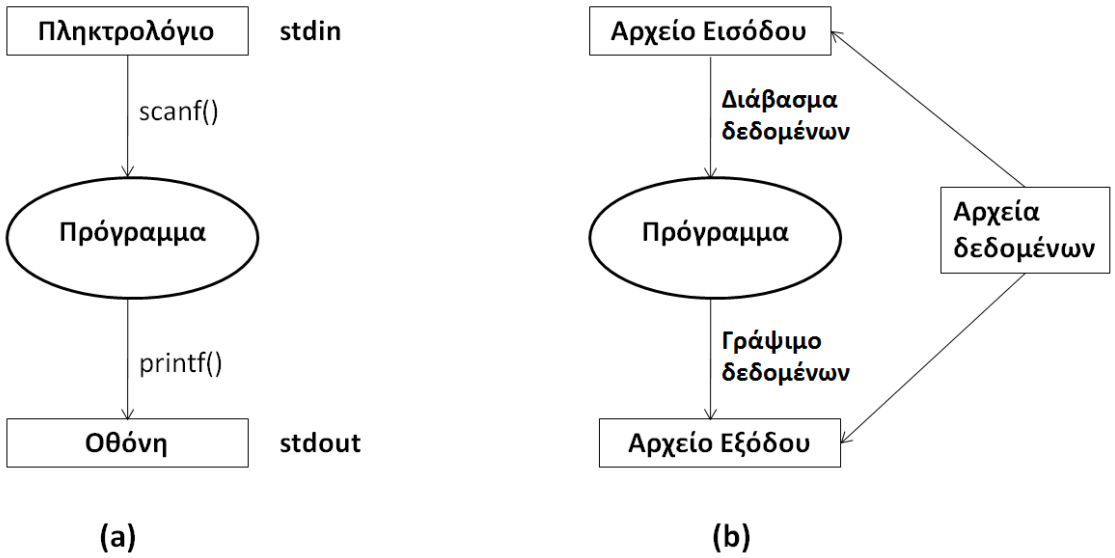
Κεφάλαιο VI:

Προσπέλαση Αρχείων.

5.1 Αρχεία δεδομένων.

Έως τώρα σε ένα πρόγραμμα έχουμε μάθει να εισάγουμε δεδομένα από το πληκτρολόγιο χρησιμοποιώντας την συνάρτηση scanf() και να εκτυπώνουμε δεδομένα στην οθόνη του υπολογιστή μας με την συνάρτηση printf(). Το πληκτρολόγιο για το πρόγραμμά μας αποτελεί την καθιερωμένη είσοδο γιαυτό και ονομάζεται **stdin** (standard input), ενώ η οθόνη την καθιερωμένη έξοδο και ονομάζεται **stdout** (standard output), όπως εικονίζεται στο σχήμα 5.1 (a).

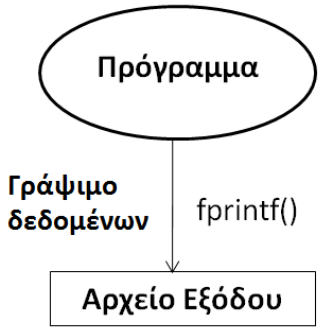
Η χρήση του πληκτρολογίου και της οθόνης ως είσοδος-έξοδος στα προγράμματά μας δεν είναι πάντοτε η ενδεδειγμένη. Σκεφθείτε εάν έχετε να εισάγετε μερικές εκατοντάδες αριθμούς στο πρόγραμμά σας, αυτό είναι πρακτικά αδύνατο να γίνει από το πληκτρολόγιο. Το ίδιο ισχύει εάν ένα πρόγραμμα παράγει πολλές γραμμές στην έξοδό του. Σε αυτή την περίπτωση η λύση είναι να χρησιμοποιήσουμε κατάλληλα **Αρχεία Δεδομένων,** τα οποία το πρόγραμμα χρησιμοποιεί ως είσοδο-έξοδο (σχήμα 5.2 (b)).



Σχήμα 5.1: a) Η καθιερωμένη είσοδος-έξοδος ενός προγράμματος. b) Τα αρχεία δεδομένων ως είσοδος-έξοδος ενός προγράμματος.

5.2 Δημιουργία αρχείου και εγγραφή δεδομένων σε αυτό.

Σε αυτή την παράγραφο θα ασχοληθούμε με το πώς μέσα από ένα πρόγραμμα μπορούμε να δημιουργήσουμε ένα νέο αρχείο και να γράψουμε μέσα σε αυτό δεδομένα. Θα ασχοληθούμε δηλαδή με το πρόβλημα που εικονίζεται στο σχήμα 5.2.



Σχήμα 5.2: Δημιουργία του αρχείου εξόδου και εγγραφή δεδομένων σε αυτό.

Ας υποθέσουμε πως θέλουμε να αναπτύξουμε ένα πρόγραμμα το οποίο να δημιουργήσει ένα αρχείο δεδομένων και να γράψει σε αυτό τα δεδομένα που περιέχει μια μεταβλητή x=3.14159. Η λύση φαίνεται παρακάτω:

```
#include<stdio.h>

int main(void){
    double x;              /* Δήλωση μεταβλητής x */
    FILE *fp;              /* Δήλωση δείκτη αρχείου με όνομα fp */
    x=3.14159;             /* Αρχικοποίηση μεταβλητής x */
    fp=fopen("arxeio.data","w");        /* Δημιουργία αρχείου arxeio.data για εγγραφή */
    fprintf(fp,"%f",x);                 /* Εγγραφή των δεδομένων στο αρχείο */
    fclose(fp);            /* Κλείσιμο του αρχείου */
    return 0;
}
```

Για την δημιουργία και την εγγραφή δεδομένων σε αρχείο απαιτούνται τα εξής βήματα:

- Το πρώτο βήμα είναι η δήλωση ενός δείκτη αρχείου. Αυτό επιτυγχάνεται με την γραμμή:
  FILE *fp;
  Ο δείκτης fp (μπορούμε να χρησιμοποιήσουμε εδώ όποιο όνομα θέλουμε σύμφωνα με τους κανόνες ονοματολογίας των μεταβλητών) είναι τύπου FILE, και μπορεί να διαχειρίζεται ένα αρχείο.
- Η δημιουργία του αρχείου γίνεται στην γραμμή:
  fp=fopen("arxeio.data","w");

Εδώ εμπλέκεται η συνάρτηση fopen() η οποία παίρνει δύο ορίσματα: Το πρώτο όρισμα είναι το όνομα του αρχείου που θέλουμε να δημιουργήσουμε, στο συγκεκριμένο παράδειγμα το όνομα που επιλέξαμε είναι το *arxeio.data*. Το δεύτερο όρισμα προσδιορίζει τι ακριβώς θέλουμε να κάνουμε με το συγκεκριμένο αρχείο, στο συγκεκριμένο παράδειγμα πρέπει να χρησιμοποιήσουμε ως δεύτερο όρισμα το "w" (αρχικό γράμμα της λέξης write) το οποίο σημαίνει πως δημιουργώ ένα νέο αρχείο το οποίο προορίζεται για εγγραφή δεδομένων. Προσοχή! Το όρισμα w θα δημιουργήσει στον σκληρό δίσκο του υπολογιστή το αρχείο με όνομα *arxeio.data.* Εάν ήδη υπάρχει αρχείο με αυτό το όνομα τα δεδομένα μέσα σε αυτό θα χαθούν. Κατά συνέπεια πρέπει να είμαστε πολύ προσεκτικοί.

- Για την εγγραφή δεδομένων μέσα στο αρχείο που δημιουργήσαμε χρησιμοποιούμε την συνάρτηση fprintf() όπως στη γραμμή:

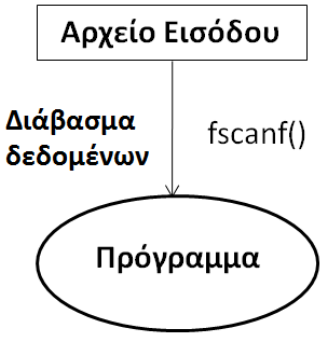
  fprintf(fp,"%f",x);

  Η σύνταξη της συνάρτησης fprintf() είναι ακριβώς η ίδια με αυτή της printf() εκτός του ότι παίρνει ένα επί πλέον όρισμα (το πρώτο) το οποίο είναι ο δείκτης του αρχείου.
- Τέλος όταν έχουμε τελειώσει πλήρως με το αρχείο μας, πρέπει να το κλείσουμε με τη χρήση της συνάρτησης fclose() όπως στη γραμμή:

  fclose(fp);

5.3 Άνοιγμα αρχείου και διάβασμα δεδομένων από αυτό.

Σε αυτή την παράγραφο θα ασχοληθούμε με το πώς μέσα από ένα πρόγραμμα μπορούμε να "ανοίξουμε" ένα αρχείο το οποίο προϋπάρχει στον σκληρό μας δίσκο αποθηκεμένο και να διαβάσουμε από αυτό δεδομένα. Θα ασχοληθούμε δηλαδή με το πρόβλημα που εικονίζεται στο σχήμα 5.3.



Σχήμα 5.3: Άνοιγμα του αρχείου εισόδου και διάβασμα δεδομένων από αυτό.

Ας υποθέσουμε πως στον σκληρό δίσκο του υπολογιστή μας προϋπάρχει το αρχείο δεδομένων με όνομα dedomena.txt και πως μέσα σε αυτό εμπεριέχεται ο αριθμός 7.689. Θέλουμε να αναπτύξουμε ένα πρόγραμμα το οποίο να "ανοίξει" το συγκεκριμένο αρχείο, να διαβάσει τον αριθμό, να τον αποθηκεύσει σε μια κατάλληλη μεταβλητή και να τον τυπώσει στην οθόνη του υπολογιστή μας. Η λύση φαίνεται παρακάτω:

```
#include<stdio.h>

int main(void){
    double y;           /* Δήλωση μεταβλητής y */
    FILE *pp;           /* Δήλωση δείκτη αρχείου με όνομα pp*/
    pp=fopen("dedomena.txt ","r");      /* Άνοιγμα αρχείου dedomena.txt για διάβασμα */
    fscanf(pp,"%lf",&y);                /* Διάβασμα των δεδομένων και αποθήκευση στην y */
    printf("y=%f \n",y);                /* Εκτύπωση στην οθόνη της μεταβλητής y */
    fclose(pp);         /* Κλείσιμο του αρχείου */
    return 0;
}
```

Για το άνοιγμα του αρχείου dedomena.txt και το διάβασμα απαιτούνται τα εξής βήματα:

- Το πρώτο βήμα είναι η δήλωση ενός δείκτη αρχείου. Αυτό επιτυγχάνεται με την γραμμή:

  FILE *pp;

  Ο δείκτης pp (μπορούμε να χρησιμοποιήσουμε εδώ όποιο όνομα θέλουμε σύμφωνα με τους κανόνες ονοματολογίας των μεταβλητών) είναι τύπου FILE, και μπορεί να διαχειρίζεται ένα αρχείο.
- Το "άνοιγμα" του αρχείου *dedomena.txt* γίνεται στην γραμμή:

  pp=fopen("dedomena.txt ","r");

  Εδώ εμπλέκεται η συνάρτηση fopen() η οποία παίρνει δύο ορίσματα: Το πρώτο όρισμα είναι το όνομα του αρχείου από το οποίο θέλουμε να διαβάσουμε, στο συγκεκριμένο παράδειγμα το όνομα του αρχείου το οποίο προϋπάρχει στον σκληρό δίσκο του υπολογιστή μας είναι το *dedomena.txt*. Το δεύτερο όρισμα προσδιορίζει τι ακριβώς θέλουμε να κάνουμε με το συγκεκριμένο αρχείο, στο συγκεκριμένο παράδειγμα πρέπει να χρησιμοποιήσουμε ως δεύτερο όρισμα το "r" (αρχικό γράμμα της λέξης read) το οποίο σημαίνει πως το αρχείο προορίζεται για ανάγνωση δεδομένων. Προσοχή! Εάν το αρχείο με όνομα *dedomena.txt* δεν υπάρχει το πρόγραμμα θα τερματιστεί παράγοντας μήνυμα λάθους.
- Για το διάβασμα δεδομένων από το αρχείο *dedomena.txt* χρησιμοποιούμε την συνάρτηση fscanf() όπως στη γραμμή:

  fscanf(pp,"%lf",&y);

  Η σύνταξη της συνάρτησης fscanf() είναι ακριβώς η ίδια με αυτή της scanf() εκτός του ότι παίρνει ένα επί πλέον όρισμα (το πρώτο) το οποίο είναι ο δείκτης του αρχείου.
- Τέλος όταν έχουμε τελειώσει πλήρως με το αρχείο μας πρέπει να το κλείσουμε με τη χρήση της συνάρτησης fclose() όπως στη γραμμή:

  fclose(pp);

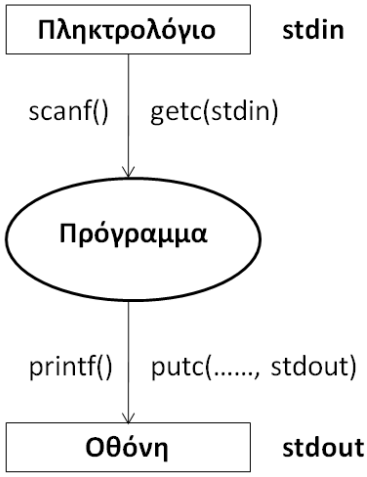
Εδώ συνοψίζοντας σχετικά με την συνάρτηση fopen(), το δεύτερο όρισμα έχει την ακόλουθη έννοια:

- w : Δημιουργία νέου αρχείου προς εγγραφή δεδομένων.
- r : "Άνοιγμα" αρχείου το οποίο προϋπάρχει προς ανάγνωση δεδομένων.

- a : “Άνοιγμα” αρχείου το οποίο προϋπάρχει προς πρόσθεση νέων δεδομένων μετά το τέλος του.

5.2 Οι συναρτήσεις putc() και getc().

Γενικά η είσοδος-έξοδος κειμένου στα προγράμματα είναι πολύ σημαντική και γιαυτό τον λόγο στη C υπάρχουν ειδικές συναρτήσεις εισόδου-εξόδου που αφορούν αποκλειστικά χαρακτήρες. Οι συναρτήσεις αυτές είναι οι **getc()** και **putc()**. Οι δύο αυτές συναρτήσεις μπορούν να χρησιμοποιηθούν είτε για είσοδο από το πληκτρολόγιο είτε για έξοδο στην οθόνη του υπολογιστή μας όπως φαίνεται στο σχήμα 5.4.



Σχήμα 5.4: Οι συναρτήσεις εισόδου-εξόδου χαρακτήρων getc() και putc().

Η σύνταξη των συναρτήσεων getc() και putc() εικονίζεται στις παρακάτω γραμμές τόσο για την είσοδο των χαρακτήρων από το πληκτρολόγιο όσο και για την έξοδο στην οθόνη. Για να γίνει κατανοητή η χρήση τους εικονίζεται δίπλα και ο αντίστοιχος κώδικας με τις συναρτήσεις scanf() και printf().

Είσοδος χαρακτήρων από το πληκτρολόγιο

Συνάρτηση getc()		Συνάρτηση scanf()
char c;		char c;
c=getc(stdin);	⇔	scanf(“%c”,&c);

Έξοδος χαρακτήρων στην οθόνη

Συνάρτηση putc()		Συνάρτηση printf()
char c;		char c;
c=’k’;		c=’k’;
putc(c,stdout);	⇔	printf(“%c”,c);

Οι συναρτήσεις getc() και putc() μπορούν να ανακατευθύνουν χαρακτήρες και σε κατάλληλα αρχεία εισόδου-εξόδου, αρκεί στην θέση των stdin και stdout να χρησιμοποιήσουμε τους αντίστοιχους δείκτες αρχείων. Ακολουθεί η σύνταξη των συναρτήσεων getc() και putc() τόσο για την είσοδο χαρακτήρων από αρχείο δεδομένων όσο και για την έξοδο.

Είσοδος χαρακτήρων από αρχείο δεδομένων

```
char  c;          /* Δήλωση μεταβλητής τύπου char */
FILE *fp;         /* Δήλωση δείκτη αρχείου με όνομα fp*/
fp=fopen("dedomena.txt","r");          /* Άνοιγμα αρχείου dedomena.txt για διάβασμα */
c=getc(fp);       /* Διάβασμα ενός χαρακτήρα από το αρχείο και αποθήκευση στην c */
fclose(fp);       /* Κλείσιμο του αρχείου */
```

Έξοδος χαρακτήρων σε αρχείο δεδομένων

```
char  c;          /* Δήλωση μεταβλητής τύπου char */
FILE *fp;         /* Δήλωση δείκτη αρχείου με όνομα fp*/
c='k';            /* Αρχικοποίηση της μεταβλητής c */
fp=fopen("arxeio.data","w");           /* Δημιουργία αρχείου arxeio.data για εγγραφή */
putc(c, fp);      /* Γράψιμο του χαρακτήρα στο αρχείο */
fclose(fp);       /* Κλείσιμο του αρχείου */
```

Σημειώνουμε εδώ πως στην περίπτωση ανακατεύθυνσης χαρακτήρων σε αρχεία εισόδου-εξόδου η C είναι εφοδιασμένη και με τις συναρτήσεις **fgetc()** και **fputc()** οι οποίες είναι ακριβώς ισοδύναμες με τις getc() και putc().

Τέλος κλείνοντας αυτή την παράγραφο θέλουμε να επισημάνουμε κάτι πολύ χρήσιμο για τα προγράμματά μας. Όταν διαβάζουμε έναν-έναν του χαρακτήρες ενός αρχείου, για να αντιληφθούμε πως έχουμε φτάσει στο τέλος του, πρέπει να ελέγχουμε για έναν ειδικό χαρακτήρα ο οποίος σημαίνει το τέλος ενός αρχείου. Ο χαρακτήρας αυτός είναι ο **EOF** (από τα αρχικά των αγγλικών λέξεων End Of File). Για παράδειγμα στις επόμενες γραμμές δίνουμε έναν εύκολο τρόπο για να διαβάσουμε όλους τους χαρακτήρες ενός αρχείου έως το τέλος του:

```
char  c;          /* Δήλωση μεταβλητής τύπου char */
FILE *fp;         /* Δήλωση δείκτη αρχείου με όνομα fp*/
fp=fopen("dedomena.txt","r");          /* Άνοιγμα αρχείου dedomena.txt για διάβασμα */

while((c=getc(fp))!=EOF){
   …….
   ……
}

fclose(fp);       /* Κλείσιμο του αρχείου */
```

Στην γραμμή: while((c=getc(fp))!=EOF){ …. } διαβάζουμε πρώτα έναν-έναν τους χαρακτήρες του κειμένου. Η εντολή while εκτελείται όσο ο χαρακτήρας που διαβάζουμε από το αρχείο δεν είναι ίσος με τον EOF.