Συναρτήσεις πρότυπης βιβλιοθήκης

Στο πρώτο μέρος δίνονται, με αλφαβητική σειρά, μερικές από τις πιο συνηθισμένες συναρτήσεις βιβλιοθήκης που συνοδεύουν τους σύγχρονους μεταγλωττιστές της C. Κάτω από το πρωτότυπο της συνάρτησης γράφεται το αρχείο επικεφαλίδας .h που πρέπει να συμπεριληφθεί για να κάνουμε χρήση της αντίστοιχης συνάρτησης και ακολουθεί σύντομη επεξήγηση της λειτουργίας της.
Στο δεύτερο μέρος δίνονται παραδείγματα της χρήσης αυτών των συναρτήσεων.
Το υλικό που παρουσιάζεται εδώ προέρχεται από την εξής πηγή: Herbert Schild "Artificial Intelligence Using C", Osborne McGraw-Hill, Berkeley, California, 1987 και υπέστη μικρές τροποποιήσεις για να είναι σύμφωνο με τα σημερινά πρότυπα της C.

1. Μερικές συνήθεις συναρτήσεις βιβλιοθήκης

Παρατηρείστε ότι συχνά οι παράμετροι εισόδου των παρακάτω συναρτήσεων έχουν τον προσδιορισμό const. Αυτό οφείλεται στο ότι θέλουμε να εξασφαλίσουμε ότι οι παράμετροι αυτές δε θα μεταβληθούν κατά την εκτέλεση της συνάρτησης. Κάθε τέτοια απόπειρα μεταβολής θα οδηγούσε σε κάποιο σφάλμα. Η ανάγκη για τον προσδιορισμό αυτό απορρέει από το γεγονός ότι οι συναρτήσεις της πρότυπης βιβλιοθήκης δεν έχουν μία και μοναδική υλοποίηση· κάθε κατασκευαστής ενός μεταγλωττιστή μπορεί να τις υλοποιήσει με διαφορετικό τρόπο. Πρέπει όμως να υπάρχει μία κοινή διασύνδεση με το χρήστη, ίδια για όλους τους μεταγλωττιστές που να εξασφαλίζει ότι οι διάφορες εκδόσεις αυτών των συναρτήσεων βιβλιοθήκης αναπτύσσονται με τέτοιο τρόπο ώστε να μην έχουμε ανεπιθύμητες παρενέργειες κατά την εκτέλεσή τους και ώστε να δίνουν το ίδιο αποτέλεσμα ανεξαρτήτως υλοποίησης.
Εξ άλλου παρατηρούμε ότι χρησιμοποιούνται και ορισμένοι "γενικευμένοι" τύποι όπως size_t που είναι στην πραγματικότητα "ψευδώνυμα" άλλων τύπων τα οποία ορίζονται σε κάποιο αρχείο επικεφαλίδας μέσω της εντολής typedef. Αυτό γίνεται επειδή δεν έχει τόση σημασία ποιον ακριβώς τύπο θα επιλέξει ο κατασκευαστής του κάθε μεταγλωττιστή για να υλοποιήσει αυτές τις συναρτήσεις, αλλά αυτό που έχει σημασία είναι η συνεπής χρήση αυτού του τύπου σε κάποιες από αυτές.

int atoi(const char *p)

#include <stdlib.h>

Μετατρέπει ένα αλφαριθμητικό που περιέχει χαρακτήρες αριθμητικών ψηφίων σε μια ακέραια τιμή, π.χ. το "123" θα μετατραπεί στην τιμή 123. Η παράμετρος είναι δείκτης char που δείχνει στην αρχή του αλφαριθμητικού που θέλουμε να μετατρέψουμε. Αν το αλφαριθμητικό δεν περιέχει μια έγκυρη αριθμητική τιμή τότε επιστρέφεται 0. Γενικά, τα κενά και οι στηλοθέτες (tabs) αγνοούνται ενώ μπορούμε να έχουμε και πρόσημο στον αριθμό που παριστάνει το αλφαριθμητικό.

int fclose(FILE *fp)

#include <stdio.h>

Γράφει ο,τι δεδομένα υπάρχουν στο χώρο προσωρινής αποθήκευσης (buffer) και κλείνει το αρχείο. Το αρχείο πρέπει να έχει ανοιχτεί πιο πριν με την fopen( ). Η παράμετρος fp είναι ο δείκτης σε αρχείο που επέστρεψε η fopen( ). Αν η fclose( ) εκτελεστεί επιτυχώς επιστρέφει 0 αλλιώς επιστρέφει -1.

FILE *fopen(const char *name, const char *mode)

#include <stdio.h>

Ανοίγει ένα αρχείο για διεργασίες εισόδου/εξόδου με χρήση του χώρου προσωρινής αποθήκευσης (buffer). Η παράμετρος name είναι το όνομα του αρχείου ενώ η παράμετρος mode είναι ένα αλφαριθμητικό που προσδιορίζει τον τρόπο προσπέλασης του αρχείου. Τυπικές τιμές της mode είναι οι εξής:

"r" ανοίγει το αρχείο μόνο για ανάγνωση
"w" ανοίγει το αρχείο μόνο για γράψιμο
"a" ανοίγει το αρχείο για προσθήκη εγγραφών στο τέλος
"rw" ανοίγει το αρχείο τόσο για ανάγνωση όσο και για γράψιμο.

Αν ανοίξουμε ένα αρχείο με "w" τότε οποιοδήποτε προϋπάρχον αρχείο με το ίδιο όνομα θα σβηστεί. Αν δε θέλουμε να συμβεί κάτι τέτοιο χρησιμοποιούμε "a". Η fopen( ) επιστρέφει ένα δείκτη τύπου FILE στο αρχείο που ανοίγει αν εκτελεστεί επιτυχώς, αλλιώς επιστρέφει NULL.

int getc(FILE *fp)

#include <stdio.h>

Επιστρέφει τον επόμενο χαρακτήρα από το αρχείο στο οποίο δείχνει ο δείκτης fp.

int getchar(void)

#include <stdio.h>

Επιστρέφει τον επόμενο χαρακτήρα από την πρότυπη είσοδο (συνήθως το πληκτρολόγιο).

char *gets(char *s)

#include <stdio.h>

Διαβάζει ένα αλφαριθμητικό από το πληκτρολόγιο και το τοποθετεί στον πίνακα χαρακτήρων s. Η είσοδος τερματίζεται είτε με χαρακτήρα επιστροφής (carriage return, \r – βασικά, όταν πατάμε “Enter”) είτε με EOF. Κανένας όμως από τους δύο αυτούς χαρακτήρες τερματισμού δε γίνεται μέρος του αλφαριθμητικού s. Το αλφαριθμητικό τερματίζεται με τον “κενό χαρακτήρα” \0. Η gets( ) επιστρέφει ένα δείκτη στο s ή NULL αν ο μεταγλωττιστής συναντήσει σφάλμα ή EOF. Η gets( ) επιτρέπει τη χρήση των πλήκτρων backspace και tab.

void *malloc(size_t size)

#include <stdlib.h>

Δεσμεύει έναν αριθμό, size, χαρακτήρων (bytes) διαθέσιμης μνήμης και επιστρέφει ένα δείκτη στην αρχή αυτής της περιοχής. Αν αυτή η ενέργεια αποτύχει, π.χ. λόγω ανεπάρκειας σε διαθέσιμη μνήμη, επιστρέφει NULL. Είναι καλό, όταν τη χρησιμοποιούμε, να ελέγχουμε πάντα αν η δέσμευση της μνήμης ήταν επιτυχής.

int printf(const char *control, arglist)

#include <stdio.h>

Επιτρέπει γενικευμένες λειτουργίες εμφάνισης δεδομένων διαφόρων τύπων σε ποικίλες μορφές. Παίρνει ως ορίσματα ένα αλφαριθμητικό και μία λίστα από μεταβλητές ή σταθερές διαφόρων τύπων. Το “αλφαριθμητικό ελέγχου” control αποτελείται από δύο ειδών στοιχεία: χαρακτήρες που θα εμφανιστούν στην οθόνη (π.χ. για να γράψουμε ένα μήνυμα) και προδιαγραφές εμφάνισης ή μορφοποίησης που κάθε μία αντιστοιχεί και σε ένα όρισμα από τη λίστα ορισμάτων arglist και καθορίζει τον τρόπο εμφάνισης αυτού του ορίσματος. Οι προδιαγραφές μορφοποίησης αποτελούνται από το χαρακτήρα % ακολουθούμενο από άλλον ένα χαρακτήρα (ή και περισσότερους). Οι πιο συνήθεις προδιαγραφές είναι οι εξής:

%c εμφάνιση ορισμάτων τύπου char
%d εμφάνιση ορισμάτων τύπου int στο δεκαδικό
%u εμφάνιση απρόσημου δεκαδικού
%o εμφάνιση απρόσημου οκταδικού
%x εμφάνιση απρόσημου δεκαεξαδικού
%f εμφάνιση ορισμάτων τύπου float
%e για δεδομένα κινητής υποδιαστολής (float, double...) με επιστημονική σημειογραφία
%g Χρησιμοποιεί όποιο από τα %e και %f είναι συντομότερο
%lf εμφάνιση ορισμάτων τύπου double
%s εμφάνιση αλφαριθμητικών
%% εμφάνιση του ίδιου του χαρακτήρα %.

Αν βάλουμε έναν ακέραιο μεταξύ του % και της προδιαγραφής, αυτός δρα ως προσδιοριστικό ελάχιστου πλάτους πεδίου (δηλαδή πόσες θέσεις θα πιάσει το αντίστοιχο όρισμα). Έτσι, η έξοδος θα συμπληρωθεί με κενά διαστήματα ή μηδενικά για να εξασφαλιστεί ότι το αντίστοιχο πεδίο θα έχει τουλάχιστον αυτό το πλάτος. Αν το όρισμα καταλαμβάνει περισσότερες θέσεις θα τυπωθεί ολόκληρο. Ο προκαθορισμένος (default) χαρακτήρας συμπλήρωσης είναι το κενό. Αν θέλουμε μηδενικά, πρέπει να βάλουμε και ένα 0 πριν το προσδιοριστικό ελάχιστου πλάτους, π.χ. το %05d θα τυπώσει μία μεταβλητή με τιμή 123 ως 00123.
Για να προσδιορίσουμε τον αριθμό δεκαδικών ψηφίων όταν έχουμε float, double κλπ, βάζουμε μία υποδιαστολή μετά από το προσδιοριστικό ελάχιστου πλάτους και μετά τον αριθμό των δεκαδικών ψηφίων που θέλουμε, π.χ. %10.4f θα τυπώσει έναν αριθμό με τουλάχιστον 10 ψηφία από τα οποία τα 4 θα είναι δεκαδικά. Αυτό δουλεύει για διάφορους τύπους μεταβλητών και επειδή δεν επιτρέπονται παραπάνω δεκαδικά από όσα προσδιορίσαμε μπορούμε να επωφεληθούμε για να βάλουμε και άνω όριο στο πλάτος ορισμάτων που τυπώνουμε. Π.χ. %5.7s θα τυπώσει ένα αλφαριθμητικό με τουλάχιστον 5 και όχι πάνω από 7 χαρακτήρες. Αν είναι μεγαλύτερο θα αποκοπεί.
Η προκαθορισμένη στοίχιση είναι στα δεξιά. Αν το πεδίο έχει καθορισμένο πλάτος μεγαλύτερο από τα δεδομένα προς εμφάνιση, αυτά θα πάνε στη δεξιά άκρη του πεδίου. Αν θέλουμε να στοιχιστούν αριστερά μπορούμε να βάλουμε ένα μείον μετά από το %, π.χ. %-10.2f

int putc(int ch, FILE *fp)

#include <stdio.h>

Γράφει ένα χαρακτήρα σε ένα αρχείο που έχει ανοιχτεί με την εντολή fopen( ) η οποία επέστρεψε το δείκτη fp. Επιστρέφει το ch αν εκτελεστεί επιτυχώς και EOF αν φτάσει στο φυσικό τέλος του αρχείου δείχνοντας ότι δε μπορεί να γίνει άλλη εγγραφή.

int scanf(const char *control, arglist)

#include <stdio.h>

Επιτρέπει γενικευμένες λειτουργίες ανάγνωσης από το πληκτρολόγιο δεδομένων διαφόρων τύπων σε ποικίλες μορφές. Αυτά τοποθετούνται, κατάλληλα μορφοποιημένα, στις μεταβλητές που περιέχονται στη λίστα ορισμάτων arglist. Κάθε τέτοιο όρισμα πρέπει να είναι δείκτης, δηλαδή να υποδηλώνει μία διεύθυνση μνήμης. Η scanf είναι περίπου η αντίστροφη της printf.
Το αλφαριθμητικό ελέγχου αποτελείται από προδιαγραφές μορφοποίησης με ένα % μπροστά, οι οποίες είναι αυτές που αναφέρονται και στη συνάρτηση printf. Αν, μετά από το % υπάρχει ένα *, τότε δεν αποδίδεται η τιμή, αλλά την προσπερνάει και πάει στο επόμενο πεδίο. Ο,τι άλλοι χαρακτήρες υπάρχουν στο αλφαριθμητικό ελέγχου ταιριάζονται με την είσοδο και αγνοούνται.

%c διαβάζει ένα χαρακτήρα
%d διαβάζει ένα int στο δεκαδικό
%h διαβάζει ένα short
%o διαβάζει ένα οκταδικό
%x διαβάζει ένα δεκαεξαδικό
%f διαβάζει ένα float
%e διαβάζει ένα float
%s διαβάζει ένα αλφαριθμητικό
%% προσπερνάει ένα χαρακτήρα %.

Όπως και στην printf, οι προδιαγραφές μορφοποίησης αντιστοιχούν μία προς μία με τις μεταβλητές εισόδου. Όλες οι μεταβλητές πρέπει να περνιούνται μέσω διεύθυνσης (με το & αν δεν είναι οι ίδιες δείκτες). Αν διαβάζουμε ένα αλφαριθμητικό, επειδή το όνομά του είναι ήδη δείκτης στην πρώτη θέση του, δε χρειάζεται το &.
Αν βάλουμε έναν ακέραιο μετά το % τότε αυτός είναι προσδιοριστικό *μέγιστου* πλάτους πεδίου, δηλαδή η εντολή

```
scanf("%20s", address)
```

θα διαβάσει μόνο τους 20 πρώτους χαρακτήρες του αλφαριθμητικού address. Αν καλέσουμε πάλι

τη scanf με το ίδιο ή κάποιο άλλο αλφαριθμητικό, τότε θα διαβαστούν και οι υπόλοιποι χαρακτήρες του address επειδή έχουν μείνει στο χώρο προσωρινής αποθήκευσης (buffer).
Όταν χρησιμοποιούμε προδιαγραφές %c για να διαβάζουμε χαρακτήρες τότε διαβάζονται επίσης και τα κενά διαστήματα, tabs και αλλαγές γραμμής που δίνουμε στο πληκτρολόγιο, ως χαρακτήρες. Π.χ. αν γράψουμε A B τότε η εντολή

```
printf("%c%c%c", &a, &b, &c);
```

θα δώσει A στο a, κενό διάστημα στο b και B στο b.
Αν χρησιμοποιούμε άλλες προδιαγραφές, τότε τα κενά διαστήματα, tabs και αλλαγές γραμμής μπορούν να χρησιμεύσουν για το διαχωρισμό των πεδίων εισόδου.
Προσοχή! Αν το αλφαριθμητικό ελέγχου περιέχει και αυτό κενά διαστήματα, tabs και αλλαγές γραμμής, τότε αυτά θα ταιριάζουν με την είσοδο και θα προσπερνιέται ότι τους αντιστοιχεί. Π.χ. στην παρακάτω εντολή

```
printf("%s ", str);
```

μετά από την προδιαγραφή %s υπάρχει ένα κενό διάστημα. Έτσι, όταν δώσουμε το αλφαριθμητικό που θέλουμε να μπει στο str και πατήσουμε Enter, δε θα συνεχιστεί το πρόγραμμα γιατί θα περιμένει έναν ακόμη χαρακτήρας (τον οποίο βέβαια θα προσπεράσει). Επειδή το αλφαριθμητικό ελέγχου της scanf δε χρησιμεύει για έξοδο μηνυμάτων στην οθόνη, όλες οι προτροπές προς το χρήστη δίνονται πριν από τη scanf με τη βοήθεια μιας printf.

char *strcat(char *s1, const char *s2)
#include <string.h>
Ενώνει το αλφαριθμητικό s1 με το s2. Το δεύτερο προσαρτάται στο τέλος του πρώτου (καταργώντας τον αντίστοιχο χαρακτήρα \0) και επιστρέφεται ένας δείκτης στην αρχή του πρώτου.

char *strcmp(const char *s1, const char *s2)
#include <string.h>
Αν τα αλφαριθμητικά s1 και s2 είναι ίδια, επιστρέφει 0. Αν το πρώτο είναι λεξικογραφικά μεγαλύτερο επιστρέφει θετικό αριθμό. Αλλιώς επιστρέφει αρνητικό αριθμό.

char *strcpy(char *s1, const char *s2)
#include <string.h>
Αντιγράφει το δεύτερο αλφαριθμητικό στο πρώτο και επιστρέφει ένα δείκτη στο πρώτο.

size_t strlen(const char *s)
#include <string.h>
Επιστρέφει τον αριθμό των χαρακτήρων του αλφαριθμητικού (εκτός από το χαρακτήρα τερματισμού, \0).

int tolower(int ch)
#include <ctype.h>
Μετατρέπει το κεφαλαίο σε μικρό. Αν ο χαρακτήρας ch είναι ήδη μικρό γράμμα το αφήνει ως έχει

int toupper(int ch)
#include <ctype.h>
Μετατρέπει το μικρό σε κεφαλαίο. Αν ο χαρακτήρας ch είναι ήδη κεφαλαίο γράμμα το αφήνει ως έχει.

2. Παραδείγματα
2.1 Είσοδος αριθμού από το πληκτρολόγιο και μετατροπή σε ακέραιο:

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

```
int main() {
    register int n;
    char s[80];
    printf("Enter a number: ");
    scanf("%s",s);
    n = atoi(s);
}
```

2.2 Άνοιγμα αρχείου, γράψιμο ενός χαρακτήρα και κλείσιμο:

```
#include <stdio.h>
int main() {
    FILE *fp;
    if((fp=fopen("test", "w"))==0) {
        printf("cannot open file.\n");
        return 1;
    }
    putc('A', fp); /* write char 'A' */
    fclose(fp);
}
```

2.3 Συνάρτηση που ανοίγει αρχείο για διάβασμα και γράψιμο. Το όνομα αρχείου περνιέται ως όρισμα στη συνάρτηση που, αν πετύχει το άνοιγμα, επιστρέφει το δείκτη του αρχείου:

```
#include <stdio.h>
FILE op_file(char *name) {
    FILE *fp;
    if((fp=fopen(name,"rw"))==0)
        printf("cannot open file");
    return fp;
}
```

2.4 Το ακόλουθο πρόγραμμα ανοίγει ένα αρχείο και διαβάζει έναν-έναν τους χαρακτήρες μέχρι να βρει τον χαρακτήρα EOF. Κάθε χαρακτήρας που διαβάζεται θα τυπωθεί στην οθόνη:

```
#include <stdio.h>
int main() {
    FILE *fp;
    char ch;
    if((fp=fopen("test", "r"))==0 {
        printf("cannot open file");
        return 1;
    }
    do {
         ch = getc(fp);
         putchar(ch);
    } while (ch != EOF);
    fclose(fp);
}
```

2.5 Συνάρτηση που κάνει ανάγνωση ενός αλφαριθμητικού με ψηφία και επιστρέφει την αριθμητική

τους τιμή:

```
#include <stdio.h>
#include <ctype.h>
int get_num() {
    char s[80], *temp;
    temp = s; /* get address of first character in s */
    do {
        *temp = getchar(); /* read a digit */
        if(isdigit(*temp)) /* only count when temp actually
            temp++;        represents a digit */
    } while (*(temp-1) != '\r'); /* until return */
    *temp = '\0'; /* null terminate */
    return (atoi(s));
}
```

2.6 Ανάγνωση ενός αλφαριθμητικού από το πληκτρολόγιο και εμφάνισή του στην οθόνη:

```
#include <stdio.h>
#include <string.h>
int main() {
    char s[80];
    register int n;
    printf("enter a string: ");
    scanf("%s", s);
    for(t = strlen(s)-1; t >= 0; --t)
        putchar(s[t]);
}
```

2.7 Πρόγραμμα που δεσμεύει 80 bytes μνήμης, τα χρησιμοποιεί για κάποιο σκοπό και μετά τα ελευθερώνει:

```
#include <stdio.h>
#include <stdlib.h>
int main() {
    char *p;
    p = malloc(80);
    if(!p) {
        printf("out of memory \n");
        exit(2);
    }
    .
    .
    .
    free(p);
}
```

2.8 Συνάρτηση που γράφει ένα αλφαριθμητικό σε ένα ορισμένο αρχείο:

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

```
void wr_string(char *s, FILE *fp) {
    while(*s)
        if(putc(*s++,fp)==EOF) {
            printf("end-of-file");
            return;
        }
}
```

2.9 Πρόγραμμα που διαβάζει αλφαριθμητικό και ακέραια τιμή από το πληκτρολόγιο:

```
#include <stdio.h>
int main() {
    int x;
    char s[80];
    printf("enter a string and an integer: ");
    scanf("%s%d",s,&x);
}
```

2.10 Εκτύπωση μηνύματος που προέρχεται από τη συνένωση δύο αλφαριθμητικών:

```
#include <stdio.h>
#include <string.h>
int main() {
    char *first[20], second[10];
    strcpy(first,"hello");
    strcpy(second,"world");
    strcat(s1,s2);
    printf(s1);
}
```

2.11 Απλή συνάρτηση επιβεβαίωσης συνθηματικού εισόδου (password):

```
#include <stdio.h>
#include <string.h>
int password( void ) {
    char s[80];
    printf("enter password: ");
    scanf("%s",s);
    if(strcmp(s,"pass")) {
        printf("invalid password\n");
        return 0;
    }
    return 1;
}
```