ΚΕΦΑΛΑΙΟ 7
ΜΙΑ ΟΛΟΚΛΗΡΩΜΕΝΗ ΕΦΑΡΜΟΓΗ ΣΤΟΝ CLIPPER

Όταν δημιουργούμε μια εφαρμογή στον Clipper, πρέπει πάντα να γράφουμε υπορουτίνες (διαδικασίες ή συναρτήσεις) για τα κομμάτια εκείνα του προγράμματος που εκτελούν μια συγκεκριμένη εργασία, όσο μικρή και ασήμαντη κι αν μας φαίνεται αυτή.

Στις πρώτες γραμμές του κυρίου προγράμματος θα πρέπει να ορίζουμε τις καθολικές μεταβλητές, να ανοίγουμε τις βάσεις δεδομένων που θα χρησιμοποιηθούν σ' όλη την εφαρμογή και να προετοιμάζουμε το περιβάλλον της εφαρμογής.

Οι συναρτήσεις σχεδιασμού της οθόνης και οι υπορουτίνες αναζήτησης, εμφάνισης, καταχώρησης και διαγραφής δεδομένων θα πρέπει να αποτελούν ξεχωριστά τμήματα. Θα δούμε εδώ μια εφαρμογή για ένα σύστημα παρακολούθησης πελατών.

Το κύριο πρόγραμμα της εφαρμογής δεν χρειάζεται να δηλωθεί σαν υπορουτίνα. Το όνομά του είναι το ίδιο με το όνομα του αρχείου του DOS, π.χ. PELATES.PRG ή CUST.PRG, όπως το έχουμε ονομάσει εδώ.

Εκείνο που θα πρέπει να θυμόμαστε είναι ότι το κύριο μενού θα παρουσιάζεται επανειλημμένα στον χρήστη με μια εντολή DO WHILE .T. ... ENDDO η οποία θα περιέχει τις κατάλληλες εντολές για τη δημιουργία του μενού (PROMPT και MENU TO) και θα βγαίνουμε απ' αυτόν τον ατέρμονα βρόχο όταν πατήσουμε το πλήκτρο ESC.

Αν πατήσουμε το πλήκτρο <enter> ενώ βρισκόμαστε σε κάποια επιλογή του μενού, το πρόγραμμα θα καλεί την αντίστοιχη υπορουτίνα η οποία μπορεί να ανοίγει κι άλλα μενού (υπομενού). Οι υπορουτίνες ενός προγράμματος μπορούν να γραφούν αμέσως μετά το κύριο πρόγραμμα ή να αποτελούν ξεχωριστά αρχεία με επέκταση .PRG.

Αν οι υπορουτίνες γραφούν μετά το κύριο πρόγραμμα, δηλ. στο ίδιο αρχείο .PRG, είναι καλό να τις τερματίζουμε πάντα με την εντολή RETURN έστω και αν δεν είναι συναρτήσεις, για να ξέρουμε πού ακριβώς τελειώνει η κάθε υπορουτίνα. Αν μια συνάρτηση δεν επιστρέφει τιμή, τότε μπορούμε να χρησιμοποιήσουμε την έκφραση RETURN .T. σαν τελευταία εντολή της.

Ακολουθεί η παρουσίαση του κυρίου προγράμματος και των υπορουτινών του.

```
/* κύριο πρόγραμμα - CUST.PRG */
LOCAL l_choice := 0
PRIVATE m_id     // η m_id είναι ορατή σ' όλες τις υπορουτίνες

USE Customer NEW              // ανοίγει τη βάση δεδομένων Customer.dbf
IF .NOT. FILE("Customer.ntx") // ελέγχει αν υπάρχει το Customer.ntx
        INDEX ON Id TO Customer   // δημιουργεί το ευρετήριο
ELSE
        SET INDEX TO Customer     // ανοίγει το ευρετήριο
ENDIF

USE Trans NEW                 // ανοίγει τη βάση δεδομένων Trans.dbf
IF .NOT. FILE("Transid.ntx")  // ελέγχει αν υπάρχει το Transid.ntx
        INDEX ON Id TO Transid    // δημιουργεί το ευρετήριο
ELSE
        SET INDEX TO Transid      // ανοίγει το ευρετήριο
ENDIF

DO WHILE .T.              // μπαίνουμε σ' έναν ατέρμονα βρόχο
        CLEAR
        @ 00, 65 SAY DATE()
        @ 00, 35 SAY "ΠΑΡΑΚΟΛΟΥΘΗΣΗ ΠΕΛΑΤΩΝ"
        @ 01, 01 PROMPT "ΠΕΛΑΤΕΣ"
        @ 01, 30 PROMPT "ΑΝΑΦΟΡΕΣ"
        @ 01, 61 PROMPT "ΣΥΝΤΗΡΗΣΗ"
        MENU TO l_choice

        DO CASE
                CASE l_choice == 0      // βγαίνουμε με το ESC
                        RETURN          // και η εντολή QUIT είναι σωστή
                CASE l_choice == 1      // επιλέξαμε ΠΕΛΑΤΕΣ
                        entry()         // υπορουτίνα πελατών
                CASE l_choice == 2      // επιλέξαμε ΑΝΑΦΟΡΕΣ
                        rpt()           // υπορουτίνα αναφορών
                CASE l_choice == 3      // επιλέξαμε ΣΥΝΤΗΡΣΗ
                        maint()         // υπορουτίνα συντήρησης
        ENDCASE
ENDDO
RETURN
/* το κυρίως πρόγραμμα ανοίγει τις δύο βάσεις δεδομένων που θα χρησιμο-
ποιήσουμε (Customer και Transid) και δημιουργεί ή ανοίγει ανάλογα και τα
ευρετήριά τους - εμφανίζει ένα οριζόντιο μενού και μπορούμε να επιλέξουμε
να πατήσουμε το πλήκτρο ESC για να επιστρέψουμε στο DOS ή να διαλέξου-
με μία από τις τρεις επιλογές, όπου η καθεμία ανοίγει ένα δικό της κατακό-
ρυφο υπομενού */
```

```
/* υποβολή ερωτήσεων στον χρήστη */
FUNCTION yesno(l_question)
        LOCAL l_ans := .T.
        @ 24, 0                 // καθάρισμα της γραμμής 24
        @ 24, 0 SAY l_question GET l_ans PICTURE "Y"
        READ
        @ 24, 0
RETURN l_ans
/* η συνάρτηση αυτή δέχεται μία απάντηση μόνο Y ή N στη γραμμή 24 και επιστρέφει το πλήκτρο που πατάμε - η παράμετρος που δέχεται, η l_question, είναι το μήνυμα που εμφανίζεται στη γραμμή 24 */


/* εμφάνιση σχολίων */
FUNCTION comment(l_comment)
        @ 24, 0
        @ 24, 0 SAY l_comment
        INKEY(0)            // περιμένει μέχρι να πατήσουμε ένα πλήκτρο
        @ 24, 0
RETURN .T.
/* εμφανίζει κάποιο μήνυμα στον χρήστη και το πρόγραμμα περιμένει να πατήσουμε ένα πλήκτρο για να συνεχίσει - το μήνυμα υπάρχει στην παράμετρο l_comment */


/* αναζήτηση μιας εγγραφής */
FUNCTION search()
        LOCAL l_scrn1 := SAVESCREEN(10, 10, 12, 50)
        m_id := SPACE(11)
        @ 11, 11 SAY "Κωδικός Πελάτη : " GET m_id
        READ
        RESTSCREEN(10, 10, 12, 50, l_scrn1)
        SELECT Customer
        SEEK m_id
RETURN FOUND()
/* η συνάρτηση αυτή ζητά από τον χρήστη να δώσει έναν κωδικό πελάτη (m_id) και χρησιμοποιεί τη συνάρτηση SEEK για να βρει αν ο κωδικός αυτός υπάρχει στο αρχείο ευρετηρίου της βάσης δεδομένων Customer - το αποτέλεσμα της αναζήτησης επιστρέφεται στη συνάρτηση FOUND() και είναι και η τιμή επιστροφής της συνάρτησης */
```

```
/*  το υπομενού των πελατών */
FUNCTION  entry()
        LOCAL  l_holdscrn
        LOCAL  l_choice := 0
        SET DELETED ON
        SAVE  SCREEN  TO  l_holdscrn          // αποθηκεύει την οθόνη
        DO  WHILE  .T.
                RESTORE  SCREEN  FROM  l_holdscrn
                @ 3, 1 PROMPT "ΔΗΜΙΟΥΡΓΙΑ"        // εμφανίζει ένα
                @ 4, 1 PROMPT "ΤΡΟΠΟΠΟΙΗΣΗ"     // κατακόρυφο
                @ 5, 1 PROMPT "ΔΙΑΓΡΑΦΗ"          // υπομενού
                MENU  TO  l_choice
                IF  l_choice == 0             // πατήσαμε το ESC και γυρνάμε
                        RETURN .T.            // πίσω στο κύριο πρόγραμμα
                ENDIF

                // η παρακάτω εντολή IF καλεί τη συνάρτηση search() για να
                // ελέγξει αν υπάρχει ο κωδικός πελάτη m_id
                IF  search()              // ο κωδικός πελάτη m_id υπάρχει
                        DO  CASE
                                CASE  l_choice == 1
                                 IF  yesno(m_id+"υπάρχει ήδη-διόρθωση Y/N ? ")
                                        modifycust() // υπορουτίνα διόρθωσης
                                 ENDIF
                                CASE l_choice == 2
                                        modifycust() // υπορουτίνα διόρθωσης
                                CASE l_choice == 3
                                        deletecust()  // υπορουτίνα διαγραφής
                        ENDCASE
                ELSE                      // ο κωδικός πελάτη m_id δεν υπάρχει
                        DO CASE
                                CASE l_choice == 1
                                        createcust()  // υπορουτίνα δημιουργίας
                                CASE l_choice == 2
                                IF yesno(m_id+"δεν υπάρχει – προσθήκη Y/N?")
                                        createcust()  // υπορουτίνα δημιουργίας
                                CASE l_choice == 3
                                        comment(m_id + "δεν υπάρχει!")
                        ENDCASE
                ENDIF
        ENDDO
RETURN .T.
/* η υπορουτίνα των πελατών εμφανίζει ένα υπομενού, αλλά πρώτα δίνουμε
με τη συνάρτηση search() τον κωδικό του πελάτη (m_id) που ψάχνουμε και
ανάλογα αν ο κωδικός υπάρχει η όχι εκτελείται η αντίστοιχη εντολή DO
CASE - αν ο πελάτης υπάρχει μπορούμε να διορθώσουμε τα στοιχεία του ή να
τον διαγράψουμε ενώ αν δεν υπάρχει, μπορούμε μόνο να τον καταχωρίσουμε
*/
```

```
/* δημιουργία νέας εγγραφής */
FUNCTION  createcust()
        appendrec()          // προσθέτει μία κενή εγγραφή
        REPLACE  Id  WITH  m_id   // ο κωδικός πελάτη δίνεται στο πεδίο Id
        custscrn()           // εμφανίζει τους τίτλους των πεδίων
        saycust()            // εμφανίζει τις τιμές των πεδίων, αν υπάρχουν
        getcust()            // καταχωρεί τις τιμές στα πεδία
RETURN .T.
/* βλέπουμε ότι για τη δημιουργία μιας νέας εγγραφής πρέπει να καλέσουμε
πολλές συναρτήσεις - μία για τη δημιουργία μιας κενής εγγραφής - μία για τη
εμφάνιση των τίτλων των πεδίων - μία για να εμφανίζει τις τιμές των πεδίων
με την εντολή SAY, που εδώ βέβαια είναι κενές, και μία για να καταχωρούμε
τιμές στα πεδία με τις εντολές GET και READ */
```

```
/* τροποποίηση υπάρχουσας εγγραφής */
FUNCTION  modifycust()
        custscrn()           // εμφανίζει τους τίτλους των πεδίων
        saycust()            // εμφανίζει τις τιμές των πεδίων
        getcust()            // διορθώνει τις τιμές των πεδίων
RETURN .T.
/* για να τροποποιηθεί μια εγγραφή, θα πρέπει πρώτα να παρουσιάζονται οι
τιμές των πεδίων στην οθόνη και μετά να γίνουν οι διορθώσεις */
```

```
/* διαγραφή μιας εγγραφής */
FUNCTION  deletecust()
        REPLACE Fname WITH " ", Lname WITH " ", Address WITH " ",;
                TK WITH " ", City WITH " ", Phone WITH " ", Id WITH " "
        DELETE               // σημειώνει για διαγραφή την τρέχουσα εγγραφή
RETURN .T.
/* μηδενίζει τις τιμές των πεδίων της τρέχουσας εγγραφής και την σημειώνει
για διαγραφή με την εντολή DELETE χωρίς παραμέτρους */
```

```
/* ανάκτηση μιας εγγραφής */
FUNCTION  appendrec()
        SET  DELETED  OFF
        GO  TOP
        IF .NOT. EMPTY(Id) .OR. EOF()
                APPEND  BLANK
        ELSE
                RECALL
        ENDIF
        SET  DELETED  ON
RETURN .T.
```

/ επειδή συνήθως οι εγγραφές που σημειώνονται για διαγραφή τοποθετούνται στην αρχή του αρχείου ευρετηρίου καθώς ο κωδικός πελάτη Id των εγγραφών αυτών έχει πάρει μηδενική τιμή, με τη ρύθμιση SET DELETED OFF κάνουμε αυτές τις εγγραφές να είναι ορατές και πάμε στην αρχή του αρχείου ευρετηρίου με την εντολή GO TOP - το πεδίο Id ελέγχεται με τη συνάρτηση EMPTY() για να διαπιστωθεί αν είναι κενό και αν δεν είναι κενό ή αν είμαστε στο τέλος του αρχείου ευρετηρίου (EOF()==.T.), τότε προστίθεται μία νέα κενή εγγραφή στη βάση δεδομένων με την εντολή APPEND BLANK - διαφορετικά αν το πεδίο Id είναι κενό, τότε η εγγραφή που σημειώθηκε για διαγραφή αναιρείται με την εντολή RECALL και αυτή είναι πλέον η ενεργή ή τρέχουσα εγγραφή όπου θα καταχωρηθούν τα νέα δεδομένα */*

```
/* η οθόνη εισαγωγής δεδομένων */
FUNCTION  custscrn()
        @ 00, 24 SAY  "ΟΘΟΝΗ ΕΙΣΑΓΩΓΗΣ ΠΕΛΑΤΩΝ "
        @ 01, 02 SAY  "Κωδικός Πελάτη : "
        @ 02, 02 SAY  "Επώνυμο         : "
        @ 03, 02 SAY  "Όνομα           : "
        @ 04, 02 SAY  "Διεύθυνση       : "
        @ 05, 02 SAY  "ΤΚ              : "
        @ 06, 02 SAY  "Πόλη            : "
        @ 07, 02 SAY  "Τηλέφωνο        : "
RETURN .T.
/* αυτή η συνάρτηση παρουσιάζει απλά τους τίτλους για τα πεδία στα οποία πρόκειται να εισαχθούν δεδομένα από τον χρήστη */
```

```
/* παρουσίαση των δεδομένων μιας εγγραφής */
FUNCTION  saycust()
        @ 01, 18 SAY  Id
        @ 02, 18 SAY  Lname
        @ 03, 18 SAY  Fname
        @ 04, 18 SAY  Address
        @ 05, 18 SAY  TK
        @ 06, 18 SAY  City
        @ 07, 18 SAY  Phone
RETURN .T.
/* αυτή η συνάρτηση εμφανίζει απλά τα δεδομένα των πεδίων μιας εγγραφής χωρίς να μπορούμε να τα διορθώσουμε */
```

```
/* εισαγωγή δεδομένων από τον χρήστη */
FUNCTION  getcust()
        @ 02, 18  GET  Lname
        @ 03, 18  GET  Fname
        @ 04, 18  GET  Address
        @ 05, 18  GET  TK
        @ 06, 18  GET  City
        @ 07, 18  GET  Phone
        READ
RETURN .T.
/* αυτή η συνάρτηση δέχεται δεδομένα από την οθόνη και τα αποθηκεύει κα-
τευθείαν στα πεδία της εγγραφής - θα μπορούσαμε να αποθηκεύσουμε τις τι-
μές των δεδομένων πρώτα σε κάποιες βοηθητικές μεταβλητές και μετά με την
εντολή REPLACE να δώσουμε τιμές στα πεδία της εγγραφής */



/* υπομενού αναφορών */
PROCEDURE  rpt
        LOCAL  choice := 0
        DO  WHILE  .T.
                @ 3, 31 PROMPT "ΚΑΤΑΛΟΓΟΣ ΠΕΛΑΤΩΝ"
                @ 4, 31 PROMPT "ΣΥΝΑΛΛΑΓΕΣ ΑΝΑ ΜΗΝΑ"
                @ 5, 31 PROMPT "ΣΥΝΑΛΛΑΓΕΣ ΑΝΑ ΠΕΛΑΤΗ"
                @ 6, 31 PROMPT "ΕΚΚΡΕΜΕΙΣ ΛΟΓΑΡΙΑΣΜΟΙ"
                @ 7, 31 PROMPT "ΕΤΙΚΕΤΕΣ ΑΛΛΗΛΟΓΡΑΦΙΑΣ"
                MENU  TO  choice
                DO  CASE
                        CASE  choice == 0
                                RETURN     // επιστροφή στο κύριο πρόγραμμα
                        CASE  choice == 1
                                // καλεί την υπορουτίνα καταλόγου πελατών
                        CASE  choice == 2
                                // καλεί την υπορουτίνα των μηνιαίων συναλλαγών
                        CASE  choice == 3
                                // καλεί την υπορουτίνα των συναλλαγών πελατών
                        CASE  choice == 4
                                // καλεί την υπορουτίνα των εκκρεμ. λογαριασμών
                        CASE  choice == 5
                                // καλεί την υπορουτίνα των ετικετ.αλληλογραφίας
                ENDCASE
        ENDDO
RETURN
```

```
/* υπομενού συντήρησης */
PROCEDURE maint
        LOCAL choice := 0
        DO WHILE .T.
                @ 3, 62 PROMPT "ΕΦΕΔΡΙΚΑ ΑΝΤΙΓΡΑΦΑ"
                @ 4, 62 PROMPT "ΑΠΟΚΑΤΑΣΤΑΣΗ ΔΕΔΟΜΕΝΩΝ"
                @ 5, 62 PROMPT "ΑΝΑΔΗΜΙΟΥΡΓΙΑ ΕΥΡΕΤΗΡΙΩΝ"
                MENU TO choice
                DO CASE
                        CASE choice == 0
                                RETURN    // επιστροφή στο κύριο πρόγραμμα
                        CASE choice == 1
                                // καλεί την υπορουτίνα εφεδρικών αντιγράφων
                        CASE choice == 2
                                // καλεί την υπορουτίνα αποκατάστ. δεδομένων
                        CASE choice == 3
                                // καλεί την υπορουτίνα αναδημιουργ. ευρετηρίων
                ENDCASE
        ENDDO
RETURN
```