

Παράρτημα

Συναρτήσεις
βιβλιοθήκης
της C

A



Η τυπική βιβλιοθήκη της C

Η τυπική βιβλιοθήκη της C περιλαμβάνει κάθε είδους συναρτήσεις οι οποίες είναι ομαδοποιημένες κατά κατηγορίες και δηλώνονται σε διαφορετικά αρχεία κεφαλίδας (header files). Θεωρήσαμε πιο χρήσιμο να αναφέρουμε τις συναρτήσεις ανάλογα με το αρχείο κεφαλίδας στο οποίο δηλώνονται.

Για να χρησιμοποιήσουμε οποιαδήποτε από αυτές τις συναρτήσεις, θα πρέπει με την οδηγία **#include** να συμπεριλάβουμε στον κώδικα του προγράμματος το αρχείο κεφαλίδας όπου δηλώνεται.

<ctype.h>

int islower(int c);

Η συνάρτηση **islower()** επιστρέφει μη μηδενική τιμή αν η παράμετρος **c** είναι πεζός χαρακτήρας, διαφορετικά επιστρέφει τιμή 0.

Παράδειγμα

Το επόμενο πρόγραμμα ζητάει συνέχεια χαρακτήρες μέχρι να πληκτρολογηθεί ένας κεφαλαίος χαρακτήρας.

```
#include <ctype.h>
main()
{
    char ch;
    do
    {
        ch=getch();
        putchar(ch);
    } while(islower(ch));
}
```

int isupper(int c);

Η συνάρτηση **isupper()** επιστρέφει μη μηδενική τιμή αν η παράμετρος **c** είναι κεφαλαίος χαρακτήρας, διαφορετικά επιστρέφει τιμή 0.

`int isprint(int c);`

Η συνάρτηση **isprint()** επιστρέφει μη μηδενική τιμή αν η παράμετρος **c** είναι εκτυπώσιμος χαρακτήρας (συμπεριλαμβανομένου και του διαστήματος), διαφορετικά επιστρέφει τιμή 0.

`int tolower(int c);`

Η συνάρτηση **tolower()** επιστρέφει τον αντίστοιχο πεζό χαρακτήρα της παραμέτρου **c**. Δεν υποστηρίζει ελληνικούς χαρακτήρες.

Παράδειγμα

Η

```
tolower('A')
```

επιστρέφει τον χαρακτήρα 'a'.

`int toupper(int c);`

Η συνάρτηση **toupper()** επιστρέφει τον αντίστοιχο κεφαλαίο χαρακτήρα της παραμέτρου **c**. Δεν υποστηρίζει ελληνικούς χαρακτήρες.

Παράδειγμα

Η

```
toupper('a')
```

επιστρέφει το χαρακτήρα 'A'.

<math.h>

`double exp(double x);`

Συνάρτηση **exp()** επιστρέφει τη βάση των φυσικών λογαρίθμων e υψωμένη στη δύναμη της παραμέτρου **x** (e^x).

Παράδειγμα

Η επόμενη πρόταση καταχωρίζει στη μεταβλητή **f** την τιμή e^4 .

```
f=exp(4)
```

double log(double x);

Η συνάρτηση **log()** επιστρέφει το φυσικό λογάριθμο της παραμέτρου **x**. Η συνάρτηση επιστρέφει μήνυμα λάθους αν η παράμετρος **x** είναι μικρότερη ή ίση από το 0.

Παράδειγμα

Το επόμενο πρόγραμμα εμφανίζει τους φυσικούς λογάριθμους των αριθμών από το 1 μέχρι το 10.

```
#include <math.h>
main()
{
    int i;
    for (i=1; i<=10; i++)
        printf("%d %f\n", i, log((double)i));
}
```

Παρατηρούμε τη μετατροπή¹¹ του **i** σε τύπο **double** ώστε να συμφωνεί με τον τύπο της παραμέτρου που απαιτεί η συνάρτηση **log()**.

double log10(double x);

Η συνάρτηση **log10()** επιστρέφει το δεκαδικό λογάριθμο της παραμέτρου **x**. Η συνάρτηση επιστρέφει μήνυμα λάθους αν η παράμετρος **x** είναι μικρότερη ή ίση από το 0.

double pow(double x, double y);

Η συνάρτηση **pow()** επιστρέφει την τιμή της πρώτης παραμέτρου **x** υψωμένη στη δύναμη της δεύτερης παραμέτρου **y** (x^y).

Παράδειγμα

Το επόμενο πρόγραμμα εμφανίζει τις δυνάμεις του 2 από το 0 έως το 7 ($2^0, 2^1, \dots, 2^7$).

¹¹ Με πολλούς μεταγλωττιστές η μετατροπή γίνεται αυτόματα και δεν είναι απαραίτητη η χρήση της μετατροπής τύπου (type casting).

```
#include <math.h>
main()
{
    int i;
    for(i=0;i<=7;i++)
        printf("%d %f\n",i,pow(2,i));
}
```

double sqrt(double x);

Η συνάρτηση **sqrt()** επιστρέφει ως τιμή την τετραγωνική ρίζα της παραμέτρου **x**. Η συνάρτηση επιστρέφει μήνυμα λάθους αν η παράμετρος **x** είναι μικρότερη από το 0.

Παράδειγμα

Ο επόμενος κώδικας υπολογίζει τις ρίζες της εξίσωσης β' βαθμού ax^2+bx+c .

```
d=b*b-4*a*c;
r1=(-b+sqrt(d))/(2*a);
r1=(-b-sqrt(d))/(2*a);
printf("r1=%f r2=%f\n",r1,r2);
```

double fabs(double x);

Η συνάρτηση **fabs()** επιστρέφει την απόλυτη τιμή του δεκαδικού αριθμού **x**

int abs(int x);

Η συνάρτηση **abs()** επιστρέφει την απόλυτη τιμή του ακέραιου αριθμού **x**

double sin(double x);

Η συνάρτηση **sin()** επιστρέφει το ημίτονο της γωνίας **x**. Η τιμή της **x** πρέπει να δοθεί σε ακτίνια.


```
#include <math.h>
#define PI 3.141593
main()
{
    int i;
    double rad;
    for (i=0; i<=360; i=i+10)
    {
        /* Μετατροπή του i σε ακτίνια */
        rad=2*PI*i/360;
        printf("%d %f\n", i, sin(rad));
    }
}
```

double cos(double x);

Η συνάρτηση **cos()** επιστρέφει το συνημίτονο της γωνίας **x**. Η τιμή της **x** πρέπει να δοθεί σε ακτίνια.

double tan(double x);

Η συνάρτηση **tan()** επιστρέφει την εφαπτομένη της γωνίας **x**. Η τιμή της **x** πρέπει να δοθεί σε ακτίνια.

<stdio.h>

EOF

Τιμή η οποία δηλώνει το τέλος ενός αρχείου κειμένου.

FOPEN_MAX

Τιμή η οποία δηλώνει τον μέγιστο αριθμό των ταυτόχρονα ανοιχτών αρχείων.

NULL

Τιμή η οποία δηλώνει ένα μηδενικό δείκτη — δηλαδή, δείκτη ο οποίος δε δείχνει πουθενά.

SEEK_CUR

Τιμή η οποία χρησιμοποιείται από την `fseek()` για να καθορίσει την τρέχουσα θέση, ως θέση σε σχέση με την οποία τοποθετείται ο δείκτης θέσης του αρχείου.

Παράδειγμα

Ο ακόλουθος κώδικας τοποθετεί το δείκτη θέσης του αρχείου 10 byte μετά από την τρέχουσα θέση του και διαβάζει ένα χαρακτήρα.

```
fseek(fp, 10, SEEK_CUR);  
ch=fgetc(fp);
```

βλέπε Κεφάλαιο 14

SEEK_END

Τιμή η οποία χρησιμοποιείται από την `fseek()` για να καθορίσει το τέλος του αρχείου, ως θέση σε σχέση με την οποία τοποθετείται ο δείκτης θέσης του αρχείου.

Παράδειγμα

Ο ακόλουθος κώδικας τοποθετεί το δείκτη θέσης του αρχείου 10 byte **πριν** από το τέλος του αρχείου και διαβάζει ένα χαρακτήρα.

```
fseek(fp, -10, SEEK_END);  
ch=fgetc(fp);
```

βλέπε Κεφάλαιο 14

SEEK_SET

Τιμή η οποία χρησιμοποιείται από την `fseek()` για να καθορίσει την αρχή του αρχείου, ως θέση σε σχέση με την οποία τοποθετείται ο δείκτης θέσης του αρχείου.

βλέπε Κεφάλαιο 14

stdin

Δείκτης τύπου FILE ο οποίος καθορίζει το τυπικό ρεύμα (κανάλι) εισόδου (*standard input stream*). Σε ένα κλασικό σύστημα Η/Υ, το ρεύμα αυτό αντιπροσωπεύει το πληκτρολόγιο και ανοίγει αυτόματα με την έναρξη της εφαρμογής.

stdout

Δείκτης τύπου FILE ο οποίος καθορίζει το τυπικό ρεύμα (κανάλι) εξόδου (*standard output stream*). Σε ένα κλασικό σύστημα Η/Υ, το ρεύμα αυτό αντιπροσωπεύει την οθόνη και ανοίγει αυτόματα με την έναρξη της εφαρμογής.

stderr

Δείκτης τύπου FILE ο οποίος καθορίζει το τυπικό ρεύμα (κανάλι) για την εμφάνιση των μηνυμάτων λάθους (*standard error stream*). Σε ένα κλασικό σύστημα Η/Υ, το ρεύμα αυτό αντιπροσωπεύει την οθόνη και ανοίγει αυτόματα με την έναρξη της εφαρμογής.

FILE

Τύπος δεδομένων που χρησιμοποιείται ως δείκτης για το χειρισμό αρχείων.

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

Ανοίγει ένα ρεύμα (stream) επικοινωνίας και το συνδέει με το αρχείο **filename** στην κατάσταση **mode** ή επιστρέφει τιμή NULL στην περίπτωση σφάλματος.

βλέπε Κεφάλαιο 14

FILE *freopen(char *filename, char *mode, FILE *fp);

Κλείνει ένα υπάρχον ρεύμα επικοινωνίας και το συνδέει με ένα διαφορετικό αρχείο.

Η **freopen()** προσπαθεί αρχικά να κλείσει το ρεύμα που προσδιορίζεται από το δείκτη **fp**. Είτε το κλείσει επιτυχώς είτε όχι, ανοίγει το αρχείο που προσδιορίζεται από τον δείκτη **filename** και το συνδέει με το ίδιο ρεύμα **fp**. Το αρχείο θα ανοίξει στην κατάσταση που προσδιορίζεται από το σύνολο χαρακτήρων που προσδιορίζεται από το δείκτη **mode**.

Η συνάρτηση **freopen()** είναι ιδιαίτερα χρήσιμη για την ανακατεύθυνση των ρευμάτων συστήματος **stdin**, **stdout**, και **stderr** σε άλλα αρχεία.

Παράδειγμα

Το επόμενο πρόγραμμα ανακατευθύνει το **stdout** (έξοδος στην οθόνη) στο αρχείο OUT.TXT.


```
#include <stdio.h>
main()
{
    printf("Αυτή η πρόταση εμφανίζεται στην οθόνη");
    reopen ("OUT.TXT", "w", stdout);
    printf("Αυτή η πρόταση γράφεται στο αρχείο OUT.TXT");
    fclose (stdout);
}
```

int fflush(FILE *fp);

Αποστέλλει στο ρεύμα **fp** ότι υπόλοιπο υπάρχει στην περιοχή προσωρινής αποθήκευσης (ή "ενδιάμεση μνήμη", *buffer*). Επιστρέφει EOF στην περίπτωση προβλήματος ή 0 στην περίπτωση επιτυχούς λειτουργίας.

βλέπε Κεφάλαιο 14

int fclose(FILE *fp);

Κλείνει το ρεύμα επικοινωνίας **fp** (το οποίο πιθανώς έχει συνδεθεί με κάποιο αρχείο) αφού αποστέλλει στο ρεύμα ότι υπόλοιπο υπάρχει στην περιοχή προσωρινής αποθήκευσης (ή "ενδιάμεση μνήμη", *buffer*). Επιστρέφει EOF στην περίπτωση προβλήματος ή 0 στην περίπτωση επιτυχούς λειτουργίας.

βλέπε Κεφάλαιο 14

int remove(char *filename);

Διαγράφει το αρχείο **filename** από τον δίσκο. Επιστρέφει μη μηδενική τιμή στην περίπτωση προβλήματος.

Παράδειγμα

Το επόμενο πρόγραμμα διαγράφει τα αρχεία **test.txt** και **old.bak** από τον βασικό ("ριζικό") φάκελο του C:.

```
#include <stdio.h>
main()
{
    char *p;
    p="c:\\test.txt";
    remove(p);
    remove("c:\\old.bak");
}
```

int rename(char *oldname, char *newname);

Αλλάζει το όνομα του αρχείου **oldname** με το όνομα **newname**. Επιστρέφει μη μηδενική τιμή στην περίπτωση προβλήματος.

int fprintf(FILE *fp, char *format, παράμετροι,...,...);

Αποστέλλει τις τιμές των παραμέτρων με τρόπο που καθορίζεται από το αλφαριθμητικό μορφοποίησης **format** στο ρεύμα που προσδιορίζει ο δείκτης **fp**. Αν το ρεύμα είναι συνδεδεμένο με κάποιο αρχείο, οι χαρακτήρες γράφονται μέσα στο αρχείο. Επιστρέφει ως τιμή τον αριθμό των χαρακτήρων που απέστειλε ή επιστρέφει αρνητική τιμή στην περίπτωση προβλήματος.

βλέπε Κεφάλαιο 14

int printf(char *format, ...);

Εμφανίζει στην οθόνη τις τιμές των παραμέτρων με τρόπο που καθορίζεται από το σύνολο χαρακτήρων **format**. Επιστρέφει ως τιμή τον αριθμό των χαρακτήρων που εμφάνισε ή επιστρέφει αρνητική τιμή στην περίπτωση προβλήματος. Η **printf()** είναι ισοδύναμη με την **fprintf(stdout, , ...)**.

βλέπε Κεφάλαια 3, 5, και 6

int sprintf(char *s, char *format, ...);

Όπως η **fprintf()**, με τη διαφορά ότι αποστέλλει τους χαρακτήρες εξόδου της όχι σε κάποιο ρεύμα, αλλά στο σύνολο χαρακτήρων στο οποίο δείχνει ο δείκτης **s**. Ο δείκτης **s** πρέπει να δείχνει σε χώρο ικανό να αποθηκεύσει την έξοδο της **sprintf()**. Οι χαρακτήρες τερματίζονται με τον ειδικό χαρακτήρα τερματισμού '\0'. Επιστρέφει ως τιμή τον αριθμό των χαρακτήρων που απέστειλε (χωρίς το '\0').

int fscanf(FILE *fp, char *format, διευθύνσεις,...,);

Διαβάζει δεδομένα από το ρεύμα που προσδιορίζει ο δείκτης **fp** και τα καταχωρίζει σε θέσεις μνήμης που προσδιορίζονται από τις **διευθύνσεις** τους. Ο τύπος και ο τρόπος με τον οποίο δίνονται τα δεδομένα καθορίζεται από το σύνολο χαρακτήρων **format**. Επιστρέφει τον αριθμό των δεδομένων που διαβάστηκαν ή EOF στην περίπτωση προβλήματος.

βλέπε Κεφάλαιο 14

int scanf(char *format, ...);

Διαβάζει δεδομένα από το πληκτρολόγιο και τα καταχωρίζει σε θέσεις μνήμης που προσδιορίζονται από τις **διευθύνσεις** τους. Ο τύπος και ο τρόπος με τον οποίο δίνονται τα δεδομένα καθορίζεται από το σύνολο χαρακτήρων **format**. Επιστρέφει τον αριθμό των δεδομένων που διαβάστηκαν ή την τιμή EOF στην περίπτωση προβλήματος.

Η **scanf()** είναι ισοδύναμη με την **fscanf(stdin, , ...)**.

βλέπε Κεφάλαια 3 και 6

int sscanf(char *s, char *format, ...);

Όπως η **fscanf()**, με τη διαφορά ότι διαβάζει τα δεδομένα από το σύνολο χαρακτήρων που προσδιορίζει ο δείκτης **s**.

int fgetc(FILE *fp);

Διαβάζει και επιστρέφει ως τιμή τον επόμενο χαρακτήρα από το ρεύμα εισόδου που προσδιορίζει ο δείκτης **fp**. Στην περίπτωση ανάγνωσης από αρχείο επιστρέφει τιμή EOF όταν φτάσει στο τέλος του.

βλέπε Κεφάλαιο 14

char *fgets(char *s, int n, FILE *fp);

Διαβάζει χαρακτήρες από το κανάλι εισόδου που προσδιορίζει ο δείκτης **fp** και τους αποθηκεύει στον πίνακα χαρακτήρων που προσδιορίζεται από το δείκτη **s**. Σταματάει όταν διαβαστούν **n-1** χαρακτήρες ή χαρακτήρας αλλαγής γραμμής ή όταν φτάσει στο τέλος του αρχείου. Προσθέτει το χαρακτήρα τερματισμού '\0' στους χαρακτήρες που διαβάστηκαν. Επιστρέφει ως τιμή ένα δείκτη στον πίνακα χαρακτήρων **s** και, σε περίπτωση λάθους, την τιμή **NULL**.

βλέπε Κεφάλαιο 14

int fputc(int c, FILE *fp);

Γράφει το χαρακτήρα **c** στο ρεύμα εξόδου που προσδιορίζει ο δείκτης **fp**. Επιστρέφει ως τιμή το χαρακτήρα που έγραψε και, στην περίπτωση προβλήματος, επιστρέφει EOF.

βλέπε Κεφάλαιο 14

char *fputs(char *s, FILE *fp);

Γράφει τους χαρακτήρες από τον πίνακα χαρακτήρων που προσδιορίζεται από το δείκτη **s** στο ρεύμα εξόδου που προσδιορίζεται από το δείκτη **fp**. Προσθέτει ένα χαρακτήρα αλλαγής γραμμής ('\n') στο τέλος των χαρακτήρων. Στην περίπτωση προβλήματος, επιστρέφει EOF.

βλέπε Κεφάλαιο 14

int getc(FILE *fp);

Ισοδύναμη με την `fgetc()`.

int getch();

Περιμένει να πατηθεί κάποιος χαρακτήρας από το πληκτρολόγιο (**stdin**) και τον επιστρέφει ως τιμή.

βλέπε Κεφάλαιο 5

char *gets(char *s);

Διαβάζει χαρακτήρες από το πληκτρολόγιο (**stdin**) και τους αποθηκεύει στον πίνακα χαρακτήρων που προσδιορίζεται από τον δείκτη **s**. Προσθέτει το χαρακτήρα τερματισμού '\0' στους χαρακτήρες που διαβάστηκαν. Επιστρέφει ως τιμή ένα δείκτη στον πίνακα χαρακτήρων **s** και, σε περίπτωση λάθους, την τιμή NULL. Πρέπει να αποφεύγεται επειδή δεν ελέγχει για υπερχείλιση μνήμης στον πίνακα **s**.

βλέπε Κεφάλαιο 12

int putc(int c, FILE *fp);

Ισοδύναμη με την `fputc()`.

int putch(int c);

Εμφανίζει το χαρακτήρα **c** στην οθόνη.

βλέπε Κεφάλαιο 5

int puts(char *s);

Εμφανίζει στην οθόνη (**stdout**) τους χαρακτήρες από τον πίνακα χαρακτήρων που προσδιορίζεται από τον δείκτη **s**. Προσθέτει ένα χαρακτήρα αλλαγής γραμ-

μής ('\n') στο τέλος των χαρακτήρων. Σε περίπτωση προβλήματος, επιστρέφει EOF.

βλέπε Κεφάλαιο 12

int fread(void *ptr, int size, int num, FILE *fp);

Διαβάζει από το ρεύμα εισόδου που προσδιορίζει ο δείκτης **fp**, **num** αντικείμενα (τμήματα μνήμης) μεγέθους **size** byte το καθένα και τα αποθηκεύει στο χώρο μνήμης που προσδιορίζεται από το δείκτη **ptr**. Επιστρέφει ως τιμή το πλήθος των αντικειμένων που διαβάστηκαν.

βλέπε Κεφάλαιο 14

int fwrite(void* ptr, int size, int num, FILE *fp);

Γράφει στο ρεύμα εξόδου που προσδιορίζει ο δείκτης **fp**, **num** αντικείμενα (τμήματα μνήμης) μεγέθους **size** byte το καθένα τα οποία προέρχονται από το χώρο μνήμης που προσδιορίζεται από το δείκτη **ptr**. Επιστρέφει ως τιμή το πλήθος των αντικειμένων που γράφηκαν.

βλέπε Κεφάλαιο 14

int fseek(FILE *fp, int apostasi, int thesi);

Τοποθετεί το δείκτη θέσης του αρχείου το οποίο προσδιορίζει ο δείκτης **fp**, σε μια συγκεκριμένη απόσταση (**apostasi**) σε σχέση με την τρέχουσα θέση, την αρχή, ή το τέλος του αρχείου (**thesi**).

βλέπε Κεφάλαιο 14

void rewind(FILE *fp);

Επαναφέρει το δείκτη θέσης του αρχείου το οποίο προσδιορίζει ο δείκτης **fp**, στην αρχή (θέση 0).

βλέπε Κεφάλαιο 14

int feof(FILE *fp);

Επιστρέφει μη μηδενική τιμή (αλήθεια) όταν έχουμε φτάσει στο τέλος αρχείου (end of file) του ρεύματος το οποίο προσδιορίζει ο δείκτης **fp**.

βλέπε Κεφάλαιο 14

<stdlib.h>

NULL

Τιμή η οποία δηλώνει ένα μηδενικό δείκτη, δηλαδή δείκτη ο οποίος δεν δείχνει πουθενά.

void *calloc(int num, int size);

Η συνάρτηση **calloc()** δεσμεύει από το σωρό **num** τμήματα μνήμης, μεγέθους **size** (σε byte) το καθένα. Η **calloc()** δεσμεύει ένα χώρο μνήμης όσο το γινόμενο της πρώτης παραμέτρου της (**num**) και της δεύτερης παραμέτρου της (**size**) δηλαδή **num*size** bytes. Η **calloc()** επιστρέφει ως τιμή τη διεύθυνση της πρώτης θέσης μνήμης του χώρου μνήμης που δέσμευσε. Στην περίπτωση που δεν υπάρχει αρκετή διαθέσιμη μνήμη για το μέγεθος του τμήματος η **calloc()** επιστρέφει δείκτη NULL.

βλέπε Κεφάλαιο 17

void *malloc(int size);

Η συνάρτηση **malloc()** δεσμεύει από το σωρό ένα τμήμα μνήμης, μεγέθους (σε byte) όσο η τιμή της παραμέτρου **size**. Η **malloc()** επιστρέφει ως τιμή τη διεύθυνση της πρώτης θέσης μνήμης του τμήματος που δέσμευσε. Στην περίπτωση που δεν υπάρχει αρκετή διαθέσιμη μνήμη για το μέγεθος του τμήματος η **malloc()** επιστρέφει δείκτη NULL.

βλέπε Κεφάλαιο 17

void *realloc(void *p, int size);

Η συνάρτηση **realloc()** μεγαλώνει ή μικραίνει το μέγεθος ενός ήδη δεσμευμένου τμήματος μνήμης από προηγούμενες κλήσεις συναρτήσεων δυναμικής κατανομής όπως η **malloc()** και η **calloc()**. Η πρώτη παράμετρος **p** είναι ένας δείκτης ο οποίος δείχνει στο υπάρχον δεσμευμένο τμήμα μνήμης και η παράμετρος **size** καθορίζει το νέο μέγεθος που θα αποκτήσει το τμήμα. Η συνάρτηση επιστρέφει ως τιμή τη διεύθυνση της πρώτης θέσης μνήμης του νέου τμήματος και NULL σε περίπτωση προβλήματος.

βλέπε Κεφάλαιο 17

void free(void *p);

Η συνάρτηση **free()** αποδεσμεύει από τον σωρό το τμήμα μνήμης το οποίο προσδιορίζεται από το δείκτη **p**.

βλέπε Κεφάλαιο 17

void exit(int status);

Η συνάρτηση **exit()** έχει αποτέλεσμα τον άμεσο τερματισμό του προγράμματος. Η παράμετρος **status** προσδιορίζει τον κωδικό εξόδου του προγράμματος.

βλέπε Κεφάλαιο 3

int system(char *s);

Η συνάρτηση **system()** εκτελεί στο περιβάλλον του λειτουργικού συστήματος την εντολή που βρίσκεται καταχωρισμένη στον πίνακα χαρακτήρων που προσδιορίζεται από το δείκτη **s**.

Παράδειγμα

Το επόμενο πρόγραμμα εμφανίζει τα περιεχόμενα του βασικού φακέλου του C: και περιμένει να πατηθεί κάποιο πλήκτρο για να τερματιστεί.

```
#include <stdlib.h>
main()
{
    system("dir c:\");
    system("pause");
}
```

int rand();

Επιστρέφει έναν ψευδοτυχαίο θετικό ακέραιο αριθμό.

Παράδειγμα

Το επόμενο πρόγραμμα γεμίζει τον πίνακα **a** με 100 τυχαίους αριθμούς.

```
#include <stdlib.h>
main()
{
    int i,a[100];
```

```
for (i=0; i<100; i++)  
    a[i]=rand();  
}
```

<string.h>

NULL

Τιμή η οποία δηλώνει ένα μηδενικό δείκτη, δηλαδή δείκτη ο οποίος δεν δείχνει πουθενά.

char *strcpy(char *str1, char *str2);

Η **strcpy()** αντιγράφει τον πίνακα χαρακτήρων που προσδιορίζει ο δείκτης **str2** μέσα στον πίνακα χαρακτήρων **str1**. Η συνάρτηση επιστρέφει ως τιμή ένα δείκτη στο **str1**.

βλέπε Κεφάλαιο 12

char *strncpy(char *str1, char *str2, n);

Η **strncpy()** αντιγράφει τους πρώτους **n** χαρακτήρες από τον πίνακα χαρακτήρων που προσδιορίζει ο δείκτης **str2** μέσα στον πίνακα χαρακτήρων **str1**. Η συνάρτηση επιστρέφει ως τιμή ένα δείκτη στο **str1**. Εάν το σύνολο χαρακτήρων **str2** περιέχει λιγότερους από **n** χαρακτήρες, τότε προσθέτει μέσα στο **str1** χαρακτήρες null (**'\0'**) μέχρι να συμπληρωθεί ο αριθμός **n**. Στην περίπτωση που το **str2** περιέχει περισσότερους από **n** χαρακτήρες, το σύνολο χαρακτήρων **str1** δεν θα τερματιστεί με χαρακτήρα null (**'\0'**).

char *strcat(char *str1, char *str2);

Προσθέτει στο τέλος του συνόλου χαρακτήρων που προσδιορίζει ο δείκτης **str1** το σύνολο χαρακτήρων που προσδιορίζει ο δείκτης **str2** και τερματίζει το **str1** με χαρακτήρα null (**'\0'**). Η συνάρτηση επιστρέφει δείκτη στο **str1**.

βλέπε Κεφάλαιο 12

char *strncat(char *str1, char *str2, n);

Προσθέτει στο τέλος του συνόλου χαρακτήρων που προσδιορίζει ο δείκτης **str1**, τους **n** πρώτους χαρακτήρες από το σύνολο χαρακτήρων που προσδιορίζει ο δείκτης **str2** και τερματίζει το **str1** με χαρακτήρα null (**'\0'**). Η συνάρτηση επιστρέφει δείκτη στο **str1**.

int strcmp(char *str1, char *str2);

Η συνάρτηση strcmp() συγκρίνει **αλφαβητικά** δύο σύνολα χαρακτήρων και επιστρέφει ως αποτέλεσμα:

-1 αν το **str1** είναι μικρότερο από το **str2**

0 αν το **str1** είναι ίσο με το **str2**

1 αν το **str1** είναι μεγαλύτερο από το **str2**

Οι παράμετροι **str1** και **str2** είναι δείκτες σε σύνολα χαρακτήρων.

βλέπε Κεφάλαιο 12

int strncmp(char *str1, char *str2, n);

Ίδια με την **strcmp()** με τη διαφορά ότι συγκρίνει τους **n** πρώτους χαρακτήρες. Αν κάποιο από τα σύνολα χαρακτήρων έχει λιγότερους από **n** χαρακτήρες, τότε συνεχίζει τη σύγκριση μέχρι να συναντήσει το χαρακτήρα τερματισμού σε ένα από τα δύο σύνολα χαρακτήρων.

Παράδειγμα

Το επόμενο πρόγραμμα συγκρίνει δύο σύνολα χαρακτήρων ως προς τους τρεις πρώτους χαρακτήρες τους.

```
#include <string.h>
main()
{
    char *p1,*p2;
    int res;
    p1="Παπαδόπουλος";
    p2="Παπουτσή";
    res=strncmp(p1,p2,3);
    if(res==0)
        printf("Οι 3 πρώτοι χαρακτήρες είναι ίδιοι");
    else
        printf("Οι 3 πρώτοι χαρακτήρες διαφέρουν");
}
```


char *strchr(char *str, int c);

Επιστρέφει ένα δείκτη στη θέση όπου εντόπισε για πρώτη φορά το χαρακτήρα **c** μέσα στο σύνολο χαρακτήρων που προσδιορίζεται από το δείκτη **str**. Εάν ο χαρακτήρας δεν εντοπιστεί, η συνάρτηση επιστρέφει τιμή NULL.

Παράδειγμα

Το επόμενο πρόγραμμα εντοπίζει μέσα σε μια φράση το χαρακτήρα '*'.

```
#include <string.h>
main()
{
    char lex[100];
    printf("Δώσε μια φράση:");
    gets(lex);
    if(strchr(lex, '*')==NULL)
        printf("Μέσα στη φράση δεν υπάρχει *\n");
    else
        printf("Μέσα στη φράση υπάρχει *\n ");
}
```

char *strrchr(char *str, int c);

Επιστρέφει ένα δείκτη στη θέση όπου εντόπισε για τελευταία φορά το χαρακτήρα **c** μέσα στο σύνολο χαρακτήρων που προσδιορίζεται από το δείκτη **str**. Εάν δεν εντοπίσει το χαρακτήρα, η συνάρτηση επιστρέφει τιμή NULL.

char *strstr(char *str1, char *str2);

Επιστρέφει ένα δείκτη στη θέση όπου εντόπισε για πρώτη φορά το σύνολο χαρακτήρων που προσδιορίζεται από το δείκτη **str2** μέσα στο σύνολο χαρακτήρων που προσδιορίζεται από το δείκτη **str1**. Εάν το σύνολο χαρακτήρων δεν εντοπιστεί, η συνάρτηση επιστρέφει τιμή NULL.

βλέπε Κεφάλαιο 12

int strlen(char *str);

Επιστρέφει το πλήθος των χαρακτήρων του συνόλου χαρακτήρων που προσδιορίζεται από το δείκτη **str**.

βλέπε Κεφάλαιο 12