χνητά επιβαλόμενη κατάσταση καθώς όλα τα αποθηκευτικά μέσα επιτρέπουν την *τυχαία* προσπέλαση.

10.2 Καταχώρηση δεδομένων σε σειριακά αρχεία

Για τη μελέτη της σειριακής επεξεργασίας αρχείων θα δημιουργήσουμε ένα πακέτο προγραμμάτων για την καταγραφή και την (αναζήτηση και) άντληση πληροφοριών που αφορούν τις ξενοδοχειακές μονάδες μιας περιοχής (π.χ. της Νομαρχίας Δωδεκανήσου). Το πακέτο θα βασισθεί σε τρία προγράμματα. Όλα τα δεδομένα θα αποθηκεύονται σε ένα αρχείο εγγραφών σταθερού μήκους το οποίο θα είναι σειριακό. Ο όρος σειριακό αποδίδει την υποχρεωτική προσπέλαση όλων των εγγραφών που προηγούνται από την ζητούμενη, ακόμα και αν αυτές δεν έχουν καμία σημασία για το προγραμμά μας. Το πρώτο πρόγραμμα σκοπό έχει την δημιουργία του αρχείου και την καταχώρηση των δεδομένων. Το δεύτερο πρόγραμμα σκοπό έχει την αναζήτηση πληροφοριών για τα καταλύματα (ξενοδοχεία) για το νησί που θα προσδιορίζει ο χειριστής. Το δεύτερο πρόγραμμα του πακέτου περιορίζεται μόνο στην αναζήτηση πληροφοριών και δεν πραγματοποιεί καμία άλλη μορφή επεξεργασίας (ενημέρωση, διαγραφή και προσθήκη εγγραφής) καθώς αυτές είναι ιδιαίτερα «επώδυνες» στα σειριακά αρχεία. (Θα αναφερθούμε στη γενική μορφή που μπορεί να έχουν οι παραληφθείσες μορφές επεξεργασίας, στο τέλος του κεφαλαίου.) Το τρίτο πρόγραμμα σκοπό έχει τη διάταξη των εγγραφών του αρχείου καταλυμάτων (ξενοδοχείων) ανά νησί (ονομασία νησιού). Στα προγράμματα του πακέτου καταλυμάτων θα χρησιμοποιήσουμε προκαθορισμένες ρουτίνες χειρισμού αρχείων, σε αντίθεση με τις πρωτογενείς κλήσεις χειρισμού αρχείων που χρησιμοποιήσαμε στα δύο προηγούμενα πακέτα. Σκοπός αυτής της αλλαγής είναι:

- να συμβάλει στην εξοικείωση του αναγνώστη με όσες περισσότερες από τις γνώσεις που θα χρειασθεί στην καριέρα του,
- να παρέχει μια ευρύτητα γνώσης για το αντικείνενο που αποφάσισε να ασχοληθεί ο αναγνώστης,
- να επιτρέψει στον αναγνώστη να αποφασίσει ο ίδιος πως θα χειρίζεται τα αρχεία στα πακέτα προγραμμάτων που θα αναπτύξει (ο ίδιος, εκ του μηδενός) και

- να του δώσει τη δυνατότητα να συντηρεί πακέτα που έχουν κατασκευάσει άλλοι.

Η εμμονή σε έναν μόνο από τους προσφερόμενους τρόπους χειρισμού αρχείων (προκαθορισμένες ρουτίνες χειρισμού αρχείων έναντι πρωτογενών κλήσεων χειρισμού αρχείων) θεωρούμε ότι είναι μία, τουλάχιστον, λανθασμένη αντιμετώπιση.

Το πρώτο πρόγραμμα του πακέτου καταλυμάτων (έστω create.c) παρουσιάζεται στη συνέχεια, σε πλήρη ανάπτυξη:

```
/* ******************************************************
   * Author: Nikitas N. Karanikolas, Assistant Professor *
   * Creation Date: February 20, 2009                    *
   ****************************************************** */

#include <stdio.h>

typedef struct {
   char name[40];
   char island[20];
   int single_bed;
   int double_bed;
   int twin_bed;
   int gt_double; /* greater than double room */
   char facilities[100];
} hotel_rec;

char *islands[]={"Patmos", "Arki", "Lipsi", "Leros",
"Kalymnos", "Pserimos", "Kos", "Astypalaia", "Nisyros",
"Symi", "Tilos", "Chalki", "Rhodes", "Kastelorizo",
"Karpathos", "Kasos"};

void upper(char *str) {
   int i;
   for (i=0; i<strlen(str); i++)
      if ((str[i]>=97) && (str[i]<=122)) str[i]=str[i]-32;
}

void blank(char buffer[], int size) {
   int i;
   for (i=0; i<size; i++) buffer[i]=0;
}
```

```
main () {
   hotel_rec h1;
   char tmp_str[40];
   int i, k, howmany;
   FILE *fp;

   fp=fopen("hotels.dat","ab");
   if (fp==NULL) {
      printf("Can not open Hotels.Dat for appending data");
      exit(1);
   }

   do {
      blank(&h1,sizeof(hotel_rec));
      printf("\nGive hotel name: ");
      gets(h1.name);
      printf("[%s]\n",h1.name);
      strcpy(tmp_str,h1.name);
      upper(tmp_str);
      if (strcmp(tmp_str,"QUIT")==0) break;
      do {
         printf("\nGive the island (");
         for (i=0; i<sizeof(islands)/sizeof(char *); i++) {
            printf("%d=%s", i+1, islands[i]);
            if (i==sizeof(islands)/sizeof(char *)-1)
               printf(") ");
            else
               printf(", ");
         }
         scanf("%d",&k);
         /* printf("[%d]\n",k); */
      } while ((k<1) || (k>sizeof(islands)/sizeof(char *)));
      printf("[%s]\n",islands[k-1]);
      strcpy(h1.island,islands[k-1]);
      gets(tmp_str); /* garbage collection */
      printf("\nGive number of single, double, twin-bed and "
             "greater-than-double rooms: ");
      scanf("%d %d %d %d", &h1.single_bed, &h1.double_bed,
            &h1.twin_bed, &h1.gt_double);
      printf("[%d] [%d] [%d] [%d]\n", h1.single_bed,
             h1.double_bed, h1.twin_bed, h1.gt_double);
      gets(tmp_str); /* garbage collection */
      printf("\nDescribe facilities: ");
      gets(h1.facilities);
      printf("[%s]\n",h1.facilities);
      howmany=fwrite(&h1,sizeof(hotel_rec),1,fp);
```

```
      /* printf("\nhowmany=%d\n", howmany); */
   } while (1);
   fclose(fp);
}
```

Η δομή του προγράμματος δημιουργίας του αρχείου και εισαγωγής των δεδομένων, όπως και στα αντίστοιχα προγράμματα των πακέτων ηλεκτρονικής ατζέντας, βασίζεται σε ένα επαναλληπτικό σχήμα do … while. Σε κάθε επανάληψη, ο χειριστής εισαγάγει πληροφορίες που χρησιμοποιούνται για τη σύνθεση μίας εγγραφής που αποθηκεύεται (γράφεται) στο αρχείο ("hotels.dat"). Παρατηρούμε ότι δεν χρησιμοποιούμε εντολές creat ή open. Αντί αυτών χρησιμοποιούμε την εντολή:

```
fp=fopen("hotels.dat","ab");
```

Την εντολή fopen την έχουμε ξανασυναντήσει, και εξηγήσει, στο κεφάλαιο οκτώ. Τότε όμως την είχαμε χρησιμοποιήσει σε συνδυασμό με τις εντολές putc, fprintf και fclose για την δημιουργία ενός αρχείου κειμένου. Τώρα, εδώ, τη χρησιμοποιούμε σε συνδυασμό με την εντολή fwrite και fclose για τη δημιουργία αρχείου εγγραφών:

```
howmany=fwrite(&h1,sizeof(hotel_rec),1,fp);
```

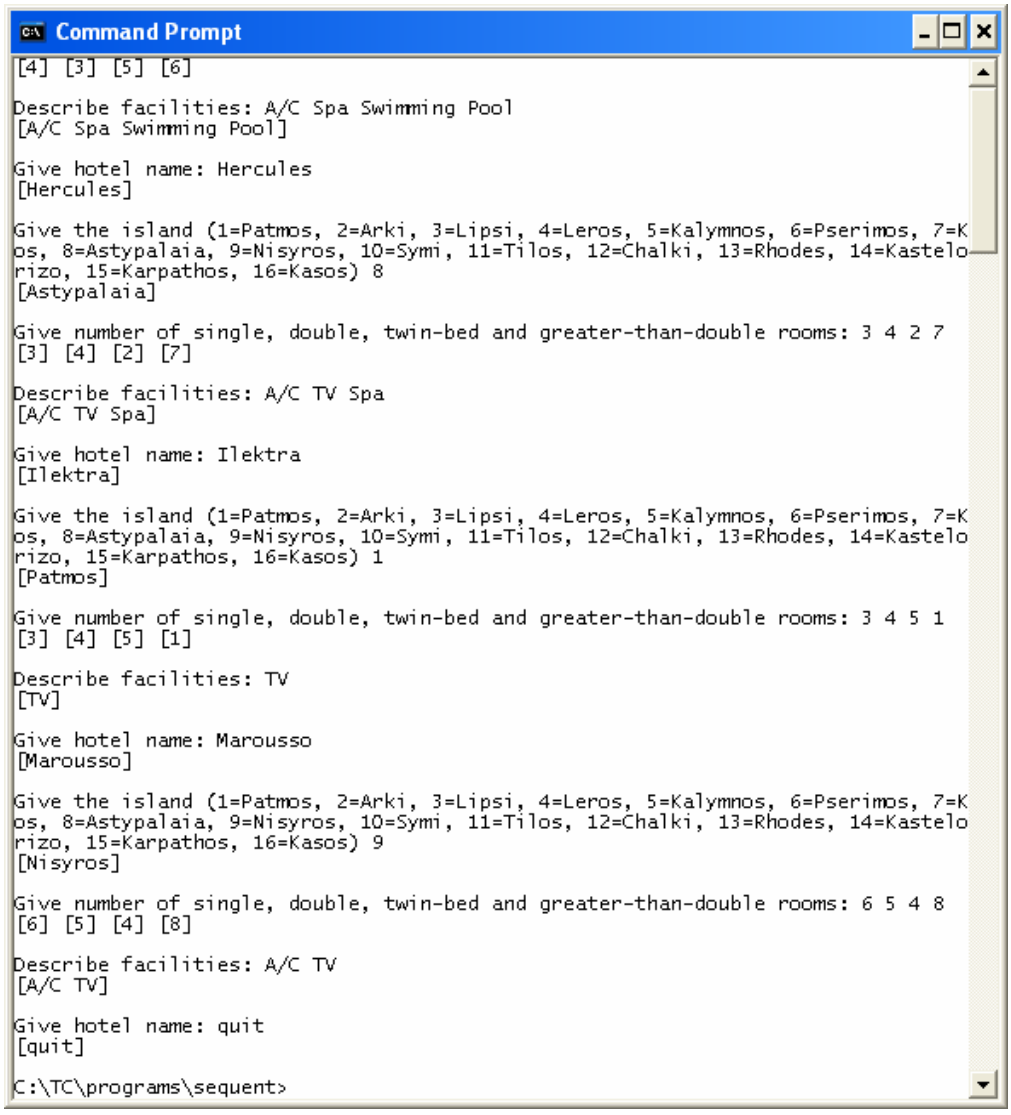
Η επικεφαλίδα της συνάρτησης fwrite είναι:

int fwrite(char *buffer, unsigned int size, int n, FILE *fp);

Το πρώτο όρισμα είναι ένας δείκτης στην εγγραφή (ή σε μια ακολουθία από διαδοχικές εγγραφές) που θα αποθηκευτούν, το δεύτερο όρισμα είναι το μέγεθος της (μίας) εγγραφής, το τρίτο όρισμα είναι το πλήθος των εγγραφών που θα αποθηκευτούν και το τέταρτο όρισμα είναι ο δείκτης σε μια δομή FILE που περιλαμβάνει πληροφορίες για το (ανοικτό) αρχείο στο οποίο θα γίνει η εγγραφή. Η συνάρτηση fwrite επιστρέφει τον αριθμό των bytes που τελικά εγγράφησαν στο αρχείο.

Μπορούμε να πουμε ότι ο συνδυασμός πρωτογενών κλήσεων χειρισμού αρχείων creat / open, write και close μπορεί να αντικατασταθεί από το συνδυασμό προκαθορισμένων ρουτινών χειρισμού αρχείων fopen, fwrite και fclose, για τη δημιουργία και αποθήκευση δεδομένων σε αρχείο εγγραφών.

Η εκτέλεση του πρώτου προγράμματος (create.c) του πακέτου καταλυμάτων (τα τελευταία βήματα αλληλεπίδρασης με το χρήστη) εμφανίζονται στην επόμενη εικόνα:



10.3 Αναζήτηση δεδομένων από σειριακά αρχεία

Το δεύτερο πρόγραμμα, όπως έχουμε αναφέρει στην αρχή του κεφαλαίου, σκοπό έχει την αναζήτηση πληροφοριών για τα καταλύματα (ξενοδοχεία) για το νησί που έχει προσδιορίσει ο χειριστής. Δηλαδή το κριτήριο αναζήτησης είναι η ονομασία του νησιού. Είναι προφανές ότι στο αρχείο δεδομένων μπορεί να μην υπάρχει καμία εγγραφή, να υπάρχει μόνο μία ή ακόμα και να υπάρχουν πολλές εγγραφές αν στο νησί αυτό υπάρχουν (και έχουν καταχωρηθεί) πολλά καταλύματα. Επίσης οι εγγραφές, που τελικά ικανοποιούν το κριτήριο, μπορεί να βρίσκονται διάσπαρτες μέσα στο αρχείο και να μην είναι συγκεντρωμένες σε διαδοχικές θέσεις. Αυτό συμβαίνει γιατί το αρχείο δεν διαθέτει καμία διάταξη, ως προς κανένα από τα πεδία των εγγραφών. Συνεπώς το πρόγραμμα αναζήτησης θα πρέπει υποχρεωτικά να διασχίσει (αναγνώσει) όλες τις εγγραφές αλλά να εμφανίζει τα περιεχόμενα μόνο από εκείνες που ικανοποιούν το κριτήριο αναζήτησης. Το έργο αυτό κάνει με επιτυχία το επόμενο πρόγραμμα (έστω search.c):

```
/* ****************************************************
   * Author: Nikitas N. Karanikolas, Assistant Professor *
   * Creation Date: February 20, 2009                    *
   **************************************************** */

#include <stdio.h>

typedef struct {
   char name[40];
   char island[20];
   int single_bed;
   int double_bed;
   int twin_bed;
   int gt_double;
   char facilities[100];
} hotel_rec;

char *islands[]={"Patmos", "Arki", "Lipsi", "Leros",
"Kalymnos", "Pserimos", "Kos", "Astypalaia", "Nisyros",
"Symi", "Tilos", "Chalki", "Rhodes", "Kastelorizo",
"Karpathos", "Kasos"};

main () {
   hotel_rec h1;
   int i, k, howmany;
```

```
   FILE *fp;

   fp=fopen("hotels.dat","rb");
   if (fp==NULL) {
      printf("Can not open Hotels.Dat for reading data");
      exit(1);
   }

   do {
      printf("\nGive the island (");
      for (i=0; i<sizeof(islands)/sizeof(char *); i++) {
         printf("%d=%s", i+1, islands[i]);
         if (i==sizeof(islands)/sizeof(char *)-1)
            printf(") ");
         else
            printf(", ");
      }
      scanf("%d",&k);
   } while ((k<1) || (k>sizeof(islands)/sizeof(char *)));
   printf("[%s]\n",islands[k-1]);


   do {
      howmany=fread(&h1,sizeof(hotel_rec),1,fp);
      if (howmany<1) break;
      if (strcmp(h1.island,islands[k-1])==0) {
         printf("\nname:%s, facilities:%s\n",
            h1.name,h1.facilities);
         printf("rooms single:%d, double:%d, "
            "twin:%d, greater:%d\n",
            h1.single_bed, h1.double_bed, h1.twin_bed,
            h1.gt_double);
      }
   } while (1);
   fclose(fp);
}
```

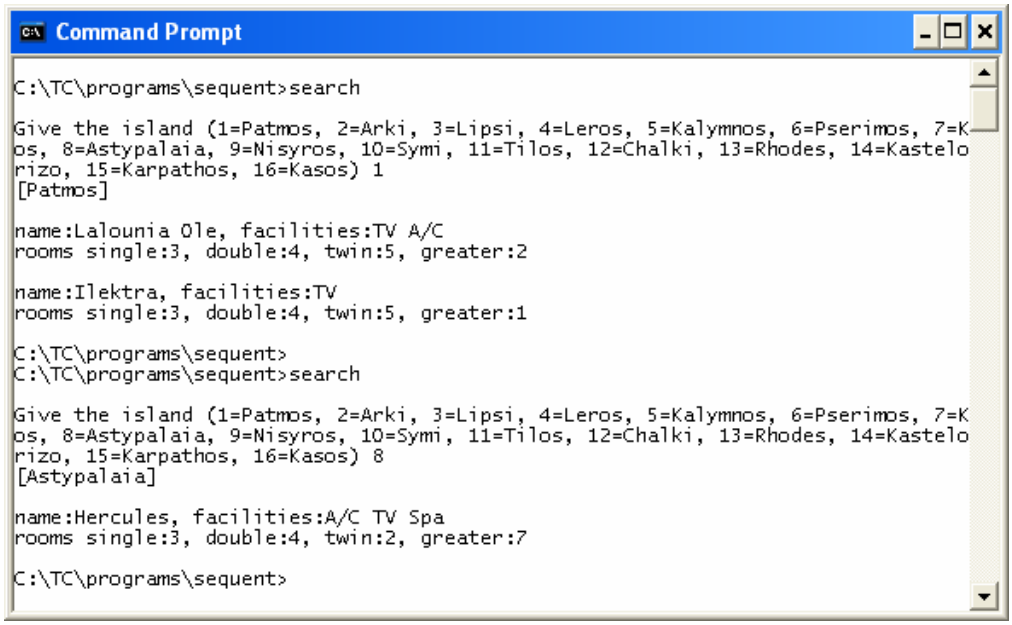
Το πρόγραμμα αυτό ζητά από το χειριστή να προσδιορίσει το νησί για το οποίο ζητάει πληροφορίες καταλυμάτων. Στη συνέχεια, με τη βοήθεια ενός do … while επαναληπτικού σχήματος και του πίνακα islands, υποχρεώνει το χειριστή να επιλέξει ένα από τα τυποποιημένα ονόματα των νησιών της περιοχής ενδιαφέροντος. Το δεύτερο επαναληπτικό σχήμα διασχίζει (διαβάζει) μία προς μία όλες τις εγγραφές. Σε κάθε βήμα της επανάληψης διαβάζεται μία εγγραφή, με την εντολή fread, και ελέγχεται αν το όνομα νησιού που

περιέχει η εγγραφή είναι ίδιο με αυτό το όνομα που έχει προσδιορίσει ο χειριστής. Αν ο έλεγχος έχει αληθές αποτέλεσμα, εμφανίζονται τα στοιχεία της εγγραφής (του καταλύματος). Διαφορετικά τα στοιχεία της εγγραφής δεν αξιοποιούνται (δεν τα κάνει τίποτα το πρόγραμμα). Το πρόγραμμα αυτό χρησιμοποιεί το συνδυασμό ρουτινών fopen, fread και fclose για να σαρώσει διαδοχικά τα δεδομένα του αρχείου καταλυμάτων. Μπορούμε να πουμε ότι ο συνδυασμός πρωτογενών κλήσεων χειρισμού αρχείων open, read και close, που είδαμε σε προηγούμενα κεφάλαια, μπορεί να αντικατασταθεί από το συνδυασμό προκαθορισμένων ρουτινών χειρισμού αρχείων fopen, fread και fclose, για την σάρωση των δεδομένων από αρχείο εγγραφών. Η μόνη νέα συνάρτηση είναι η fread. Η επικεφαλίδα της οποίας είναι:

```
int fread(char *buffer, unsigned int size, int n, FILE *fp);
```

Το πρώτο όρισμα είναι ένας δείκτης στην εγγραφή (ή σε μια ακολουθία από διαδοχικές εγγραφές) που θα φορτωθούν (καταχωρηθούν) τα στοιχεία (εγγραφές) που θα διαβάσει η fread, το δεύτερο όρισμα είναι το μέγεθος της (μίας) εγγραφής, το τρίτο όρισμα είναι το πλήθος των εγγραφών που απαιτείται να διαβασθούν και το τέταρτο όρισμα είναι ο δείκτης σε μια δομή FILE που περιλαμβάνει πληροφορίες για το (ανοικτό) αρχείο από το οποίο θα γίνει η ανάγνωση. Η συνάρτηση επιστρέφει τον αριθμό των εγγραφών που τελικά διαβάσθηκαν. Συνήθως, όταν η συνάρτηση επιστρέψει ένα αριθμό μικρότερο από τον αριθμό εγγραφών που απαιτήθηκε να διαβασθούν, έχουμε φτάσει στο τέλος αρχείου. Σε αυτή την ιδιότητα (υπόθεση) βασίζεται και το πρόγραμμά μας και διακόπτει (τερματίζει) το επαναληπτικό σχήμα ανάγνωσης στοιχείων από το αρχείο των καταλυμάτων, όταν η fread διαβάσει λιγότερες από μία εγγραφές.

Η εκτέλεση του δεύτερου προγράμματος του πακέτου καταλυμάτων (δύο διαδοχικές εκτελέσεις) εμφανίζονται στην επόμενη εικόνα:



10.4 Αλγόριθμοι ενημέρωσης και διαγραφής εγγραφών από σειριακά αρχεία

Η ενημέρωση μίας ή περισσοτέρων εγγραφών ενός σειριακού αρχείου, με δεδομένη την αδυναμία επιστροφής σε μία εγγραφή που έχει αναγνωσθεί (μετακίνηση προς τα πίσω), επιβάλει τη χρήση δύο αρχείων. Ο επόμενος Αλγόριθμος (ψευδοκώδικας) αποτυπώνει την κεντρική ιδέα στην οποία μπορούμε να βασισθούμε για να κατασκευάσουμε ένα πρόγραμμα που μεταβάλει κάποιες εγγραφές (όποιες επαληθεύουν κάποιο κριτήριο) ενός σειριακού αρχείου:

```
Άνοιξε για ανάγνωση το κύριο αρχείο
Δημιούργησε ένα βοηθητικό αρχείο για εγγραφή
Όσο δεν εξαντλήθηκαν οι εγγραφές του κύριου αρχείου
   Δίαβασε μία εγγραφή από το κύριο αρχείο
   Αν η εγγραφή που αναγνώσθηκε ικανοποιεί το κριτήριο
      Κάνε μεταβολές στην εγγραφή
      Αποθήκευσε την αλλαγμένη εγγραφή στο βοηθητικό αρχείο
   Διαφορετικά
      αποθήκευσε την εγγραφή στο βοηθητικό αρχείο
   Τέλος Αν
```