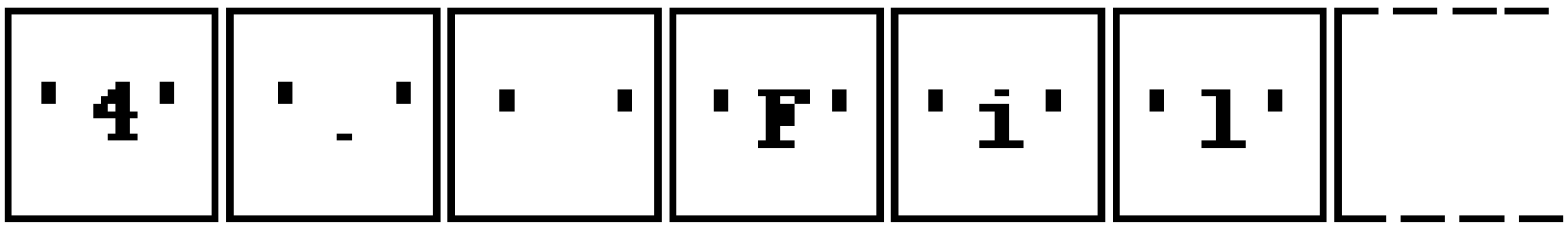
Προγραμματισμός

Αρχεία

Χρησιμότητα

- Μεγάλος όγκος δεδομένων
- Αποθήκευση μετά το σβήσιμο του ΗΥ
- Δευτερεύουσα μνήμη
- Ένα απλό αρχείο "Lecture_4"



Αρχεία

- Η C++ βλέπει κάθε αρχείο σαν μια σειρά από bytes
  - Κάθε αρχείο «κλείνει» με end-of-file marker
- Μπορούμε να ανοίξουμε ένα αρχείο για διάβασμα, είτε για γράψιμο
- Το άνοιγμα του αρχείου συνεπάγεται την επιστροφή ενός δείκτη σε FILE structure

Streams

- Σειριακή επικοινωνία
- Μονόδρομη προσπέλαση
- Διάκριση σε *output* και *input stream*.
- "cin«, πληκτρολόγιο
- "cout«, οθόνη

- Τα "cin" και "cout" είναι instances κάποιων (stream) *objects*.
- Header file "fstream".

- #include<fstream>

Creating Streams

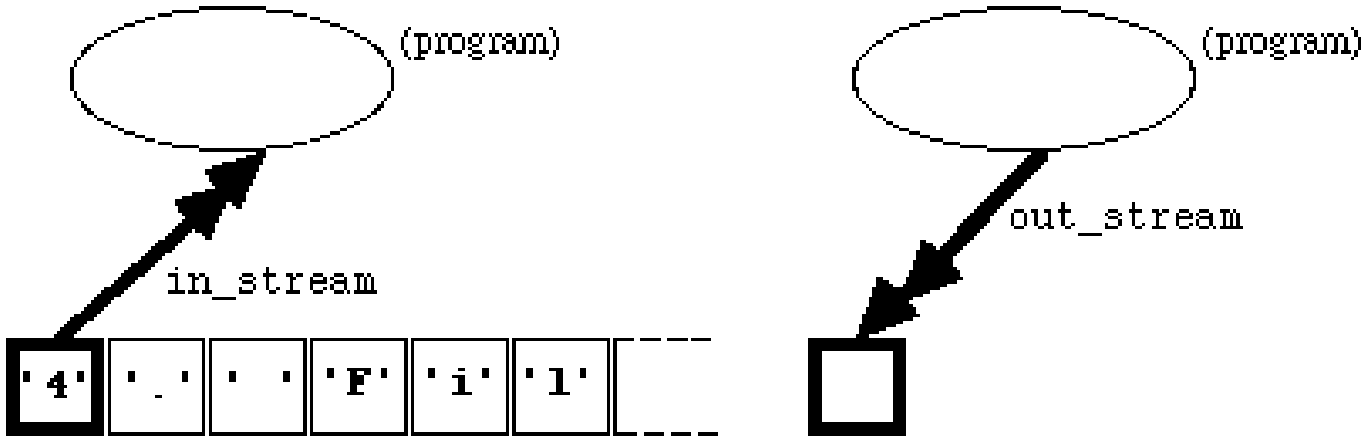
- Δήλωση μεταβλητών
- ifstream in_stream;
- ofstream out_stream;

- Είναι μεταβλητές που δείχνουν στο αρχείο (pointers).
- Το "in_stream1 = in_stream2" δεν αντιγράφει τα αρχεία.

Connecting and Disconnecting Streams to Files

Connects ifstream "in_stream" to the file "Lecture_4":
in_stream.open("Lecture_4.txt");
This connects "in_stream" to the beginning of "Lecture_4".

Connects the ofstream "out_stream" to the file "Lecture_4":
out_stream.open("Lecture_4.txt");



Although this connects "out_stream" to "Lecture_4", it also deletes the previous contents of the file, ready for new input.

Σειριακή προσπέλαση

```
 1// basic file operations
 2#include <iostream>
 3#include <fstream>
 4using namespace std;
 5
 6int main () {
 7  ofstream myfile;
 8  myfile.open ("example.txt");
 9  myfile << "Writing this to a file.\n";
10  myfile.close();
11  return 0;
12}
```

```
[file example.txt]
Writing this to a file.
```

```
int main()
{
  char str[10];

  //Creates an instance of ofstream, and opens example.txt
  ofstream a_file ( "example.txt" );
  // Outputs to example.txt through a_file
  a_file<<"This text will now be inside of example.txt";
  // Close the file stream explicitly
  a_file.close();
  //Opens for reading the file
  ifstream b_file ( "example.txt" );
  //Reads one string from the file
  b_file>> str;
  //Should output 'This'
  cout<< str <<"\n";
  b_file.close();
}
```

Include files

- **#include <ifstream>**
- Library to specifically deal with input of text files. Does not deal with output.
- **#include <ofstream>**
- Library to specifically deal with output of text files. Does not deal with input.
- **#include <fstream>**
- Library to deal with BOTH input and output.

Διαχείριση σφαλμάτων

Η κλήση in_stream.fail(); επιστρέφει True αν η προηγούμενη ενέργεια στην "in_stream" ήταν ανεπιτυχής (π.χ. προσπαθήσαμε να ανοίξουμε ανύπαρκτο αρχείο για ανάγνωση)

```
int main() {
ifstream in_stream;
in_stream.open("Lecture_4");
if (in_stream.fail())
{
        cout << "Sorry, the file couldn't be opened!\n";
        exit(1);
}
...
```

```
int main () {
        ofstream myfile;
        myfile.open ("example.txt");
        if(myfile.is_open())
        {
                myfile << "Hello world! " << endl;
                myfile.close();
        }
        else
        cout << "Can't open the file!" << endl;
}
```

ios::in	Open for input operations.
ios::out	Open for output operations.
ios::binary	Open in binary mode.
ios::ate	Set the initial position at the end of the file. If this flag is not set to any value, the initial position is the beginning of the file.
ios::app	All output operations are performed at the end of the file, appending the content to the current content of the file. This flag can only be used in streams open for output-only operations.
ios::trunc	If the file opened for output operations already existed before, its previous content is deleted and replaced by the new one.

File open modes

mode			description	starts..
r	rb		open for reading	beginning
w	wb		open for writing (creates file if it doesn't exist). Deletes content and overwrites the file.	beginning
a	ab		open for appending (creates file if it doesn't exist)	end
r+	rb+	r+b	open for reading and writing	beginning
w+	wb+	w+b	open for reading and writing. Deletes content and overwrites the file.	beginning
a+	ab+	a+b	open for reading and writing (append if file exists)	end

File open modes

```
ofstream myfile;
myfile.open ("example.bin", ios::out | ios::app | ios::binary);
```

class	default mode parameter
ofstream	ios::out
ifstream	ios::in
fstream	ios::in | ios::out

```
ofstream myfile ("example.bin", ios::out | ios::app | ios::binary);
```

Κλείσιμο Αρχείων

- **Ότι ανοίγει, πρέπει να κλείνει**
  - Το πρόγραμμα δεν πρέπει να αφήνει ανοιχτά αρχεία
    - μπορεί να μην *περάσουν* όλες οι αλλαγές σας
    - δυσχεραίνει η πρόσβαση σε αυτά από άλλα προγράμματα
- **close()**
  - *κλείνει* το αρχείο
  - Επιστρέφει 0 αν είναι επιτυχής και EOF διαφορετικά
  - Γίνεται αυτόματα με το τέλος του προγράμματος
    - *Είναι καλή πρακτική να γίνεται από τον προγραμματιστή*

myfile.close();

Text files - Output

```
 1 // writing on a text file
 2 #include <iostream>
 3 #include <fstream>
 4 using namespace std;
 5
 6 int main () {
 7   ofstream myfile ("example.txt");
 8   if (myfile.is_open())
 9   {
10     myfile << "This is a line.\n";
11     myfile << "This is another line.\n";
12     myfile.close();
13   }
14   else cout << "Unable to open file";
15   return 0;
16 }
```

```
[file example.txt]
This is a line.
This is another line.
```

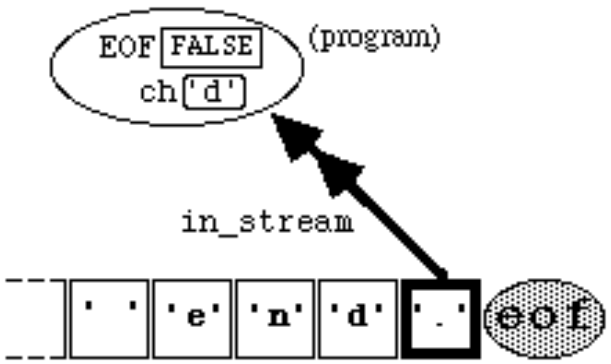
Text files - Input

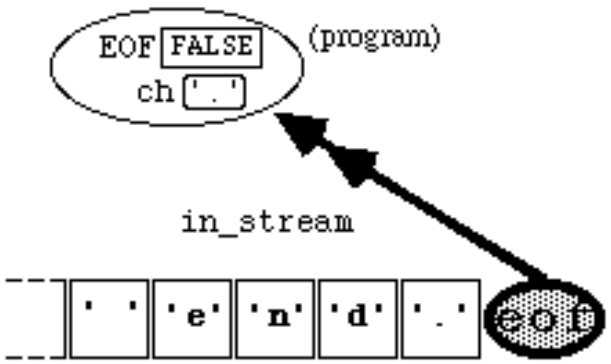
```
 1 // reading a text file
 2 #include <iostream>
 3 #include <fstream>
 4 #include <string>
 5 using namespace std;
 6
 7 int main () {
 8   string line;
 9   ifstream myfile ("example.txt");
10   if (myfile.is_open())
11   {
12     while ( myfile.good() )
13     {
14       getline (myfile,line);
15       cout << line << endl;
16     }
17     myfile.close();
18   }
19
20   else cout << "Unable to open file";
21
22   return 0;
23 }
```

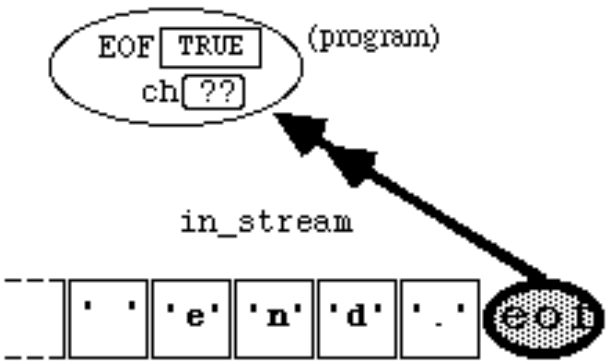
```
This is a line.
This is another line.
```

Τέλος προσπέλασης

- The boolean expression
- in_stream.eof() will now evaluate to True.

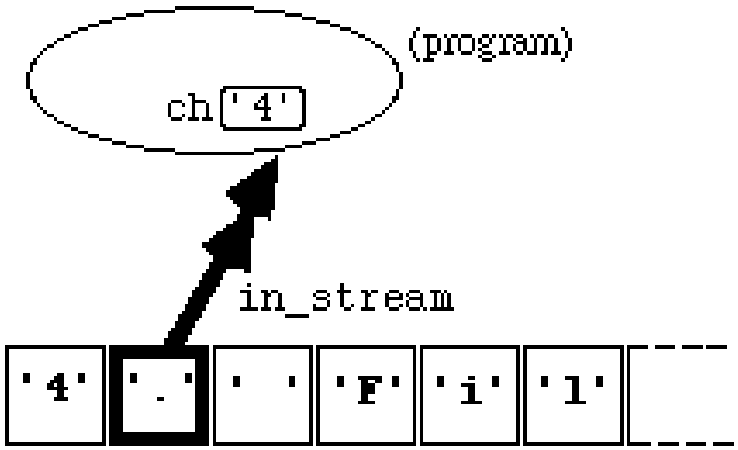






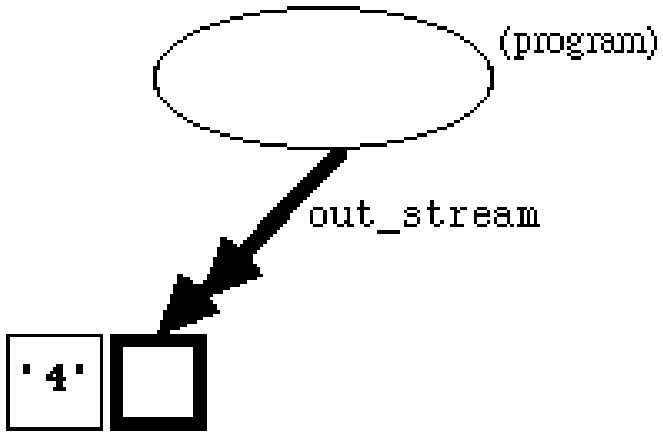
Character Input and Output

- **"get(...)"**
- This function takes a single argument of type "char".
- in_stream.get(ch) :
  - the variable "ch" is assigned the value "'4'", and
  - the ifstream "in_stream" is re-positioned so as to be ready to input the next character in the file.



Character Input and Output

- **"put(...)"**
- This function takes a single argument of type "char". We can input or *write* single characters to a file opened via an ofstream using the member function "put(...)".
- out_stream.put('4'); changes the state to:



- Αντιγραφή του "Lecture_4" σε οθόνη και στο αρχείο "Copy_of_4".

```
int main() {
char character;
ifstream in_stream;
ofstream out_stream;
in_stream.open("Lecture_4");
out_stream.open("Copy_of_4");
in_stream.get(character);
```

```
while (!in_stream.eof())
{
        cout << character;
        out_stream.put(character);
        in_stream.get(character);
}
out_stream.close();
in_stream.close();
return 0;
}
```

State flags

bad()

Returns true if a reading or writing operation fails. For example in the case that we try to write to a file that is not open for writing or if the device where we try to write has no space left.

fail()

Returns true in the same cases as bad(), but also in the case that a format error happens, like when an alphabetical character is extracted when we are trying to read an integer number.

eof()

Returns true if a file open for reading has reached the end.

good()

It is the most generic state flag: it returns false in the same cases in which calling any of the previous functions would return true.

Παράμετροι σε συναρτήσεις

```
int main()
{
    ifstream in_stream;
    ofstream out_stream;
    in_stream.open("Lecture_4");
    out_stream.open("Copy_of_4");
    copy_to(in_stream,
    out_stream);
    out_stream.close();
    in_stream.close();
    return 0;
}
```

```
void copy_to(ifstream& in,
ofstream& out)
{
    char character;
    in.get(character);
    while (!in.eof())
    {
        cout << character;
        out.put(character);
        in.get(character);
    }
}
```

Τυχαία προσπέλαση

Τρέχουσα Θέση

- Μία από τις πληροφορίες που διατηρούνται στις δομές τύπου FILE για κάθε αρχείο είναι η τρέχουσα θέση εγγραφής/ανάγνωσης
- Όταν κάνουμε εγγραφή κειμένου σε ένα αρχείο η εγγραφή γίνεται στην τρέχουσα θέση μέσα στο αρχείο.
- Όταν κάνουμε ανάγνωση κειμένου η ανάγνωση γίνεται από την τρέχουσα θέση στο αρχείο.
- Η τρέχουσα θέση αυξάνει ανάλογα με τους χαρακτήρες που έχουμε γράψει ή διαβάσει
- Η τρέχουσα θέση όταν πρωτοανοίξουμε ένα αρχείο εξαρτάται από το mode στην fopen
- Μπορούμε να αλλάξουμε την τρέχουσα θέση με την συνάρτηση fseek (αναλυτικά μετά)

Βασικές εντολές

- **fseek**() Sets the position to a desired point in the file
- **ftell**() Gives the current position in the file
- **rewind**() Sets the position to the beginning of the file

get and put stream pointers

- ifstream, like istream, has a pointer known as the *get pointer* that points to the element to be read in the next input operation
- ofstream, like ostream, has a pointer known as the *put pointer* that points to the location where the next element has to be written.
- Finally, fstream, inherits both, the get and the put pointers, from iostream (which is itself derived from both istream andostream).
- These internal stream pointers that point to the reading or writing locations within a stream can be manipulated using the following member functions:

tellg() and tellp()

- These two member functions have no parameters and return a value of the member type pos_type, which is an integer data type representing the current position of the get stream pointer (in the case of tellg) or the put stream pointer (in the case of tellp).

seekg() and seekp()

- These functions allow us to change the position of the get and put stream pointers. Both functions are overloaded with two different prototypes. The first prototype is:
  seekg (position);
  seekp (position);
  Using this prototype the stream pointer is changed to the absolute position position (counting from the beginning of the file). The type for this parameter is the same as the one returned by functions tellg and tellp: the member type pos_type, which is an integer value.

```
 1 // obtaining file size
 2 #include <iostream>
 3 #include <fstream>
 4 using namespace std;
 5
 6 int main () {
 7   long begin,end;
 8   ifstream myfile ("example.txt");
 9   begin = myfile.tellg();
10   myfile.seekg (0, ios::end);
11   end = myfile.tellg();
12   myfile.close();
13   cout << "size is: " << (end-begin) << "
14 bytes.\n";
15   return 0;
   }
```

```
size is: 40 bytes.
```

Προγραμματισμός

Δυαδικά αρχεία

- File streams include two member functions specifically designed to input and output binary data sequentially: write and read. The first one (write) is a member function of ostream inherited by ofstream. And read is a member function of istream that is inherited by ifstream. Objects of class fstream have both members. Their prototypes are:

```
write ( memory_block, size );
read ( memory_block, size );
```

```
 1 // reading a complete binary file
 2 #include <iostream>
 3 #include <fstream>
 4 using namespace std;
 5
 6 ifstream::pos_type size;
 7 char * memblock;
 8
 9 int main () {
10   ifstream file ("example.bin", ios::in|ios::binary|ios::ate);
11   if (file.is_open())
12   {
13     size = file.tellg();
14     memblock = new char [size];
15     file.seekg (0, ios::beg);
16     file.read (memblock, size);
17     file.close();
18
19     cout << "the complete file content is in memory";
20
21     delete[] memblock;
22   }
23   else cout << "Unable to open file";
24   return 0;
25 }
```

the complete file content is in memory

Προγραμματισμός

Παραδείγματα σε C

Άνοιγμα Αρχείων

- **FILE *fp;**
  - Δημιουργεί έναν **FILE** pointer με όνομα **fp**
- FILE *fopen(const char **fname*, const char **mode*);
  - Δύο ορίσματα:
    - filename: το όνομα του αρχείου που θα ανοίξει
    - mode: Καθορίζει αν το αρχείο ανοιχτεί για διάβασμα ή γράψιμο
  - Η fopen επιστρέφει έναν **FILE** pointer στο αρχείο
    - Σε περίπτωση αποτυχίας επιστρέφει NULL

```
FILE *fopen(const char *path, const char *mode);
```

Τέλος προσπέλασης

- Οι συναρτήσεις ανάγνωσης επιστρέφουν **EOF**, ή **(FILE *) NULL**, όταν φτάσουν στο τέλος του αρχείου
- Η συνάρτηση:

```
int feof ( FILE * stream );
```

- επιστρέφει 0 για τέλος αρχείου και διαφορετικά (!0), αλλιως, Π.χ.

```
while (!feof(ifp))
{
   if (fscanf(ifp, "%s %d", username, &score) != 2)
      break;
   fprintf(ofp, "%s %d", username, score+10);
}
```

Ανίχνευση λαθών

```
if ((output_file = fopen("output_file", "w")) == (FILE *) NULL)

   fprintf(stderr, "Cannot open file.\n");
```

Δημιουργία αρχείου κειμένου

```
main( )
{
        FILE *fp;
        int index;

        fp = fopen("mytestfile.txt","w"); /* open for writing */

        for (index = 1;index <= 10;index++)
                fprintf(fp,"Line number %d\n", index);

        fclose(fp); /* close the file before ending program */
}
```

Ανάγνωση και προβολή αρχείου κειμένου

```
int main( )
{
        FILE *funny;
        int c;
        funny = fopen("file2read.txt","r");
        if (funny == NULL)
        {
                printf("File doesn't exist\n");
                return 1;
        }
        do
        {
                c = getc(funny); /* get a character from the file */
                putchar(c); /* display it on the monitor */
        } while (c != EOF); /* until EOF */
        fclose(funny);
}
```

Αντιγραφή αρχείου σε άλλο

```
void filecopy(FILE *ifp, FILE *ofp)
{
     int c;
     while ((c = getc(ifp)) != EOF)
          putc(c, ofp);
}
```

```
FILE *ifp = fopen("input.txt","r");
FILE *ofp = fopen("output.txt","w");

filecopy(ifp, ofp);

fclose(ifp);
fclose(ofp);
```

Append a file

```
#include <stdio.h>
int main()
{
    FILE *fp file = fopen("file.txt","a");
    fprintf(fp,"%s","This is just an example :)");
    /*append some text*/
    fclose(fp);
    return 0;
}
```

Παράδειγμα – ανάγνωση λέξεων

υποθέτει πως 1 γραμμή <999 chars

```
void somefunction(char *file)
{
    FILE *fp;
    char s[1000];

    if ((fp = fopen(file, "r")) == NULL)
    {
        printf("Can not open the file %s\n",file);
        return;
    }

    while (fscanf(fp,"%s",s) != EOF)
    {
        printf("%s",s);
    }
    fclose(fp);
}
```

Ανάγνωση/Εγγραφή byte προς byte

- **fgetc** *(αντίστοιχη της getchar)*
  - Διαβάζει έναν χαρακτήρα από αρχείο
  - Όρισμα: **FILE** pointer
  - **fgetc(stdin)** ισοδύναμο με **getchar()**
- **fputc** *(αντίστοιχη της putchar)*
  - Γράφει έναν χαρακτήρα σε αρχείο
  - Όρισμα: **FILE** pointer και ένας χαρακτήρας για να γραφεί
  - **fputc('a', stdout)** ισοδύναμο με **putchar('a')**

Ανάγνωση/Εγγραφή γραμμή προς γραμμή

- **fgets**
  - Διαβάζει μια γραμμή από ένα αρχείο
  - *char *fgets(char *line, int maxline, FILE *fp)*
- **fputs**
  - Γράφει μια γραμμή σε ένα αρχείο
  - *int fputs(char *line, FILE *fp)*

Μορφωμένη (formated) ανάγνωση/εγγραφή

- **fscanf / fprintf**
  - Ισοδύναμες των **scanf** και **printf για αρχεία**

EOF pitfall

A common mistake when using fgetc, getc, or getchar is to assign the result to a variable of type char *before* comparing it to EOF. The following code fragments exhibit this mistake, and then show the correct approach (using type int):

Mistake	Correction

char c;
while ((c = getchar()) != EOF)
putchar(c);
	int c;
	while ((c = getchar()) != EOF)
	putchar(c);

- Σύγχηση του EOF με έναν από τους 256 χαρακτήρες
- Αλλίως: check feof and ferror after getchar returns EOF.

Standard file pointers in UNIX

FILE *stdin, *stdout, *stderr;

- automatically open to all C programs
- since these files are already open, there is no need to use fopen on them

Τρία Ειδικής Χρήσης Αρχεία

- Το λειτουργικό σύστημα, με το που αρχίζει να τρέχει το πρόγραμμά, ανοίγει τρία αρχεία και ορίζει τους δείκτες stdin, stdout, stderr να δείχνουν στις αντίστοιχες δομές FILE. Κανονικά :
  - **stdin**: αντιστοιχεί στο πληκτρολόγιο! Ότι κουμπί πατιέται θεωρείται ότι διαβάζεται από ένα αρχείο
  - **stdout**: αντιστοιχεί στην οθόνη μας. Ότι εγγράφετε σε αυτό το αρχείο εκτυπώνεται στην οθόνη
  - **stderr**: αντιστοιχεί επίσης στην οθόνη μας. Χρησιμοποιείτε για μηνύματα λάθους.

Αναδιεύθυνση (redirection) στο Unix

- Μπορούμε να ζητήσουμε από το λειτουργικό να *ανακατευθύνει* τα stdin, stdout, stderr με αρχεία στο δίσκο:
  - a.out < someinputfile
    - Το stdin δεν είναι πλέον το πληκτρολόγιο, αλλά το αρχείο somefile
  - a.out > someoutputfile
    - Το stdout δεν είναι πλέον η οθόνη αλλά το someoutputfile
  - a.out < inputfile > outputfile
  - a.out < inputfile > outputfile >& errfile

Παράδειγμα

```
#include <stdio.h>

int main()
{
    int i;
    FILE *fp;

    if ((fp = fopen("myfirstfile.txt", "w")) == NULL)
    {
        printf("Could not open file.\n");
        return 0;
    }

    fprintf(fp, "Lets write something here.\n");
    for (i = 0; i < 10; ++i)
        fprintf(fp, "%5d, %5d, %5d\n",i,i*i,i*i*i);

    fclose(fp);
    // Check out the contents of the file myfirstfile.txt
}
```

```
Lets write something here.
    0,     0,     0
    1,     1,     1
    2,     4,     8
    3,     9,    27
    4,    16,    64
    5,    25,   125
    6,    36,   216
    7,    49,   343
    8,    64,   512
    9,    81,   729
```

```
Enter the account, name, and
balance.
Enter EOF to end input.
? 100 Jones 24.98
? 200 Doe 345.67
? 300 White 0.00
? 400 Stone -42.16
? 500 Rich 224.62
?
```

```
1
2
3  #include <stdio.h>
4
5  int main()
6  {
7     int account = 1;
8     char name[ 30 ];
9     double balance;
10    FILE *cfPtr;   /* cfPtr = clients.dat file pointer */
11
12    if ( ( cfPtr = fopen( "clients.dat", "w" ) ) == NULL )
13       printf( "File could not be opened\n" );
14    else {
15       printf( "Enter the account, name, and balance.\n" );
16       printf( "Enter EOF to end input.\n" );
17       printf( "? " );
18       scanf( "%d%s%lf", &account, name, &balance );
19
20       while (account > 0) {
21          fprintf( cfPtr, "%d %s %.2f\n",
22                   account, name, balance );
23          printf( "? " );
24          scanf( "%d%s%lf", &account, name, &balance );
25       }
26
27       fclose( cfPtr );
28    }
29
30    return 0;
31 }
```

Η Βιβλιοθήκη Εισόδου/Εξόδου

- Στο stdio.h
  - Δηλώνεται ο τύπος δεδομένων FILE (με typedef) με όλες τις πληροφορίες που χρειάζονται για προσπέλαση του αρχείου από τις αντίστοιχες συναρτήσεις
  - Ορίζονται 3 αρχεία
    - stdin, stdout, stderr (θα δούμε ποια είναι αυτά σε λίγο)
  - Ορίζονται οι τιμές των σταθερών
    - NULL
    - EOF (End Of File)
  - Δηλώνονται συναρτήσεις προσπέλασης αρχείων

fread() / fwrite()

- `size_t fread(void *ptr, size_t size_of_elements, size_t number_of_elements, FILE *a_file);`

- `size_t fwrite(const void *ptr, size_t size_of_elements, size_t number_of_elements, FILE *a_file);`

Μπορεί να χρησιμοποιηθεί για κάθε μεταβλητή (ακόμη και για απλές ή στατικές με χρήση του &)

Δυαδικά αρχεία

- **fwrite**
  - «γράφει» δεδομένα από μια περιοχή μνήμης σε ένα αρχείο
- Παράδειγμα:

  fwrite(&number, sizeof(int), 1, myPtr);
  - **&number** – περιοχή μνήμης (δείκτης)
  - **sizeof(int)** – μέγεθος αντικειμένων
  - **1** – αριθμός αντικειμένων (1 για απλές μεταβλητές)
  - **myPtr** – αρχείο για μεταφορά
- **fread**
  - «διαβάζει» δεδομένα από ένα αρχείο σε μια περιοχή μνήμης

    fread(&client, sizeof (int), 1, myPtr);
  - Μπορεί να «διαβάσει» ολόκληρο τον πίνακα

Μετακίνηση του δείκτη αρχείου

- **fseek**
  - Πάει το δείκτη σε μια συγκεκριμένη θέση
  - **fseek(** *pointer, offset, symbolic_constant* **);**
    - *pointer* – file pointer
    - *offset* – θέση file pointer (0 η αρχή του αρχείου)
    - *symbolic_constant* – καθορίζει από πού αρχίζουμε να μετράμε
      - **SEEK_SET** – αρχή αρχείου
      - **SEEK_CUR** – τρέχουσα θέση
      - **SEEK_END** – τέλος αρχείου

Παράδειγμα

```
FILE *fp; fp=fopen("c:\\test.bin", "wb");
char x[10]="ABCDEFGHIJ";
fwrite(x, sizeof(x[0]), sizeof(x)/sizeof(x[0]), fp);
fclose(fp);
```

Παράδειγμα

```
int main(void)
{
    FILE *fp; size_t count;
    const char *str = "hello\n";
    fp = fopen("sample.txt", "w");
    if(fp == NULL)
    {
        perror("failed to open sample.txt");
        return EXIT_FAILURE;
    }
    count = fwrite(str, 1, strlen(str), fp);
    printf("Wrote %d bytes. fclose(fp) %s.\n", count, fclose(fp) == 0 ?
        "succeeded" : "failed");
    return EXIT_SUCCESS;
}
```

```c

#include <stdio.h>

struct clientData {
   int acctNum;
   char lastName[ 15 ];
   char firstName[ 10 ];
   double balance;
};

int main()
{
   int i;
   struct clientData blankClient = { 0, "", "", 0.0 };
   FILE *cfPtr;

   if ( ( cfPtr = fopen( "credit.dat", "w" ) ) == NULL )
      printf( "File could not be opened.\n" );
   else {

      for ( i = 1; i <= 100; i++ )
         fwrite( &blankClient,
                 sizeof( struct clientData ), 1, cfPtr );

      fclose( cfPtr );
   }

   return 0;
```

```
1  //Εγγραφή στο αρχείο «credit.dat» στοιχειών ταξινομημένα με βάση το
2
3  #include <stdio.h>
4
5  struct clientData {
6     int acctNum;
7     char lastName[ 15 ];
8     char firstName[ 10 ];
9     double balance;
10 };
11
12 int main()
13 {
14    FILE *cfPtr;
15    struct clientData client = { 0, "", "", 0.0 };
16
17    if ( ( cfPtr = fopen( "credit.dat", "r+" ) ) == NULL )
18       printf( "File could not be opened.\n" );
19    else {
20       printf( "Enter account number
21                ( 1 to 100, 0 to end input )\n? " );
22       scanf( "%d", &client.acctNum );
23
24       while ( client.acctNum != 0 ) {
25          printf( "Enter lastname, firstname, balance\n? " );
26          fscanf( stdin, "%s%s%lf", client.lastName,
27                 client.firstName, &client.balance );
28          fseek( cfPtr, ( client.acctNum - 1 ) *
29                 sizeof( struct clientData ), SEEK_SET );
30          fwrite( &client, sizeof( struct clientData ), 1,
31                  cfPtr );
32          printf( "Enter account number\n? " );
33          scanf( "%d", &client.acctNum );
34       }
35
36       fclose( cfPtr );
37       return 0;
38
39    }
```

```
Enter account number (1 to 100, 0 to end input)
? 37
Enter lastname, firstname, balance
? Barker Doug 0.00
Enter account number
? 29
Enter lastname, firstname, balance
? Brown Nancy -24.54
Enter account number
? 96
Enter lastname, firstname, balance
? Stone Sam 34.98


Enter account number
? 88
Enter lastname, firstname, balance
? Smith Dave 258.34
Enter account number
? 33
Enter lastname, firstname, balance
? Dunn Stacey 314.33
Enter account number
? 0
```