Προγραμματισμός Υπολογιστών

Αρχεία

Νικόλαος Ζ. Ζάχαρης
Καθηγητής

Πανεπιστήμιο Δυτικής Αττικής
Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών

Τα είδη των αρχείων

Οι πληροφορίες στον δίσκο του υπολογιστή οργανώνονται με τη μορφή αρχείων, τα οποία χωρίζονται σε δύο μεγάλες κατηγορίες, τα αρχεία κειμένου (text) και τα δυαδικά (binary) αρχεία. Και τα δύο είδη αρχείων περιέχουν bits (0 ή 1) τα οποία στα αρχεία κειμένου αναπαριστούν χαρακτήρες ενώ στα δυαδικά αρχεία μπορούν να αναπαριστούν οποιοδήποτε τύπο δεδομένων.

flower.png

flower.png

flower.png

```
1  ‰PNG
2  SUB
3  NUL NUL NUL
4  IHDR NUL NUL SOH Ξ NUL NU
5  liCCPICC Profile NUL
6  ½H '.*1  DLE Jî NUL "6D
7  NUL °¦eμΩï‡mi NAK NUL ]
8  M@ DLE „…Θ DLE NAK R‡t C
9  C: Ob DC2 ï NAK z BEL #0
```

message.txt

```
1  Φιλοδοξία της Ένωσης είναι να
2  δοθεί η δυνατότητα σε όλους
3  τους Ευρωπαίους να κυκλοφορούν
4  ασφαλέστερα και να χρησιμοποιούν
5  λιγότερο ρυπογόνα οχήματα και
6  πιο προηγμένες τεχνολογικές λύσεις,
7  υποστηρίζοντας, ταυτόχρονα, την
8  ανταγωνιστικότητα της βιομηχανίας
9  της Ε.Ε.
```

Ένα δυαδικό αρχείο μπορεί να αναπαριστά πληροφορία από ένα ή περισσότερους τύπους δεδομένων. Για παράδειγμα σε ένα αρχείο μπορεί να είναι αποθηκευμένη μία εικόνα και ένα βίντεο και κείμενο ή οτιδήποτε άλλο. Για την ανάγνωση της πληροφορίας από το αρχείο θα πρέπει να γνωρίζουμε τον τρόπο της αποθήκευσης του.

Ο τελευταίος χαρακτήρας σε ένα αρχείο είναι το End Of File (EOF) που δείχνει το τέλος του αρχείου.

Οι προκαθορισμένες ροές στη C

Η γλώσσα C χειρίζεται την είσοδο και την έξοδο από ένα πρόγραμμα, σαν την ανάγνωση και εγγραφή πληροφοριών από ροές χαρακτήρων (streams), οι οποίες είναι είτε εισόδου – από το εξωτερικό περιβάλλον προς το πρόγραμμα – είτε εξόδου – από το πρόγραμμα προς το εξωτερικό περιβάλλον. Για αυτό το λόγο δημιουργεί αυτόματα για κάθε πρόγραμμα τρείς ροές :

Όνομα ροής	Περιγραφή
stdin	Standard Input - είναι η προκαθορισμένη ροή για την είσοδο σε ένα πρόγραμμα και συνδέεται με τη συσκευή πληκτρολόγιο.
stdout	Standard Output - είναι η προκαθορισμένη ροή για την έξοδο από ένα πρόγραμμα και συνδέεται με τη συσκευή οθόνης.
stderror	Standard Error - είναι η προκαθορισμένη ροή για την έξοδο σφαλμάτων από ένα πρόγραμμα και συνδέεται με τη συσκευή οθόνης.

Αυτά τα ονόματα είναι δείκτες τύπου struct FILE και είναι και καθολικές μεταβλητές που ορίζονται στο stdio.h. Η αρχικοποίηση αυτών των δεικτών και η σύνδεση τους με τις αντίστοιχες συσκευές γίνεται αυτόματα.

Άνοιγμα αρχείου

Για να κάνουμε διάφορες λειτουργίες σε ένα αρχείο, όπως είναι η ανάγνωση και η εγγραφή δεδομένων, θα πρέπει πρώτα να δημιουργήσουμε ένα δείκτη τύπου FILE ώστε στη συνέχεια η αναφορά μας στο αρχείο να γίνεται με τη χρήση του δείκτη. Η fopen μας επιτρέπει να δημιουργήσουμε ένα δείκτη σε ένα αρχείο και η σύνταξη της είναι :

FILE * fopen (const char * filename, const char * mode);

filename : είναι ένα αλφαριθμητικό που αντιστοιχεί στο πλήρες μονοπάτι του αρχείου. Για παράδειγμα "c:\\data\\may.txt" ενώ αν είναι μόνο το όνομα του αρχείου π.χ. "may.txt" σημαίνει ότι το αρχείο είναι στο ίδιο φάκελο με την εφαρμογή.

mode : είναι ένα αλφαριθμητικό που αντιστοιχεί στο είδος της πρόσβασης που επιθυμούμε στο αρχείο, δες επόμενη διαφάνεια, και μπορεί να είναι μόνο για ανάγνωση δεδομένων, μόνο για εγγραφή δεδομένων, μόνο για προσάρτηση (προσθήκη) δεδομένων στο τέλος του αρχείου. Επίσης μπορούμε να ορίσουμε ότι σε ένα αρχείο θέλουμε να κάνουμε και ανάγνωση και εγγραφή καθώς επίσης να ορίσουμε ότι πρόκειται για δυαδικό αρχείο.

Η συνάρτηση fopen επιστρέφει ένα δείκτη στο συγκεκριμένο αρχείο ή τη τιμή NULL για να δηλώσει σφάλμα στη δημιουργία του δείκτη.

Οι καταστάσεις για το άνοιγμα ενός αρχείου

Κατάσταση	Περιγραφή
r	Ανοίγει ένα αρχείο σε κατάσταση ανάγνωσης. Το αρχείο πρέπει να υπάρχει.
w	Ανοίγει ένα αρχείο σε κατάσταση εγγραφής. Αν δεν υπάρχει το αρχείο το δημιουργεί, διαφορετικά καταστρέφει τα περιεχόμενα του.
a	Ανοίγει το αρχείο σε κατάσταση προσάρτησης στο τέλος του αρχείου και αν δεν υπάρχει τότε δημιουργεί το αρχείο.
r+	Ανοίγει ένα αρχείο κειμένου σε κατάσταση ανάγνωσης και εγγραφής. Το αρχείο πρέπει να υπάρχει.
w+	Ανοίγει ένα αρχείο σε κατάσταση ανάγνωσης και εγγραφής. Αν δεν υπάρχει το αρχείο το δημιουργεί, διαφορετικά καταστρέφει τα περιεχόμενα του.-
a+	Ανοίγει ένα αρχείο κειμένου σε κατάσταση ανάγνωσης και εγγραφής. Αν δεν υπάρχει τότε δημιουργεί το αρχείο διαφορετικά κάνει προσάρτηση στο τέλος.
rb	Ότι και οι ανωτέρω καταστάσεις και το b δηλώνει δυαδικό αρχείο.
wb	
ab	
rb+	
wb+	
ab+	

Κλείσιμο αρχείου

Μετά το επιτυχημένο άνοιγμα ενός αρχείου και με την ολοκλήρωση όλων των ενεργειών που θέλουμε να κάνουμε σε αυτό, όπως της ανάγνωσης ή/και εγγραφής δεδομένων, θα πρέπει πάντα να κλείνουμε το αρχείο με τη συνάρτηση fclose, η οποία συντάσσεται όπως παρακάτω :

int fclose (FILE * stream);

stream : είναι ο δείκτης στο αρχείο που θέλουμε να κλείσουμε και να αποδεσμεύσουμε τους πόρους (εσωτερικοί buffers) που χρησιμοποιούνται για την ανάγνωση και την εγγραφή δεδομένων.

Η fclose επιστρέφει την τιμή 0 για το επιτυχημένο κλείσιμο του αρχείου και την τιμή EOF σε περίπτωση σφάλματος. Μετά τη κλήση της fclose δεν μπορούμε να χρησιμοποιήσουμε το δείκτη για καμία ενέργεια στο αρχείο, αφού ακόμα σε περίπτωση σφάλματος έχουν αποδεσμευτεί όλοι οι πόροι που σχετίζονται με το δείκτη.

Το EOF ορίζεται σαν σταθερά στην επικεφαλίδα stdio.h και έχει συνήθως τη τιμή -1 που δηλώνει είτε το τέλος του αρχείου είτε ότι έχει συμβεί ένα σφάλμα.

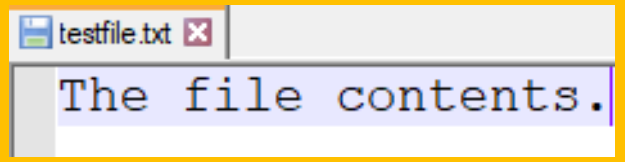
Παράδειγμα άνοιγμα και κλείσιμο αρχείου

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
  FILE *fp;
  fp = fopen ("testfile.txt", "r");
  if (fp == NULL)
  {
    printf("Error opening the file.\n");
    printf("Verify that file exists.\n");
  }
  else
  {
    printf("The file opened successfully\n");
    fclose (fp);
  }
    return 0;
}
```

Όταν εκτελέσετε την εφαρμογή θα εμφανιστεί το παρακάτω μήνυμα επειδή δεν υπάρχει το αρχείο testfile.txt. Αν υπάρχει να το διαγράψετε.

```
Error opening the file.
Verify that file exists.
```

Εν συνεχεία, με ένα επεξεργαστή κειμένου, π.χ. Notepad, δημιουργήστε το testfile.txt και με περιεχόμενο όπως παρακάτω :



Όταν εκτελέσετε ξανά την εφαρμογή θα εμφανιστεί το διπλανό μήνυμα επειδή τώρα υπάρχει το testfile.txt

```
The file opened successfully
```

Η συνάρτηση fprintf

Για την εγγραφή μορφοποιημένης πληροφορίας σε αρχεία κειμένου χρησιμοποιούμε τη :

int fprintf(FILE *fptr, const char *format, ...);

fptr: είναι ο δείκτης στο αρχείο που θα κάνουμε την εγγραφή των δεδομένων.

format : είναι το αλφαριθμητικό μορφοποίησης των δεδομένων και ισχύουν τα ίδια με τη printf π.χ. "%d %s" για την εγγραφή ενός ακεραίου και ενός αλφαριθμητικού.

... : προαιρετικά τα δεδομένα.

Έχει την ίδια χρήση με τη printf μόνο που έχει ένα επιπλέον όρισμα, το δείκτη στο αρχείο που πρόκειται να γίνει η εγγραφή των δεδομένων.

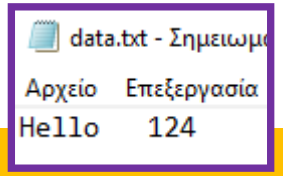
Όταν η εκτέλεση της συνάρτησης γίνει με επιτυχία τότε επιστρέφει το πλήθος των χαρακτήρων που αποθηκεύτηκαν, διαφορετικά επιστρέφει μια αρνητική τιμή.

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    fprintf(stdout, "%s\t%d", "Hello", 124);
    return 0;
}
```

Έξοδος στην οθόνη

```
Hello   124
```

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    FILE *fp = fopen("data.txt", "w");
    fprintf(fp, "%s\t%d", "Hello", 124);
    fclose(fp);
    return 0;
}
```



Η συνάρτηση fscanf

Για την ανάγνωση μορφοποιημένης πληροφορίας από αρχεία κειμένου χρησιμοποιούμε τη :

int fscanf(FILE *fptr, const char *format, ...);

fptr: είναι ο δείκτης στο αρχείο που θα κάνουμε την ανάγνωση των δεδομένων.

format : είναι το αλφαριθμητικό μορφοποίησης των δεδομένων

… : οι διευθύνσεις των μεταβλητών για την αποθήκευση των δεδομένων.

Έχει την ίδια χρήση με τη scanf μόνο που έχει ένα επιπλέον όρισμα, το δείκτη στο αρχείο που πρόκειται να γίνει η ανάγνωση των δεδομένων.

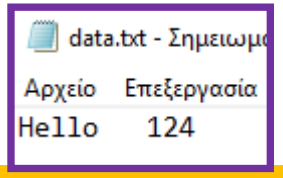
Όταν η εκτέλεση της συνάρτησης γίνει με επιτυχία τότε επιστρέφει το πλήθος των αναθέσεων τιμών σε μεταβλητές, διαφορετικά επιστρέφει τη τιμή 0.

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    char str[50], c;
    int x;
    fscanf(stdin, "%s", str);
    fscanf(stdin, "%d", &x);
    printf("%s\n", str);
    printf("%d\n", x);
    return 0;
}
```

```
hello
123
hello
123
```

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    char str[50], c;
    int x;
    FILE *fp = fopen("data.txt", "r");
    fscanf(fp, "%s\t%d", str, &x);
    printf("%s\n", str);
    printf("%d\n", x);
    fclose(fp);
    return 0;
}
```

```
Hello
124
```



Η συνάρτηση fgets

Για την ανάγνωση μιας ολόκληρης γραμμής από αρχείο κειμένου χρησιμοποιούμε τη :

char *fgets(char *str, int n, FILE *fptr)

fptr: είναι ο δείκτης στο αρχείο που θα κάνουμε την ανάγνωση των δεδομένων.

str : είναι το αλφαριθμητικό οποίο θα αποθηκεύσουμε τα δεδομένα

n : το μέγιστο πλήθος των χαρακτήρων που επιτρέπεται να αποθηκευτούν στο str.

Η συνάρτηση σταματά όταν διαβάσει το χαρακτήρα \n τον οποίο αποθηκεύει στο αλφαριθμητικό, είτε διαβάσει n-1 χαρακτήρες είτε φτάσει στο τέλος του αρχείου.

Όταν η εκτέλεση της συνάρτησης γίνει με επιτυχία τότε επιστρέφει τη διεύθυνση του str, διαφορετικά επιστρέφει τη τιμή NULL.

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    char name[10];
    printf("Type your name : ");
    fgets(name, 10, stdin);
    printf("Your name is : %s", name);
    return 0;
}
```

```
Type your name : 12
Your name is : 12
```

```
Type your name : 123456789012345
Your name is : 123456789
```

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    char line[100];
    FILE *fp = fopen("data.txt","r");

    fgets(line, 100, fp);
    printf("First Line : %s", line);
    fgets(line, 100, fp);
    printf("Second Line : %s", line);
    fclose(fp);
    return 0;
}
```

data.txt - Σημειω

Αρχείο	Επεξεργασ
May	1240
June	143
July	221

```
First Line : May         1240
Second Line : June       143
```

Η συνάρτηση feof

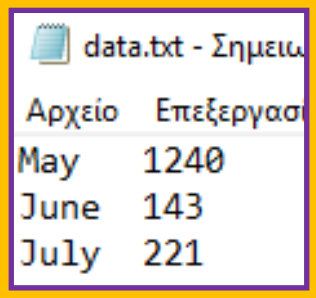
Για να ελέγξουμε αν υπάρχουν και άλλα δεδομένα προς ανάγνωση χρησιμοποιούμε τη

int feof(FILE *fptr)

fptr: είναι ο δείκτης στο αρχείο που θα κάνουμε έλεγχο αν έφτασε στο τέλος ή όχι, δηλαδή αν υπάρχουν και άλλα δεδομένα προς ανάγνωση.

Αν υπάρχουν και άλλα δεδομένα προς ανάγνωση τότε η συνάρτηση επιστρέφει τη τιμή 0 διαφορετικά μια θετική τιμή.

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    char line[100];
    FILE *fp = fopen("data.txt","r");
    int n = 1;
    while(!feof(fp))  {
      fgets(line, 100, fp);
      printf("%d )  Line : %s", n++, line);
    }

    fclose(fp);
    return 0;
}
```




```
1 )  Line : May   1240
2 )  Line : June  143
3 )  Line : July  221
```

Συνήθως χρησιμοποιούμε την feof ώστε να διαβάζουμε επαναληπτικά πληροφορίες από ένα αρχείο μέχρις ότου να φτάσουμε στο τέλος του.

Οι συναρτήσεις ferror και clearerr

Για την ανάγνωση της τιμής του αναγνωριστικού σφαλμάτων (error indicator) σε ένα αρχείο χρησιμοποιούμε τη :

int ferror(FILE *fptr)

fptr: είναι ο δείκτης στο αρχείο που θα κάνουμε έλεγχο αν έχει συμβεί σφάλμα.

Αν υπάρχει σφάλμα τότε η συνάρτηση επιστρέφει μια θετική τιμή διαφορετικά τη τιμή 0

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    char line[100];
    FILE *fp = fopen("salary.txt","w");
    fgets(line, 100, fp);
    if(ferror(fp)){
        printf("An error has occured.");
    }

    clearerr(fp);

    if(ferror(fp)){
        printf("Another error has occured.");
    }

    fclose(fp);
    return 0;
}
```

Δημιουργούμε το αρχείο salary.txt μόνο για εγγραφή και εν συνεχεία προσπαθούμε να κάνουμε ανάγνωση τιμών

```
An error has occured.
```

Για να καθαρίσουμε το εσωτερικό αναγνωριστικό (indicator) λαθών σε ένα δείκτη αρχείου χρησιμοποιούμε τη συνάρτηση :

void clearerr(FILE *fptr)

fptr: είναι ο δείκτης στο αρχείο που θα μηδενίσουμε το εσωτερικό αναγνωριστικό σφαλμάτων

Οι συναρτήσεις fgetc και fputc

Για την ανάγνωση του επόμενου χαρακτήρα από ένα αρχείο χρησιμοποιούμε τη :

int fgetc(FILE *fptr)

fptr: είναι ο δείκτης στο αρχείο που θα διαβάσουμε το χαρακτήρα.

Η συνάρτηση επιστρέφει τον επόμενο χαρακτήρα που διάβασε το αρχείο ή το EOF σε περίπτωση που δεν υπάρχουν άλλα δεδομένα για ανάγνωση.

Για την εγγραφή ενός χαρακτήρα σε ένα αρχείο χρησιμοποιούμε τη :

int fputc(int ch, FILE *fptr)

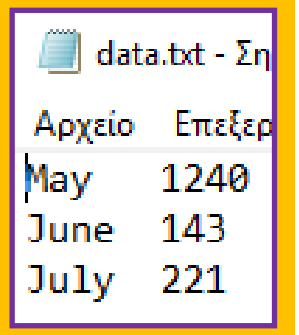
ch : είναι ο χαρακτήρας που θα αποθηκεύσουμε στο αρχείο.

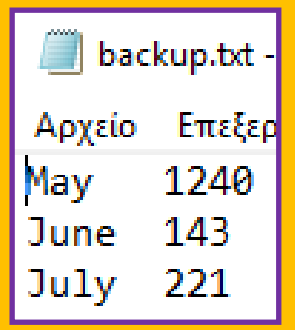
fptr: είναι ο δείκτης στο αρχείο που θα διαβάσουμε το χαρακτήρα.

Η συνάρτηση επιστρέφει τον ίδιο χαρακτήρα σε περιπτωση επιτυχημένης εγγραφής διαφορετικά επιστρέφει το EOF.

Και οι δύο συναρτήσεις κάνουν casting τον χαρακτήρα (unsigned char) σε int.

Αντιγραφή αρχείου κειμένου






```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    FILE *fpsrc, *fpdst;
    int ch;

    fpsrc = fopen("data.txt", "r");
    fpdst = fopen("backup.txt", "w");

    while(1){
        ch = fgetc(fpsrc);
        if(feof(fpsrc))
            break;

        fputc(ch, fpdst);
    }
    fclose(fpsrc);
    fclose(fpdst);
    printf("The backup.txt created successfully.");
    return 0;
}
```

Για δυαδικό αρχείο χρησιμοποιήστε το rnd.pdf

Για δυαδικό αρχείο "rb"

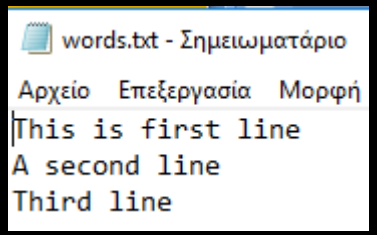
Για δυαδικό αρχείο "wb"

Ο ίδιος κώδικας μπορεί να χρησιμοποιηθεί και για την αντιγραφή ενός δυαδικού αρχείου.

Δημιουργία αρχείου κειμένου

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
   FILE *fp;
   char ch;

   fp = fopen("words.txt", "w");
   printf("Enter data and type Ctrl-Z (windows) or Ctrl-D (Linux) to close file.\n");

   while( (ch = getchar()) != EOF) {
      putc(ch, fp);
   }
   fclose(fp);


   printf("\n\nFile contents\n\n");
   fp = fopen("words.txt", "r");

   while((ch = getc(fp)) != EOF)
     printf("%c",ch);


   fclose(fp);
   return 0;
}
```

```
Enter data and type Ctrl-Z (windows) or Ctrl-D (Linux) to close file.
This is first line
A second line
Third line
^Z


File contents

This is first line
A second line
Third line
```



Οι συναρτήσεις getw και putw

Οι δύο συναρτήσεις χρησιμοποιούνται αντίστοιχα για την ανάγνωση και την εγγραφή ενός ακεραίου αριθμού σε αρχείο. Η σύνταξη της getw είναι :

int getw(FILE *fp);

fp: Είναι ο δείκτης στο αρχείο μέσα στο οποίο θα διαβάσουμε τον ακέραιο αριθμό.

Η getw επιστρέφει τον ακέραιο αριθμό που διάβασε με επιτυχία από το αρχείο διαφορετικά επιστρέφει την τιμή EOF σε περίπτωση σφάλματος ή τέλους αρχείου.

int putw(int no, FILE *fp);

int no : είναι ο ακέραιος αριθμός που θα αποθηκεύσουμε στο αρχείο.

fp: Είναι ο δείκτης στο αρχείο μέσα στο οποίο θα αποθηκεύσουμε τον ακέραιο αριθμό.

Η putw επιστρέφει τον ακέραιο αριθμό που αποθήκευσε με επιτυχία στο αρχείο διαφορετικά επιστρέφει την τιμή EOF σε περίπτωση σφάλματος ή τέλους αρχείου.

Παράδειγμα getw και putw

```c
#include <stdio.h>
#include <stdlib.h>

int main (){
   FILE *fp;
   int x=61, y=62, z=63, no;
   fp = fopen ("test.dat","w");
   putw(x,fp);
   putw(y,fp);
   putw(z,fp);
   fclose(fp);

   fp = fopen ("test.dat","r");
   int n = 1;
   while( (no= getw(fp)) != EOF ) {
      printf("%d number in test.dat file is %d \n",n++, no);
   }
   fclose(fp);
   return 0;
}
```

```
1 number in test.dat file is 61
2 number in test.dat file is 62
3 number in test.dat file is 63
```

Οι συναρτήσεις fwrite και fread (1-2)

Οι δύο συναρτήσεις χρησιμοποιούνται αντίστοιχα για την ανάγνωση και την εγγραφή δεδομένων σε δυαδικά αρχεία. Η σύνταξη της fwrite είναι :

size_t fwrite(const void *ptr, size_t size, size_t n, FILE *fp);

ptr: είναι ο δείκτης στο τμήμα της μνήμης που περιέχει τα δεδομένα που θέλουμε να αποθηκεύσουμε στο αρχείο.

size: Είναι το μέγεθος σε bytes του καθενός από τα δεδομένα που πρόκειται να αποθηκευτούν στο αρχείο.

n: Είναι το πλήθος των δεδομένων που πρόκειται να αποθηκευτούν στο αρχείο.

fp: Είναι ο δείκτης στο αρχείο μέσα στο οποίο θα αποθηκεύσουμε τα δεδομένα.

Η fwrite επιστρέφει το πλήθος από τα αντικείμενα που αποθηκεύτηκαν στο αρχείο. Μία τιμή επιστροφής που είναι διαφορετική από το n δηλώνει σφάλμα κατά την αποθήκευση

Οι συναρτήσεις fwrite και fread (2-2)

Η σύνταξη της fread είναι :

size_t fread(void *ptr, size_t size, size_t n, FILE *fp);

ptr: είναι ο δείκτης στο τμήμα της μνήμης που θα αποθηκεύσουμε τα δεδομένα που θα διαβάσουμε από το αρχείο.

size: Είναι το μέγεθος σε bytes του καθενός από τα δεδομένα που πρόκειται να διαβάσουμε από το αρχείο.

n: Είναι το πλήθος των δεδομένων που πρόκειται να διαβάσουμε από το αρχείο.

fp: Είναι ο δείκτης στο αρχείο από το οποίο θα διαβάσουμε τα δεδομένα.

Η fread επιστρέφει το πλήθος από τα δεδομένα που διαβάσαμε από το αρχείο και μπορεί να διαφέρει από το πλήθος **n** γιατί είτε έχει συμβεί λάθος και μπορούμε να το ελέγξουμε με τη χρήση της ferror είτε είναι το τέλος του αρχείου και δεν υπάρχουν άλλα δεδομένα, και μπορούμε να το ελέγξουμε με τη χρήση της feof.

Εγγραφή και ανάγνωση δομής δεδομένων (1-3)

```
1) id = 21 name = Jim    surname = Brown        salary = 34657.612000
2) id = 11 name = Tom    surname = Red  salary = 24627.170000
3) id = 9 name = Mary    surname = Yello        salary = 44127.120000

The file contains 3 records.
```

```
#include <stdio.h>
#include <stdlib.h>

typedef struct {
    int id;
    char pname[20];
    char psurname[20];
    double salary;
} Person;

void SaveData() {
    Person jim = {21, "Jim", "Brown", 34657.612};
    Person tom = {11, "Tom", "Red", 24627.17};
    Person mary = {9, "Mary", "Yello", 44127.12};

    FILE *fp = fopen ("citizens.dat", "wb");

    if (fp == NULL)  {
        printf("\nError opening file citizens.dat\n");
        exit (1);
    }
```

Το αρχείο με τις 3 εγγραφές θα χρησιμοποιηθεί για ανάγνωση και από άλλα προγράμματα.

Εγγραφή και ανάγνωση δομής δεδομένων (2-3)

```
 if( fwrite (&jim, sizeof(jim), 1, fp) != 1 ){
       printf("\nError writing data to file citizens.dat\n");
       exit (1);
    }

    if( fwrite (&tom, sizeof(Person), 1, fp) != 1 ){
       printf("\nError writing data to file citizens.dat\n");
       exit (1);
    }

    if( fwrite (&mary, sizeof(Person), 1, fp) != 1 ){
       printf("\nError writing data to file citizens.dat\n");
       exit (1);
    }

    fclose(fp);
}
```

Το μέγεθος σε bytes του jim και του Person είναι το ίδιο.

Εγγραφή και ανάγνωση δομής δεδομένων (3-3)

```
void ReadData() {
     FILE *fp = fopen ("citizens.dat", "rb");
    if (fp == NULL)  {
        printf("\nError opening file citizens.dat\n");
        exit (1);
    }

    Person temp;
    int n = 0;
    while(fread( &temp, sizeof(Person), 1, fp) == 1) {
       printf ("%d) id = %d name = %s \t surname = %s \tsalary = %lf\n", ++n, temp.id,
temp.pname, temp.psurname, temp.salary);
   }

   printf("\nThe file contains %d records.\n", n);
   fclose(fp);
}

int main(int argc, char *argv[]) {
    SaveData();
    ReadData();
   return 0;
}
```

Ανάγνωση δεδομένων από αρχείο σε πίνακα (1-2)

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct {
    int id;
    char pname[20];
    char psurname[20];
    double salary;
} Person;

Person Citizens[3];
Person *Omada;

void ShowData(Person *Atoma, int n){
  int i;
  for(i = 0; i < n; i++){
     printf ("%d) id = %d name = %s \t surname = %s \tsalary = %lf\n", i,
        Atoma[i].id, Atoma[i].pname, Atoma[i].psurname, Atoma[i].salary);
   }
}

int main(int argc, char *argv[]) {
    printf("Size of Person : %d\n", sizeof(Person));
    printf("Size of Citizens : %d\n", sizeof(Citizens));
    printf("Size of Omada : %d\n", sizeof(Omada));

    ReadData();
    ReadDataDynamic();
    return 0;
}
```

```
Size of Person : 56
Size of Citizens : 168
Size of Omada : 8
0) id = 21 name = Jim    surname = Brown        salary = 34657.612000
1) id = 11 name = Tom    surname = Red  salary = 24627.170000
2) id = 9 name = Mary    surname = Yello        salary = 44127.120000
0) id = 21 name = Jim    surname = Brown        salary = 34657.612000
1) id = 11 name = Tom    surname = Red  salary = 24627.170000
2) id = 9 name = Mary    surname = Yello        salary = 44127.120000
```

Ανάγνωση δεδομένων από αρχείο σε πίνακα (2-2)

```
void ReadData(){
    FILE *fp = fopen ("citizens.dat", "rb");
    if (fp == NULL)  {
        printf("\nError opening file citizens.dat\n");
        exit (1);
    }

    if(fread( Citizens, sizeof(Citizens), 1, fp) != 1) {
       printf("\nAn error has occurred.\n");
   }
   else {
    ShowData(Citizens, 3);
   }
   fclose(fp);
}

void ReadDataDynamic(){
    FILE *fp = fopen ("citizens.dat", "rb");
    if (fp == NULL)  {
        printf("\nError opening file citizens.dat\n");
        exit (1);
    }

    if( (Omada = (Person *) malloc(3 * sizeof(Person))) == NULL)  {
        printf("Not enough memory.\n");
        exit(1);
    }
    if(fread( Omada, sizeof(Person), 3, fp) != 3) {
       printf("\nAn error has occurred.\n");
   }
   else {
    ShowData(Omada, 3);
   }
   fclose(fp);
   free(Omada);
}
```

Οι συναρτήσεις fseek, rewind() και ftell (1-2)

Για τη μετακίνηση του δείκτη θέσης είτε για ανάγνωση ή εγγραφή, σε συγκεκριμένο σημείο μέσα στο αρχείο χρησιμοποιούμε τη

int fseek(FILE *fptr, long int offset, int origin)

fptr: είναι ο δείκτης στο αρχείο που θα μετακινήσουμε το δείκτη θέσης.

offset: είναι το πλήθος των bytes που θα μετακινηθεί από το origin, και μπορεί να είναι είτε θετικός είτε αρνητικός αριθμός. Σε αρχεία κειμένου θα πρέπει να χρησιμοποιείται η τιμή επιστροφής της ftell.

origin: είναι το σημείο αναφοράς και μπορεί να είναι μια από τις παρακάτω σταθερές.

Όνομα σταθεράς	Περιγραφή
SEEK_SET	Έναρξη του αρχείου
SEEK_CUR	Τρέχουσα θέση
SEEK_END	Τέλος αρχείου

Η συνάρτηση επιστρέφει το 0 μετά την επιτυχημένη μετακίνηση διαφορετικά επιστρέφει μια οποιαδήποτε άλλη τιμή εκτός από 0.

Παραδείγματα	
fseek (fptr , 5 , SEEK_SET);	Μετακίνηση 5 bytes από την αρχή του αρχείου
fseek (fptr , -2 , SEEK_CUR);	Μετακίνηση πίσω 2 bytes από τη τρέχουσα θέση.
fseek (fptr , -3 , SEEK_END);	Μετακίνηση πίσω 3 bytes από το τέλος του αρχείου.

Οι συναρτήσεις fseek, rewind() και ftell (2-2)

Επιπλέον για τη μετακίνηση της θέσης του δείκτη ανάγνωσης/εγγραφής στην αρχή του αρχείου μπορούμε να χρησιμοποιούμε και τη

void rewind(FILE *fptr)

fptr: είναι ο δείκτης στο αρχείο που θα μετακινήσουμε το δείκτη θέσης στην αρχή του.

Για τη εύρεση της τρέχουσας θέσης του δείκτη σε αρχείο, χρησιμοποιούμε τη

long int ftell(FILE *fptr)

fptr: είναι ο δείκτης στο αρχείο που θα βρούμε τη θέση του δείκτη εγγραφής/ανάγνωσης.

Η συνάρτηση επιστρέφει τη τρέχουσα θέση του δείκτη και σε περίπτωση σφάλματος τη τιμή -1.

Παράδειγμα fseek και ftell

```
File size = 168
id = 9 name = Mary       surname = Yello        salary = 44127.120000
```

```c
#include <stdio.h>
#include <stdlib.h>

typedef struct {
    int id;
    char pname[20];
    char psurname[20];
    double salary;
} Person;

void ReadData() {
    FILE *fp = fopen ("citizens.dat", "rb");
    if (fp == NULL)  {
        printf("\nError opening file citizens.dat\n");
        exit (1);
    }
    fseek(fp, 0, SEEK_END);
    long int fsize = ftell(fp);
    printf("File size = %ld\n", fsize);

    fseek(fp, 2 * sizeof(Person), SEEK_SET);

    Person temp;

    if(fread( &temp, sizeof(Person), 1, fp) == 1) {
       printf ("id = %d name = %s \t surname = %s \tsalary = %lf\n",
          temp.id, temp.pname, temp.psurname, temp.salary);
   }
   fclose(fp);
}

int main(int argc, char *argv[]) {
    ReadData();
    return 0;
}
```

Μετακίνηση στο τέλος του αρχείου

Μετακίνηση μετά από δυο εγγραφές από την αρχή του αρχείου

Ανάγνωση της τρίτης εγγραφής

Η συνάρτηση fflush και setbuf (1-2)

Για τη άμεση αποθήκευση όλων των δεδομένων που βρίσκονται στον εσωτερικό πίνακα (output buffer) στο αρχείο χρησιμοποιούμε τη

int fflush(FILE *fptr)

fptr: είναι ο δείκτης στο αρχείο που θα ζητήσουμε να αποθηκευτούν άμεσα όλα τα δεδομένα.

Η συνάρτηση επιστρέφει τη τιμή 0 κατά την επιτυχημένη εκτέλεση της, διαφορετικά επιστρέφει τη τιμή EOF.

Η κλήση της συνάρτησης θα πρέπει να γίνεται με παράμετρο ένα ρεύμα εξόδου. Στο περιβάλλον των Windows υπάρχουν μεταγλωττιστές που επιτρέπουν τη χρήση της συνάρτησης με το ρεύμα εισόδου stdin για να το αδειάσουν από οποιοδήποτε χαρακτήρα, υπάρχει μέσα στο buffer του πληκτρολογίου. Προτείνεται η χρήση της διπλανής λειτουργίας.

```c
int main(void) {
    int x;
    char c;
    printf("Type an integer : ");
    scanf("%d", &x);
    flush(stdin);
    printf("Type a char : ");
    scanf("%c", &c);
    printf("\n\nYou typed %c and %d.\n\n", c, x);
    return 0;
}
```

```c
void clrkbd(){
    char ch;
    while( (ch = getchar()) != '\n');
}
```

Η συνάρτηση fflush και setbuf (1-2)

Η εγγραφή των δεδομένων από τον εσωτερικό χώρο αποθήκευσης (buffer) προς το ρεύμα εξόδου γίνεται κάθε φορά που είτε γεμίσει ο πίνακας, είτε υπάρχει ο ειδικός χαρακτήρας \n, είτε γίνει μετακίνηση του δείκτη θέσης με τις fseek, ftell, rewind, είτε κλείσει το αρχείο. Η συνάρτηση fflush ανεξαρτήτως όλων των ανωτέρω καταστάσεων αδειάζει άμεσα το περιεχόμενο του buffer προς το ρεύμα εξόδου.

Για τη αλλαγή του εσωτερικού χώρου αποθήκευσης (internal buffer) των δεδομένων σε ένα άλλο πίνακα χαρακτήρων χρησιμοποιούμε τη

void setbuf(FILE *stream, char *buffer)

fptr: είναι ο δείκτης στο αρχείο που θα αντικαταστήσουμε τον εσωτερικό χώρο αποθήκευσης

buffer: είναι ο χώρος αποθήκευσης των δεδομένων προς αποθήκευση στο αρχείο.

Η εκτέλεση της setbuf για την αλλαγή του χώρου αποθήκευσης θα πρέπει να γίνει πριν την εκτέλεση οποιασδήποτε εξόδου στο αρχείο.

Παράδειγμα setbuf και flush

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    char buf[BUFSIZ];

    setbuf(stdout, buf);

    printf("A line of output.\n");
    printf("Another line of output.\n");
    printf("1 + 2 = %d\n", 1 + 2);
    char ch = getch();

    fflush(stdout);

    return 0;
}
```

Με την εκτέλεση του προγράμματος, παρά τις εντολές εκτυπώσεις δεν εμφανίζεται τίποτα στην οθόνη και είναι στην αναμονή για να πατήσουμε ένα πλήκτρο.

Μόλις πατήσουμε ένα πλήκτρο, τότε αποθηκεύονται όλα τα δεδομένα στο αρχείο εξόδου και εμφανίζονται στην οθόνη.

```
A line of output.
Another line of output.
1 + 2 = 3
```

Η σταθερά BUFSIZ είναι το πλήθος των χαρακτήρων του εσωτερικού buffer που χρησιμοποιείται κατά την ανάγνωση ή εγγραφή σε αρχείο. Μπορείτε να προσθέσετε τη παρακάτω εντολή ώστε να δείτε τη τιμή της σταθεράς

printf("%d\n", BUFSIZ);