Έτσι, για να εκτελεστεί, να «τρέξει» όπως λέμε, ένα πρόγραμμα στον υπολογιστή, θα πρέπει προηγουμένως να μετατραπεί σε γλώσσα μηχανής, για να μπορεί να το καταλάβει ο υπολογιστής.

Υπάρχουν ***γλώσσες χαμηλού επιπέδου*** *(low level languages)* και ***γλώσσες υψηλού επιπέδου*** *(high level languages)*. Ο διαχωρισμός αυτός δεν έχει να κάνει με τις δυνατότητες μιας γλώσσας, αλλά με το πόσο εύκολα μπορεί να την γράψει και να την καταλάβει ο άνθρωπος. Γλώσσες χαμηλού επιπέδου λέμε αυτές που είναι πιο κοντά στον υπολογιστή, όπως είναι η γλώσσα μηχανής (machine language) και η συμβολική γλώσσα (assembly language).

Η γλώσσα μηχανής είναι γραμμένη με τα δυαδικά ψηφία 0 και 1, αλλά μπορεί να κάνει λειτουργίες που είναι δύσκολο να τις κάνουμε με γλώσσες υψηλού επιπέδου, όπως είναι η μεταφορά δεδομένων από και προς τις περιφερειακές συσκευές (σκληρός δίσκος, εκτυπωτής κ.ά.).

Επειδή ο προγραμματισμός με χρήση των δυαδικών ψηφίων 0 και 1 είναι πάρα πολύ δύσκολος, επινοήθηκε η γλώσσα assembly, όπου σε κάθε εντολή της assembly αντιστοιχεί μία εντολή γλώσσας μηχανής. Η assembly είναι κατά ένα επίπεδο γλώσσα υψηλότερου επιπέδου σε σχέση με τη γλώσσα μηχανής, αλλά θεωρείται κι αυτή γλώσσα χαμηλού επιπέδου. Έτσι, για παράδειγμα, για να προσθέσουμε δύο αριθμούς, γράφουμε στην assembly : ***ADD A, B***, κάτι που είναι αμέσως κατανοητό και από κάποιον τρίτο.

Για να κάνουμε το ίδιο στη γλώσσα μηχανής θα πρέπει να γράψουμε μια σειρά από ψηφία 0 και 1, κάτι που είναι πολύ κουραστικό, πολύ επικίν-

δυνο για λάθη και πολύ δύσκολο για να το καταλάβει κάποιος τρίτος. Υπάρχει ένα ειδικό πρόγραμμα, ο ***συμβολομεταφραστής*** *(assembler)*, ο οποίος μεταφράζει κάθε εντολή της assembly σε εντολή γλώσσας μηχανής.

Όσον αφορά τώρα τις γλώσσες υψηλού επιπέδου, όπως είναι ο Clipper, η C, η Pascal, η Basic, η Cobol, η Visual Basic κ.ά., αυτές είναι πιο κοντά στον άνθρωπο, κι αυτό σημαίνει πρακτικά ότι περιέχουν πολλές εντολές που είναι γραμμένες στην αγγλική γλώσσα και μπορούμε έτσι να τις καταλάβουμε και να τις επεξεργαστούμε καλύτερα απ' ό,τι μια γλώσσα χαμηλού επιπέδου. Πρέπει να έχουμε υπόψη μας ότι σε μια εντολή γλώσσας υψηλού επιπέδου αντιστοιχούν μια ή και περισσότερες εντολές γλώσσας μηχανής.

Για τις γλώσσες υψηλού επιπέδου υπάρχουν δύο τρόποι για να μεταφραστούν τα προγράμματά τους σε γλώσσα μηχανής : η ***μεταγλώττιση*** *(compilation)* και η ***ερμηνεία*** *(interpretation)*. Όταν ένα πρόγραμμα μεταγλωττίζεται (compiled), κάθε εντολή του προγράμματος μετατρέπεται στην αντίστοιχη ή στις αντίστοιχές της στη γλώσσα μηχανής και δημιουργείται ένα ***εκτελέσιμο αρχείο*** *(executable file)*, το οποίο μπορούμε να το τρέξουμε κατευθείαν μόνο του και όποτε θέλουμε, γράφοντας μόνο το όνομά του, ακόμη κι αν έχει χαθεί το αρχικό πρόγραμμα από το οποίο προήλθε. Όταν, όμως, κάνουμε και την παραμικρή αλλαγή στο αρχικό πρόγραμμα, τότε θα πρέπει να κάνουμε πάλι μεταγλώττιση για να ισχύσουν οι αλλαγές.

Όταν ένα πρόγραμμα ερμηνεύεται (interpreted), οι εντολές του προγράμματος μεταφράζονται μία-μία κατά τον χρόνο εκτέλεσης και δεν δημιουργείται εκτελέσιμο αρχείο. Ακόμη, όταν κάνουμε κάποιες αλλαγές στο πρόγραμμά μας, τότε μπορούμε να το εκτελέσουμε αμέσως χωρίς να χρειαστεί να γίνει ξανά όλη διαδικασία της μεταγλώττισης. Η ερμηνεία έχει, όμως, το μειονέκτημα ότι είναι σχετικά αργή σε σχέση με τη μεταγλώττιση. Ο Clipper συνδυάζει στοιχεία μεταγλώττισης και ερμηνείας.

Λίγα Λόγια για τον Clipper

Ο Clipper περιέχει περισσότερες από 350 εντολές και συναρτήσεις, οι οποίες μας παρέχουν απεριόριστες δυνατότητες και είναι μια καθαρά επαγγελματική γλώσσα προγραμματισμού με το μεγάλο πλεονέκτημα ότι μας λύνει τα χέρια στον χειρισμό των αρχείων.

Ο Clipper ξεκίνησε σαν μεταγλωττιστής για τη δημοφιλή γλώσσα διαχείρισης βάσεων δεδομένων dBASE και βοήθησε στη γρήγορη εκτέλεση των εντολών της. Ο Clipper συνδυάζει στοιχεία από γλώσσες υψηλού επιπέδου όπως η Cobol, που κάνει πολύ καλό χειρισμό αρχείων και γλώσσες όπως η Pascal και η C, που έχουν ισχυρές και εύχρηστες εντολές στον προγραμματισμό.

Ο Clipper έχει δικές του εντολές ελέγχου και επανάληψης, όπως όλες οι γλώσσες προγραμματισμού. Εκείνο που αλλάζει, βέβαια, είναι ο τρόπος

σύνταξης των εντολών αυτών. Θα κάνουμε εδώ μια γενική παρουσίαση των βασικών εντολών του Clipper και θα δούμε ένα απλό πρόγραμμα.

Η εντολή ***IF*** στον Clipper συντάσσεται ως εξής :

```
IF  <συνθήκη>
        <εντολή-1>
ELSE
        <εντολή-2>
ENDIF
```

Η εντολή ***WHILE*** στον Clipper συντάσσεται ως εξής :

```
DO WHILE <συνθήκη>
        <εντολές>
ENDDO
```

Η εντολή ***CASE*** στον Clipper συντάσσεται ως εξής :

```
DO CASE
        CASE <συνθήκη-1>
                <εντολή-1>
        CASE <συνθήκη-2>
                <εντολή-2>
        OTHERWISE
                <εντολή-3>
ENDCASE
```

Η εντολή επανάληψης ***FOR*** στον Clipper συντάσσεται ως εξής :

```
FOR I:=1 TO 10 STEP 2
        <εντολές>
NEXT I
```

Ο Clipper χρησιμοποιεί διαδικασίες (procedures) και συναρτήσεις (functions) όπως όλες οι γλώσσες προγραμματισμού και μπορούμε να μεταβιβάσουμε σ' αυτές τιμές κατ' αξία και κατ' αναφορά.

Με τον Clipper μπορούμε πολύ εύκολα να σχεδιάσουμε γραμμές και ορθογώνια πλαίσια στην οθόνη και να χρωματίσουμε με μια μεγάλη ποικιλία χρωμάτων. Ακόμη, μπορούμε πολύ εύκολα να δημιουργήσουμε μενού για την άμεση επιλογή εργασιών.

Οι μεταβλητές που χρησιμοποιεί ο Clipper είναι καθολικές (public), ιδιωτικές (private), τοπικές (local) και στατικές (static) και δεν είμαστε υποχρεωμένοι να δηλώσουμε τον τύπο δεδομένων μιας μεταβλητής στην αρχή του προγράμματος. Ο Clipper καθορίζει μόνος του τους τύπους δεδομένων των μεταβλητών από τις εκχωρήσεις τιμών που γίνονται στη διάρκεια του

προγράμματος. Έτσι, μπορεί μια μεταβλητή να αλλάξει τύπο δεδομένων μέσα στο ίδιο το πρόγραμμα πολλές φορές.

Για να γράψουμε σχόλια στον Clipper μπορούμε να χρησιμοποιήσουμε τα σύμβολα /* και */, όπου τα σχόλια αρχίζουν με το /* και κλείνουν με το */. Ακόμη, με το * στην πρώτη στήλη μιας γραμμής, θεωρείται όλη η γραμμή σαν σχόλιο. Σχόλια εισάγουμε επίσης και με τα σύμβολα // και &&, αλλά το σχόλιο ισχύει απ' αυτό το σημείο μέχρι το πέρας της γραμμής.

Για την εκχώρηση τιμών σε μεταβλητές μπορούμε να χρησιμοποιήσουμε τα σύμβολα = και := και για τη σύγκριση ισότητας τα σύμβολα = και ==. Ο Clipper χρησιμοποιεί τους γνωστούς μας λογικούς τελεστές .AND., .OR. και .NOT., αλλά με τελείες δίπλα απ' τα ονόματά τους.

Με την εντολή ? μπορούμε να τυπώσουμε στην οθόνη ή στον εκτυπωτή και να πάμε στην επόμενη γραμμή, ενώ με την εντολή ?? ο δρομέας παραμένει στο τέλος του κειμένου που τυπώθηκε.

Δείτε και το πρώτο πρόγραμμα σε Clipper :

```
/* Αυτό είναι το πρώτο πρόγραμμα σε Clipper για τους σπουδαστές του
IEK Φλώρινας - Φλώρινα 2/10/1997 */
CLEAR SCREEN            // η εντολή αυτή καθαρίζει την οθόνη
x := 1                  && εκχώρηση τιμής σε μεταβλητή
DO WHILE x < 5
        ? 'Το πρώτο πρόγραμμα σε Clipper'
        x := x + 1
ENDDO
RETURN
```

Αυτό το πρόγραμμα καθαρίζει την οθόνη με την εντολή CLEAR SCREEN και εκχωρεί την τιμή 1 στη μεταβλητή x. Η πρόταση DO WHILE ελέγχει αν η τιμή της μεταβλητής x είναι μικρότερη από 5 και αν ναι, τότε εκτελούνται οι εντολές μεταξύ των προτάσεων DO WHILE και ENDDO.

Η εντολή ? τυπώνει στην οθόνη το μήνυμα που είναι μέσα σε εισαγωγικά και μετακινεί το δρομέα στην επόμενη γραμμή. Η εντολή x:=x+1 αυξάνει την τιμή της x κατά 1. Όταν η τιμή της x γίνει ίση με 5, τότε η ροή του προγράμματος βγαίνει έξω από τον βρόχο και εκτελείται η εντολή RETURN.

Λίγα Λόγια για τις Βάσεις Δεδομένων

Μια ***Βάση Δεδομένων*** *(Database)* είναι μια οργανωμένη συγκέντρωση πληροφοριών που αναφέρονται σε μια συγκεκριμένη εφαρμογή. Για παράδειγμα, οι μαθητές ενός σχολείου, οι υπάλληλοι μιας εταιρείας, οι ασθενείς ενός νοσοκομείου, τα προϊόντα μιας επιχείρησης, οι πελάτες ενός γραφείου κ.ά. είναι παραδείγματα βάσεων δεδομένων.

Εγγραφή *(record)* είναι μια γραμμή της βάσης δεδομένων που αφορά π.χ. έναν συγκεκριμένο πελάτη μιας εταιρείας. Οι εγγραφές χωρίζονται στα ***πεδία*** *(fields)*. Τα πεδία αναφέρονται στα ιδιαίτερα στοιχεία του πελάτη, όπως είναι το επώνυμό του, η διεύθυνσή του, το τηλέφωνό του, το υπόλοιπό του κ.ά. Όλες οι εγγραφές μιας βάσης δεδομένων έχουν την ίδια δομή και το ίδιο μέγεθος και αλλάζουν μόνο τα στοιχεία των πελατών που περιέχουν.

Μπορούμε να έχουμε μια ***κύρια βάση δεδομένων*** που θα περιέχει τα βασικά στοιχεία των πελατών και μια ***βοηθητική (δευτερεύουσα) βάση δεδομένων*** που θα περιέχει τα στοιχεία των συναλλαγών τους. Ο συνδυασμός δύο ή και περισσότερων βάσεων δεδομένων μπορεί να γίνει με τη χρήση ενός πεδίου που θα είναι κοινό και στις δύο βάσεις δεδομένων.

΄Ενα τέτοιο πεδίο θα μπορούσε να είναι ο κωδικός αριθμός του πελάτη (Customer-ID). Αυτός ο τρόπος οργάνωσης είναι γνωστός σαν ***σχεσιακή βάση δεδομένων*** *(relational database)*. Τα δύο αρχεία θεωρούνται μαζί σαν μια βάση δεδομένων και πολύ εύκολα μπορούμε να πάρουμε ό,τι πληροφορίες θέλουμε είτε αφορούν έναν πελάτη ή τις παραγγελίες του.

Ο Clipper έχει πολύ μεγάλες δυνατότητες για την αποτελεσματική διαχείριση βάσεων δεδομένων, αφού υποστηρίζει τα ***αρχεία ευρετηρίου*** *(index files)*. Με τα αρχεία ευρετηρίου γίνεται αυτόματα η ταξινόμηση των εγγραφών που καταχωρούμε σε μια βάση δεδομένων με βάση όποιο πεδίο ή πεδία επιλέγουμε, ανεξάρτητα από τη σειρά με την οποία καταχωρούνται οι εγγραφές.

Πώς Γράφουμε και πώς Εκτελούμε ΄Ενα Πρόγραμμα στον Clipper

Για να γράψουμε ένα πρόγραμμα στον Clipper, χρησιμοποιούμε τον editor του DOS. Από την προτροπή C:\> του DOS, γράφουμε : ***edit clip01.prg*** και μετά πληκτρολογούμε τις εντολές του προγράμματος. Αποθηκεύουμε το πρόγραμμα και βγαίνουμε πάλι στην προτροπή του DOS.

Εκεί γράφουμε ***cl clip01*** για να γίνει η μεταγλώττιση και η σύνδεση του προγράμματος. Αν ο μεταγλωττιστής δεν βρει λάθη, τότε μπορούμε να τρέξουμε το πρόγραμμά μας, γράφοντας απλά το όνομά του, π.χ. ***clip01***, χωρίς την επέκταση .prg.

ΚΕΦΑΛΑΙΟ 2
Ο ΜΕΤΑΓΛΩΤΤΙΣΤΗΣ, Ο ΣΥΝΔΕΤΙΚΟΣ ΔΙΟΡΘΩΤΗΣ ΚΑΙ Ο ΠΡΟΕΠΕΞΕΡΓΑΣΤΗΣ ΤΟΥ CLIPPER

Ο Μεταγλωττιστής

Ο ***μεταγλωττιστής*** *(compiler)* είναι ένα πρόγραμμα (ρουτίνα) που επιτρέπει στον υπολογιστή να μεταφράζει ένα πρόγραμμα από μια γλώσσα ψευδοκώδικα σε γλώσσα μηχανής. Ακόμη, μπορεί να προεπεξεργάζεται ή να μεταφράζει τον ψευδοκώδικα σε άλλη γλώσσα ψευδοκώδικα για μετέπειτα μετάφραση και μεταγλώττιση.

Ο ***ψευδοκώδικας*** είναι ένα πρόγραμμα που πρέπει να μετατραπεί πριν χρησιμοποιηθεί από τον υπολογιστή. Ονομάζεται επίσης συμβολικός κώδικας και είναι ανεξάρτητος από το υλικό του υπολογιστή.

Οι μεταγλωττιστές είναι προτιμότεροι από τους *ερμηνευτές (interpreters)* γιατί μ' αυτούς εκτελούνται ταχύτερα οι εφαρμογές και δημιουργείται ένα ανεξάρτητο αρχείο με επέκταση .EXE, απαλλαγμένο από την ανάγκη της παρουσίας ενός ερμηνευτή για την εκτέλεση του προγράμματος.

Στην ουσία, ο μεταγλωττιστής μετατρέπει τον κώδικά μας, ο οποίος είναι γραμμένος σε μια γλώσσα που μοιάζει με τα αγγλικά (γλώσσα ανωτέρου επιπέδου), σε σύμβολα επιπέδου μηχανής που είναι γνωστά σαν γλωσσικά σημεία (tokens).

Οι Διακόπτες Μεταγλώττισης

Όταν δίνουμε την εντολή για να γίνει μεταγλώττιση ενός προγράμματος από τον Clipper, μπορούμε να δώσουμε και διάφορες παραμέτρους ή διακόπτες που ορίζουν κάποιες επιλογές που θέλουμε να κάνουμε. Οι διακόπτες μεταγλώττισης μπαίνουν στη γραμμή εντολής είτε με μια κάθετη (/) είτε με μια παύλα (-), μπορούν να χρησιμοποιηθούν με οποιαδήποτε σειρά και εμφανίζονται μετά το όνομα του αρχείου που πρόκειται να μεταγλωττιστεί.

Η βασική σύνταξη για να κάνουμε μεταγλώττιση σ' ένα αρχείο είναι η εξής :

Clipper <όνομα αρχείου> [<διακόπτες μεταγλώττισης>]

Ακολουθεί μια παρουσίαση των βασικότερων από τους διακόπτες μεταγλώττισης :

/**D** Ορισμός Σταθεράς
Σύνταξη : **/D<σταθερά>[=<ορισμός>]**
Ο διακόπτης αυτός δημιουργεί μια σταθερά (λέξη-κλειδί) η οποία ορίζεται ως κυριολεκτική σταθερά πριν τη διεργασία μεταγλώττισης και ορίζεται κατευθείαν από τη γραμμή εντολών. Όταν δεν ορίζεται <ορισμός> για τη <σταθερά>, η <σταθερά> θα έχει μηδενική τιμή αλλά θα θεωρείται ορισμένη.

Δείτε το παρακάτω παράδειγμα :

Clipper Myprog /DDEMO

Στο πρόγραμμα MYPROG.PRG μπορεί να υπάρχουν τα εξής :

```
#ifdef DEMO
        record_limit := 50
#else
        record_limit := 99999999999999999999
#endif
```

Στο παραπάνω παράδειγμα, αν το πρόγραμμα μεταγλωττίζεται με τον διακόπτη /DDEMO και ο προεπεξεργαστής ελέγχει αν το DEMO έχει καθοριστεί, η μεταβλητή record_limit θα έχει πάρει την τιμή 50, διαφορετικά θα πάρει μια πολύ μεγάλη τιμή.

/**P** Δημιουργία Λίστας Εξόδου του Προεπεξεργαστή
Σύνταξη : **/P[<όνομα αρχείου>]**
Παράγει μια λίστα εξόδου μετά το τέλος της ανίχνευσης του τμήματος ή του αρχείου προγράμματος από τον προεπεξεργαστή. Η προεπεξεργασμένη λίστα εξόδου είναι ένα ενδιάμεσο αρχείο του Clipper που δημιουργείται πριν από το πραγματικό βήμα μεταγλώττισης.

Εξ ορισμού, το αρχείο εξόδου θα έχει ίδιο όνομα με το αρχείο .PRG ή .CLP, αλλά με επέκταση *.PPO (Pre-Processor Output)*. Μπορούμε να ορίσουμε πάντως εμείς το όνομα του αρχείου εξόδου καθώς και τη διαδρομή του.

Δείτε τα παρακάτω παραδείγματα :

Clipper Gendata /P\bantam\c5\ppo
Clipper Gendata /P
Clipper Gendata /P\bantam\c5\ppo\output

Στο πρώτο παράδειγμα το αρχείο εξόδου είναι το gendata.ppo και δημιουργείται στον υποκατάλογο \bantam\c5\ppo, στο δεύτερο παράδειγμα το αρχείο είναι το gendata.ppo και θα δημιουργηθεί στον τρέχοντα υποκατάλο-

γο και στο τελευταίο παράδειγμα το αρχείο εξόδου θα ονομαστεί output.ppo και θα βρίσκεται στον υποκατάλογο \bantam\c5\ppo.

/**I** Ενσωμάτωση Λίστας Αναζήτησης Καταλόγου Αρχείων
Σύνταξη : **/I <διαδρομή αναζήτησης>**
Ο διακόπτης αυτός προσθέτει τον κατάλογο που καθορίζεται στη <διαδρομή αναζήτησης> στην αρχή της λίστας καταλόγων όπου θα αναζητηθούν τα συμπεριλαμβανόμενα αρχεία.

Το παρακάτω παράδειγμα δίνει εντολή στον Clipper να ψάξει στον κατάλογο \bantam\include για πιθανά αρχεία κεφαλίδων του Clipper.

Clipper Genadata /I\bantam\include

/**L** Ακύρωση Αριθμών Γραμμής
Σύνταξη : **/L**
Λέει στον μεταγλωττιστή να μη συμπεριλάβει στο μεταγλωττισμένο αντικειμενικό αρχείο αριθμούς γραμμής από τα πηγαία αρχεία προγράμματος. Αυτό γίνεται για οικονομία χώρου. Εξ ορισμού, οι αριθμοί γραμμής συμπεριλαμβάνονται.
Παράδειγμα :
Clipper Gendata /L

/**O** Ορισμός Ονόματος Αντικειμενικού Αρχείου
Σύνταξη : **/O<όνομα αρχείου>**
Ο διακόπτης αυτός παράγει ένα αρχείο .OBJ το οποίο θα ονομαστεί <όνομα αρχείου> με επέκταση .OBJ. Ο διακόπτης /O μπορεί να χρησιμοποιηθεί και για την τοποθέτηση του αρχείου .OBJ σε κάποιον καθορισμένο οδηγό και κατάλογο.

Δείτε τα παρακάτω παραδείγματα :

Clipper Gendata /L /O\bantam\object
Clipper Gendata /OData
Clipper Gendata /O\bantam\object\data

Στο πρώτο παράδειγμα το αρχείο προέλευσης gendata.prg θα μεταγλωττιστεί χωρίς αριθμούς γραμμής και θα καταλήξει σ' ένα μεμονωμένο αρχείο. Το αντικειμενικό αρχείο που θα δημιουργήσει ο Clipper θα ονομαστεί εξ ορισμού gendata.obj και θα σταλεί στον κατάλογο \bantam\object.

Στο δεύτερο παράδειγμα το αντικειμενικό αρχείο θα μετονομαστεί σε data.obj και θα τοποθετηθεί στον τρέχοντα κατάλογο. Τέλος, στο τρίτο παράδειγμα το αρχείο προέλευσης gendata.prg θα γραφεί στον κατάλογο \bantam\ object με το όνομα data.obj.

/**Q** Κρυφή Λειτουργία
Ο διακόπτης αυτός εμποδίζει τους αριθμούς γραμμής του πηγαίου κώδικα να εμφανιστούν στην οθόνη κατά τη λειτουργία του μεταγλωττιστή. Αυ-

τό επηρεάζει μόνο την οθόνη και όχι την εγγραφή των αριθμών γραμμής στο αρχείο .OBJ, που επηρεάζεται με την εντολή /L.

/S ΄Ελεγχος μόνο της Σύνταξης

Ο διακόπτης αυτός λέει στο μεταγλωττιστή να ελέγξει μόνο τη σύνταξη των καθορισμένων αρχείων προγράμματος και δεν παράγεται αντικειμενικό αρχείο.

/T Θέση Προσωρινών Αρχείων

Σύνταξη : **/T<διαδρομή>**

Ο διακόπτης αυτός καθορίζει μια διαφορετική διαδρομή καταλόγου για όλα τα προσωρινά αρχεία που μπορεί να δημιουργηθούν στη διάρκεια της μεταγλώττισης. Αν δεν οριστεί διαδρομή, θα χρησιμοποιηθεί ο τρέχων κατάλογος.

/U Τυποποιημένο Αρχείο Κεφαλίδων Ορισμένο από το Χρήστη

Σύνταξη : **/U<όνομα αρχείου>**

Ο διακόπτης αυτός οδηγεί το μεταγλωττιστή να χρησιμοποιήσει ένα εναλλακτικό αρχείο κεφαλίδων για τον προεπεξεργαστή. Αν δεν ορίζεται <όνομα αρχείου>, δε θα χρησιμοποιηθεί τυποποιημένο αρχείο κεφαλίδων για τον προεπεξεργαστή και ο πηγαίος κώδικας δεν θα μπει στη διαδικασία προεπεξεργασίας.

Δείτε τα παρακάτω παραδείγματα :

Clipper Gendata /U\bantam\c5\header\std.ch
Clipper Gendata /Umystd.ch
Clipper Gendata /U

Το πρώτο παράδειγμα λέει στο μεταγλωττιστή ότι το αρχείο κεφαλίδων του προεπεξεργαστή που θα χρησιμοποιηθεί είναι το std.ch και ότι αυτό βρίσκεται στον κατάλογο \bantam\c5\header.

Το δεύτερο παράδειγμα λέει στο μεταγλωττιστή ότι το αρχείο κεφαλίδων για τον προεπεξεργαστή δεν είναι το std.ch αλλά το mystd.ch. Μια και δεν ορίζεται διαδρομή, ο μεταγλωττιστής θα το αναζητήσει στον τρέχοντα κατάλογο και οδηγό, στους καταλόγους που ορίζονται με το διακόπτη /I, και, τελικά, στον κατάλογο που ορίζεται με τη μεταβλητή περιβάλλοντος INCLUDE του DOS.

Στο τρίτο παράδειγμα, ο διακόπτης λέει στον Clipper ότι δεν υπάρχει τυποποιημένο αρχείο κεφαλίδων για τον προεπεξεργαστή.

/W Προειδοποιητικά Μηνύματα για Ασαφείς Αναφορές Μεταβλητών

Ο διακόπτης αυτός παρουσιάζει στην οθόνη προειδοποιητικά μηνύματα για όλες τις αναφορές σε αδήλωτες και χωρίς ψευδώνυμο μεταβλητές, καθώς και σε εκτελέσιμες προτάσεις που προηγούνται της πρώτης πρότασης διαταγών PROCEDURE ή FUNCTION.

/**Z** Μη Βελτιστοποίηση Κώδικα

Κανονικά, ο Clipper βελτιστοποιεί τον κώδικα χρησιμοποιώντας μια τεχνική με την οποία η αξιολόγηση των παραστάσεων που περιέχουν λογικούς τελεστές (.AND. και .OR.) θα σταματήσει όταν θα μπορεί να καθοριστεί το αποτέλεσμα της παράστασης σαν σύνολο.

Αυτή η τεχνική ονομάζεται "*συντόμευση*" *(shortcutting)* και εμποδίζει την εκτέλεση όλων των τμημάτων μιας παράστασης, έτσι ώστε τα αποτελέσματα να μπορούν να προσδιοριστούν χωρίς να εκτελεστούν όλα τα τμήματα του κώδικα.

Επειδή αυτή η δυνατότητα υπάρχει από την έκδοση 5.01 του Clipper και μετά, με στόχο τη συμβατότητα με τις προηγούμενες εκδόσεις, αυτή η τεχνική συντόμευσης μπορεί να ακυρωθεί με την επιλογή μεταγλώττισης /Z.

Το Περιβάλλον του DOS

Υπάρχουν διάφορες μεταβλητές περιβάλλοντος του DOS που μπορούν να καθοριστούν για να βοηθήσουν στη διεργασία μεταγλώττισης. Οι μεταβλητές αυτές μπορούν να δηλωθούν απευθείας στη γραμμή εντολών ή να συμπεριληφθούν στο αρχείο AUTOEXEC.BAT και είναι οι εξής :

CLIPPERCMD

Η μεταβλητή αυτή ορίζεται ως εξής :

SET CLIPPER=<οδηγίες μεταγλωττιστή>

Η λίστα των <οδηγιών μεταγλωττιστή> καθορίζει τους διακόπτες μεταγλώττισης του Clipper που μπορεί να λαμβάνει υπόψη του ο μεταγλωττιστής κάθε φορά που καλείται. Οποιοσδήποτε διακόπτης μεταγλώττισης ορίζεται στη λίστα αυτών των οδηγιών θα ληφθεί υπόψη πριν απ' αυτούς που ορίζονται στη γραμμή εντολών.

Χρειάζεται προσοχή, γιατί σε ορισμένες περιπτώσεις, όταν ένας διακόπτης ορίζεται και στη στοίβα της εντολής SET και στη γραμμή εντολών, υπάρχει περίπτωση να απενεργοποιηθεί και να μην ισχύσει.

Παράδειγμα :

SET CLIPPERCMD=/M /P /V /U\bantam\b2\std.ch

Η απενεργοποίηση της μεταβλητής CLIPPERCMD γίνεται ως εξής :

SET CLIPPERCMD=

TMP

Η μεταβλητή αυτή ορίζεται ως εξής :

SET TMP=<διαδρομή>

Σε μερικές περιπτώσεις ο Clipper δημιουργεί προσωρινά αρχεία για να ολοκληρώσει το βήμα προεπεξεργασίας ή και το βήμα μεταγλώττισης. Αυτά τα προσωρινά αρχεία μπορεί να τοποθετηθούν σ' έναν άλλο κατάλογο του σκληρού δίσκου ή και στον δίσκο RAM.

Μόλις τελειώσει ο Clipper, τα αρχεία αυτά θα διαγραφούν αυτόματα. Ο κατάλογος, όμως, όπου θα γραφούν αυτά τα ειδικά αρχεία ορίζεται με την εντολή SET TMP. Η ίδια διαδρομή θα χρησιμοποιηθεί επίσης και από τον συνδετικό διορθωτή (linker) του Clipper.

Παράδειγμα :

SET TMP=H:

΄Ετσι, τα προσωρινά αρχεία θα δημιουργηθούν στον κατάλογο H:\.

Για να ακυρωθεί αυτή η εντολή περιβάλλοντος του DOS, δίνουμε :

SET TMP=

Ο διακόπτης μεταγλώττισης /T ακυρώνει αυτή την εντολή.

INCLUDE
Η μεταβλητή αυτή ορίζεται ως εξής :

SET INCLUDE=<διαδρομή>

Αυτή η μεταβλητή περιβάλλοντος του DOS σάς επιτρέπει να γνωστοποιείτε στον μεταγλωττιστή την προεπιλεγμένη θέση όλων των αρχείων INCLUDE.

Η διαδρομή μπορεί επίσης να οριστεί κατευθείαν με τις οδηγίες του προεπεξεργαστή :

#include "\bantam\values.ch"

Για να απενεργοποιηθεί δίνουμε :

SET INCLUDE=

CLIPPER
Αυτή η εντολή περιβάλλοντος χρησιμοποιείται για το χειρισμό διαφόρων προγραμμάτων, όπως του Διαχειριστή Εικονικής Μνήμης του Clipper, της ρύθμισης της διευρυμένης μνήμης για μια εξωτερική εντολή RUN, καθώς και των ρυθμίσεων του διαχειριστή των προγραμμάτων επικάλυψης.

Όλες αυτές οι ρυθμίσεις μπορούν να γίνουν μαζί, με την προϋπόθεση ότι χρησιμοποιείται ένα ελληνικό ερωτηματικό (;) για να ξεχωρίζει τη μια από την άλλη.

Ακολουθούν μερικά παραδείγματα :

SET CLIPPER=F:50
SET CLIPPER=E:1024
SET CLIPPER=NOIDLE

Ο Συνδετικός Διορθωτής

Ο ***συνδετικός διορθωτής*** *(linker)* παίρνει τις συμβολικές αναφορές ή τις ρουτίνες (που ονομάζονται διαδικασίες ή συναρτήσεις) ενός προγράμματος και βρίσκει τους ορισμούς τους σ' άλλα αντικειμενικά αρχεία (.OBJ) ή σ' ένα αρχείο βιβλιοθήκης. Η μεταγλώττιση αποτελεί μόνο το μισό της δουλειάς. Το δεύτερο μισό είναι η σύνδεση των κομματιών μιας εφαρμογής μεταξύ τους.

Ένα ***αρχείο βιβλιοθήκης*** *(.LIB)* είναι ένα αρχείο που περιλαμβάνει ένα ή περισσότερα αντικειμενικά αρχεία. Ο συνδετικός διορθωτής θα βρει την αναφορά σε μια καλούμενη υπορουτίνα και θα συνδέσει στην εφαρμογή τη λειτουργική μονάδα του αντικειμενικού αρχείου του αρχείου .LIB που περιέχει αυτή τη συγκεκριμένη διαδικασία ή συνάρτηση.

Οι ορισμοί όλων των συναρτήσεων του Clipper είναι συγκεντρωμένοι σε τέσσερα αρχεία βιβλιοθήκης. Οι βασικές συναρτήσεις του Clipper και η διάλεκτος εντολών της dBase βρίσκονται στο *CLIPPER.LIB*. Στο *EXTEND. LIB* βρίσκονται επεκτεταμένες συναρτήσεις και λειτουργίες καθώς και επιπρόσθετες συναρτήσεις (που ορίζονται από τον χρήστη).

Το *TERMINAL.LIB* περιέχει τις αντικειμενικές λειτουργικές μονάδες που αναφέρονται ειδικά στον έλεγχο της οθόνης, ενώ το *DBFNTX.LIB* είναι το αρχείο βιβλιοθήκης που περιέχει την αντικειμενική λειτουργική μονάδα που χειρίζεται τη μορφή αρχείου .NTX (αρχεία index). Οι τέσσερις αυτές βιβλιοθήκες είναι ενσωματωμένες στην κεφαλίδα οποιουδήποτε αντικειμενικού αρχείου που δημιουργείται από τον Clipper 5.01.

Οι Εντολές του RTLink

Στον Clipper υπάρχει ένας ειδικός συνδετικός διορθωτής που λέγεται ***RTLink*** και ο οποίος αποτελείται από τα παρακάτω πέντε αρχεία :

RTLINK.EXE

Είναι ο πραγματικός συνδετικός διορθωτής που δημιουργεί εκτελέσιμα αρχεία (.EXE) και προσυνδεδεμένες βιβλιοθήκες (.PLL και .PLT).

RTLINKST.COM

Περιέχει πληροφορίες έναρξης για τον RTLINK που μπορούν να χρησιμοποιηθούν κάθε φορά που δημιουργείται μια προσυνδεδεμένη βιβλιοθήκη. Το αρχείο αυτό θα πρέπει να είναι στον ίδιο κατάλογο με το RTLINK. EXE.

RTUTILS.LIB

Περιλαμβάνει το διαχειριστή του στατικού προγράμματος επικάλυψης.

RTLINK.DAT

Περιλαμβάνει όλα τα μηνύματα σφαλμάτων του συνδετικού διορθωτή. Το αρχείο αυτό πρέπει να βρίσκεται στον ίδιο κατάλογο με το RTLINK. EXE.

RTLINK.HLP

Περιλαμβάνει τις βοηθητικές πληροφορίες που απεικονίζονται όταν δίνεται η εντολή HELP. Το αρχείο αυτό πρέπει να βρίσκεται στον ίδιο κατάλογο με το RTLINK.EXE.

Οι Εντολές Περιβάλλοντος

Υπάρχουν διάφορες μεταβλητές περιβάλλοντος του DOS που βοηθούν τη διαδικασία σύνδεσης, βοηθούν, δηλαδή, τον συνδετικό διορθωτή να βρίσκει αρχεία, του παρέχουν μια προεπιλεγμένη σειρά εντολών ή του καθορίζουν τη διαδρομή των αρχείων εξόδου.

Οι σημαντικότερες απ' αυτές είναι :

SET RTLINKCMD=<παρΧ1>

Ορίζει τις προεπιλεγμένες οδηγίες της γραμμής εντολών για τον συνδετικό διορθωτή. Η <παρΧ1> είναι μια λίστα εντολών.

Π.χ. με την εντολή :

SET RTLINKCMD=/INCREMENTAL:30

η προεπιλεγμένη κατάσταση θα είναι να γίνει αυξητική σύνδεση με κατανομή χαμένου χώρου κατά 30%.

SET TMP=<παρΧ1>

Μπορούμε να κατευθύνουμε την έξοδο των προσωρινών αρχείων που δημιουργεί ο συνδετικός διορθωτής σ' έναν οδηγό και κατάλογο που θέλουμε εμείς, ενώ εξ ορισμού γράφονται στον τρέχοντα οδηγό και κατάλογο.

Π.χ. :

SET TMP=H:

SET LIB=<παρΧ1>

Όταν συνδέονται συνηθισμένα αντικειμενικά αρχεία του Clipper, τα προεπιλεγμένα ονόματα βιβλιοθήκης που ενσωματώνονται στις κεφαλίδες των αντικειμενικών αρχείων πληροφορούν τον συνδετικό διορθωτή για το ποιες βιβλιοθήκες χρειάζονται.

Τότε, ο συνδετικός διορθωτής ψάχνει αυτόματα στον τρέχοντα κατάλογο για τις βιβλιοθήκες. Αν τα αρχεία δεν είναι στον τρέχοντα κατάλογο, τότε θα ερευνηθεί η διαδρομή που ορίζεται στην <παρΧ1> της εντολής SET LIB.

Π.χ. :

SET LIB=\CLIPPER\50\LIB
SET LIB=\CLIPPER\50\LIB; \TOOLKIT\50

Στο πρώτο παράδειγμα ο συνδετικός διορθωτής ψάχνει για πιθανά αρχεία βιβλιοθήκης που δεν βρίσκονται στον τρέχοντα κατάλογο, αλλά στον κατάλογο \CLIPPER\50\LIB. Στο δεύτερο παράδειγμα φαίνεται πώς μπορούμε να ορίσουμε περισσότερους καταλόγους χωρισμένους με το ελληνικό ερωτηματικό (;).

SET PLL=<παρΧ1>
Είναι παρόμοια με τη μεταβλητή SET LIB, αλλά αναφέρεται μόνο στα προσυνδεδεμένα αρχεία βιβλιοθήκης. δηλ. στα αρχεία με επεκτάσεις .PLL και .PLT.

SET OBJ=<παρΧ1>
Σε μερικές περιπτώσεις, τα σύμβολα που χρειάζεται να εντοπιστούν για να αναλυθούν οι άγνωστες εξωτερικές αναφορές μπορεί να βρεθούν σ' άλλα αντικειμενικά αρχεία (.OBJ). Αν τα αρχεία αυτά βρίσκονται έξω από τον τρέχοντα κατάλογο, μπορεί να χρειαστεί η μεταβλητή SET OBJ.

Η Προεπεξεργασία

Ο Clipper 5.01 είναι και προεπεξεργαστής και μεταγλωττιστής. Στην ουσία, πριν αρχίσει την πραγματική μεταγλώττιση, ο Clipper σαρώνει πρώτα το πρόγραμμά μας και επεξεργάζεται ορισμένα στοιχεία. Έτσι, θα είναι πιο εύκολο να γίνουν κάποιες τυποποιήσεις, όχι μόνο για μια εφαρμογή, αλλά και για κάθε πρόγραμμα που είναι γραμμένο στον Clipper 5.01. Ο προεπεξεργαστής αποτελεί μέρος του μεταγλωττιστή.

Υπάρχουν διάφορα αρχεία με επέκταση .CH στις δισκέτες του Clipper και αυτά είναι τα τυποποιημένα αρχεία κεφαλίδων για τον προεπεξεργαστή. Το αρχείο STD.CH, συγκεκριμένα, είναι ένα ειδικό αρχείο που περιέχει όλους τους ορισμούς εντολών στη διάλεκτο του Clipper.

Θα πρέπει να έχετε υπόψη σας ότι η διεργασία μεταγλώττισης της εφαρμογής σας περιλαμβάνει πάντοτε και το βήμα προεπεξεργασίας. Αυτό το βήμα δεν αφήνει ίχνη μια και το αρχείο εξόδου διαγράφεται μόλις παραχθεί το αρχείο .OBJ. Για να αναγκάσουμε, όμως, τον Clipper να διατηρήσει το αρχείο εξόδου θα πρέπει να χρησιμοποιήσουμε τον διακόπτη μεταγλώττισης /E.

Η Σύνταξη των Εντολών του Προεπεξεργαστή

Τις εντολές του προεπεξεργαστή μπορούμε να τις τοποθετήσουμε κατευθείαν μέσα στον κώδικα του Clipper. Αυτές λέγονται οδηγίες προεπεξεργαστή και θα πρέπει στην αρχή τους να έχουν τον χαρακτήρα #. Οι εντολές αυτές είναι οι εξής :

#define
#undef
#ifdef, #else, #endif
#ifndef, #else, #endif
#include
#translate, #xtranslate
#command, #xcommand
#error

Κάθε εντολή μπορεί να τοποθετηθεί σε οποιοδήποτε σημείο ενός αρχείου που πρόκειται να διαβαστεί από τον προεπεξεργαστή, αλλά κανονικά θα πρέπει να τοποθετούνται στην αρχή του κύκλου μεταγλώττισης, εκτός από σπάνιες περιπτώσεις.

#define

Σύνταξη #define <pΣΤΑΘΕΡΑ> [<τιμή>]
#define <ΣΥΝΑΡΤΗΣΗ> ([<λίστα ορισμάτων>])>
[<παράσταση>]

Θα πρέπει να έχετε υπόψη σας ότι τα κεφαλαία και τα πεζά γράμματα είναι διαφορετικά. Αυτή η εντολή προεπεξεργαστή μάς επιτρέπει να καθορίσουμε τις λέξεις-κλειδιά, που είναι γνωστές σαν *δηλωτικές σταθερές*, ώστε να έχουν συγκεκριμένες σημασίες ή να καθορίσουμε μια *ψευδοσυνάρτηση*.

Δείτε τα παρακάτω παραδείγματα :

```
#define pNORMAL "1/7, 7/1"
#define pBLUE "1"
#define pWHITE "7"
#define pESC 27
...
SETCOLOR (pNORMAL)
SETCOLOR (pBLUE+"/"+pWHITE)
...
if LASTKEY() = pESC
...
```

Είναι καλό να τοποθετούμε ένα πεζό p πριν από κάθε δηλωτική σταθερά που πρόκειται να επεξεργαστεί ο προεπεξεργαστής και αυτό γιατί μόλις συναντήσει μια εντολή #define, αντικαθιστά μέσα στο πρόγραμμα τη λέξη που ακολουθεί την εντολή #define με ό,τι ισοδύναμο έχουμε ορίσει και έτσι είναι πολύ πιθανό να συμβούν ολέθρια λάθη.

Στην ουσία, η #define είναι μια εντολή αντικατάστασης αλφαριθμητικού για τον προεπεξεργαστή. Λέμε στον Clipper να αντικαταστήσει όλες τις λέξεις που βρίσκονται αριστερά, με τα αλφαριθμητικά που βρίσκονται δεξιά. Το κενό μεταξύ των δύο είναι ο οριοθέτης (delimiter).

Θα πρέπει επίσης να έχετε υπόψη σας ότι οι εντολές #define ισχύουν μόνο για το αρχείο στο οποίο βρίσκονται και δεν επιδρούν σε κανένα άλλο αρχείο. Ωστόσο, αν κάποια δηλωτική σταθερά ορίζεται σ' ένα συμπεριλαμβανόμενο αρχείο, αυτό το αρχείο θα πρέπει να τοποθετηθεί στο αρχείο πηγαίου προγράμματος με την εντολή #include.

Μπορείτε επίσης να δημιουργήσετε μια δηλωτική σταθερά κατά τη μεταγλώττιση με τον διακόπτη /D. Η εντολή #define σάς επιτρέπει να οργανώσετε τις δικές σας λέξεις-κλειδιά έτσι ώστε να αντιπροσωπεύουν ακόμα και ολόκληρες φράσεις οι οποίες επαναλαμβάνονται σ' όλη την εφαρμογή.

Δείτε και μια χρήση της #define για τον ορισμό ψευδοσυνάρτησης :

#define pColor(exp1, epx2) epx1+"/"+epx2

#undef

Σύνταξη #undef [pΣΤΑΘΕΡΑ]

Αυτή η εντολή μάς επιτρέπει να ακυρώσουμε τον ορισμό δηλωτικών σταθερών ή ψευδοσυναρτήσεων οι οποίες είχαν καθοριστεί προηγουμένως με μια εντολή #define.

Δείτε το παρακάτω παράδειγμα :

```
#define HARDCR CHR(13)+CHR(10)
...
#undef HARDCR
HARDCR := "Αυτό είναι μια δοκιμή"
...
```

#ifdef/#else/#endif και #ifndef/#else/#endif

Σύνταξη
```
#ifdef <pΣΤΑΘΕΡΑ>
        <εντολές>
[#else]
        <εντολές>
[#endif]
```

ή

```
#ifndef <pΣΤΑΘΕΡΑ>
        <εντολές>
[#else]
        <εντολές>
[#endif]
```

Αυτές οι εντολές μάς επιτρέπουν να δημιουργούμε εφαρμογές μεταγλώττισης υπό συνθήκη. Αν η δηλωτική σταθερά που αναφέρεται στη συνθήκη #ifdef είχε οριστεί προηγουμένως, τότε θα μεταγλωττιστούν οι <εντολές> που ακολουθούν.

Αν όχι, αυτές οι εντολές δεν θα μεταγλωττιστούν, και αν υπάρχει αντίστοιχη εντολή #else, θα μεταγλωττιστούν οι <εντολές> που ακολουθούν την #else.

Αν, όμως, η δηλωτική σταθερά δεν καθορίζεται με μια πρόταση #define και χρησιμοποιείται η συνθήκη #ifndef, τότε θα μεταγλωττιστούν οι <εντολές> που ακολουθούν την #ifndef. Αν, όμως, η δηλωτική σταθερά έχει οριστεί, τότε θα ισχύσει η συνθήκη #else και θα μεταγλωττιστούν οι αντίστοιχες <εντολές>.

Ανάλογα με τη φύση των αναγκών προγραμματισμού που έχουμε, μπορεί να θέλουμε να μεταγλωττίσουμε κάτι υπό τον όρο ότι έχει οριστεί ή δεν έχει οριστεί μια δηλωτική σταθερά. Οι δηλωτικές σταθερές μπορούν να οριστούν με τις εντολές #define και #undef ή με τον διακόπτη μεταγλώττισης /D.

Θα πρέπει να έχετε υπόψη σας ότι όλες οι εντολές του προεπεξεργαστή εξετάζονται πριν από τη μεταγλώττιση του κώδικα και ουσιαστικά προσθέτουν κώδικα σε μια εφαρμογή πριν ο κώδικας προέλευσης μετατραπεί σε αντικειμενικό αρχείο.

Δείτε το παρακάτω παράδειγμα :

```
#define pDEMO
CLS
#ifdef pdemo
        ? "Αυτή η γραμμή κώδικα θα προστεθεί"
        ? "επειδή αυτό είναι πρόγραμμα επίδειξης"
#else
        ? "Όχι, αυτή η ενότητα κώδικα θα"
        ? "προστεθεί στο πρόγραμμα αντί"
        ? "για τις άλλες δύο γραμμές..."
#endif
```

Στο παραπάνω παράδειγμα η #define έχει ορίσει το σύμβολο pDEMO που είναι διαφορετικό από το pdemo και έτσι εκτελούνται οι εντολές που ακολουθούν την #else.

#include

Σύνταξη #include "<όνομα αρχείου>"

Με μια εντολή #include μπορείτε να διατάξετε τον προεπεξεργαστή να εισαγάγει τα περιεχόμενα ενός άλλου αρχείου στη συγκεκριμένη θέση μέσα στο πρόγραμμά σας. Τα αρχεία που συμπεριλαμβάνονται μπορεί να

περιέχουν τμήματα κώδικα και εντολές προεπεξεργαστή όπως οι #define, #undef και #ifdef/#ifndef.

Δείτε το παρακάτω παράδειγμα :

#include "Newdefs.inc"

Το αρχείο Newdefs.inc, που θα συμπεριληφθεί στο πρόγραμμα που περιέχει την παραπάνω εντολή, μπορεί να περιέχει τις εξής εντολές :

#define pESC 27
#define pABORT (LASTKEY()=pESC)
#define pPAUSE INKEY(0)

Για να διευκολύνετε την εργασία σας μπορείτε, να ορίσετε ένα συμπεριλαμβανόμενο αρχείο για όλους τους ορισμούς χρώματος, ένα άλλο για τις τιμές των πλήκτρων και ένα τρίτο για τις κοινόχρηστες μεταβλητές.

Θα μπορούσε ακόμη να δημιουργηθεί ένα άλλο συμπεριλαμβανόμενο αρχείο που θα καλούσε, στη συνέχεια, τα άλλα συμπεριλαμβανόμενα αρχεία :

#include "Publics.inc"
#include "Keys.ch"
#include "Color.ch"

#translate, #xtranslate, #command και #xcommand

Σύνταξη

#translate <μορφή προέλευσης> => <μορφή εξόδου>
#command <μορφή προέλευσης> => <μορφή εξόδου>
#xtranslate <μορφή προέλευσης> => <μορφή εξόδου>
#xcommand <μορφή προέλευσης> => <μορφή εξόδου>

Οι εντολές αυτές επιτρέπουν να αλλάξει η σύνταξη μιας συνάρτησης πριν ξεκινήσει η μεταγλώττιση. Επίσης, με τη βοήθειά τους μπορούμε να μεταβάλλουμε τις συναρτήσεις του Clipper ώστε να μοιάζουν με τις συναρτήσεις από άλλες γλώσσες προγραμματισμού, όπως π.χ. της C και της Pascal.

Πρέπει να έχετε υπόψη σας ότι η εντολή #translate λειτουργεί in-line στον κώδικα προέλευσης, ενώ η εντολή #command μετασχηματίζει μια ολόκληρη γραμμή εντολών. Και οι δύο αυτές εντολές ακολουθούν τους ίδιους βασικούς κανόνες για τις οδηγίες μετάφρασης : Διαβάζουν και επιχειρούν να εντοπίσουν τη <μορφή προέλευσης> και παράγουν τη <μορφή εξόδου>.

Η διαφορά μεταξύ των εντολών #translate και #xtranslate, καθώς και μεταξύ των εντολών #command και #xcommand, είναι ότι η παραδοσιακή ικανότητα για σύντμηση μιας εντολής στα τέσσερα πρώτα γράμματά της, όπως συμβαίνει στην dBase, υποστηρίζεται από τις #translate και #command, ενώ οι κανόνες σύντμησης των τεσσάρων γραμμάτων δεν ισχύουν για καμία

από τις εντολές του τύπου #x. Στην ουσία, οι εντολές αυτές απαιτούν ακριβή ταιριάσματα για όλες τις λέξεις-κλειδιά του κειμένου εισόδου.

Η συνάρτηση που παίρνει πληροφορίες για το περιβάλλον είναι η εξής:

? GETENV()

Σ' άλλες μορφές της διαλέκτου dBase, η παραπάνω συνάρτηση είναι ισοδύναμη με την :

? GETENVE()

Με τη χρήση της εντολής #translate μπορούμε απλώς να αλλάξουμε το συντακτικό της γλώσσας χωρίς να αλλάξουμε όλες τις εντολές κωδικοποίησης σ' όλο το πρόγραμμα :

#translate GETENVE() => GETENV()

Ο προεπεξεργαστής του Clipper θα αφαιρέσει τη συνάρτηση GETENVE() και θα την αντικαταστήσει με την GETENV().

Μπορούμε ακόμα να δημιουργήσουμε νέες συναρτήσεις συνδυάζοντας ήδη υπάρχουσες ή και απλές εντολές του Clipper. Δείτε το παρακάτω παράδειγμα που έχει αρκετό ενδιαφέρον :

```
#translate Isdigit (<χαρακτήρας>) =>
        ((<χαρακτήρας>) >= "0" .and. (<χαρακτήρας>) <= "9")
```

Η συνάρτηση Isdigit() επιστρέφει true όταν το πρώτο byte του <χαρακτήρα> είναι μεταξύ 0 και 9, δηλ. είναι ψηφίο, και επιστρέφει false σε κάθε άλλη περίπτωση. Στην ουσία, με την παραπάνω εντολή λέμε στον προεπεξεργαστή να αντικαταστήσει όπου βρει τη συνάρτηση Isdigit με ό,τι εντολές την ακολουθούν.

Δείτε και ένα παράδειγμα με την εντολή #command :

```
#command TITLE <*λέξεις*> =>;
        CLS;;
        @ 0, 0 SAY #<λέξεις>
```

#error

Σύνταξη #error [<κείμενο για έξοδο στην οθόνη>]

Αυτή η εντολή προεπεξεργαστή θα έχει σαν αποτέλεσμα ένα σφάλμα μεταγλώττισης και θα παρουσιάσει στην οθόνη το προαιρετικό μήνυμα.

Δείτε το παρακάτω παράδειγμα :

```
#define pDEMO
...
#ifndef pDEMO
#error Δεν έχει ακόμη καθοριστεί η κατάσταση επίδειξης
#endif
```

Στο παραπάνω παράδειγμα, αν δεν οριστεί η δηλωτική σταθερά pDEMO, είτε με την εντολή #define είτε με τη χρήση του διακόπτη μεταγλώττισης /D, η εργασία μεταγλώττισης θα διακοπεί και θα βγει στην οθόνη το αντίστοιχο μήνυμα.