Προγραμματισμός σε C

Σημειώσεις

7 Συναρτήσεις

ΤΕΙ Θεσσαλίας

Λάρισα, 2019

1 Τι είναι συνάρτηση

Συνάντηση είναι ένα αυτόνομο κομμάτι κώδικα το οποίο υλοποιεί ολοκληρωμένα κάποια συγκεκριμένη εργασία. Αυτός ο κώδικας μπορεί να χρειάζεται κάποια δεδομένα για να δουλέψει - τα αποκαλούμε **παραμέτρους** - και μπορεί να υπολογίζει και κάτι το οποίο **μπορεί να το επιστρέψει** στην συνάρτηση που την κάλεσε.

Η **κύρια συνάρτηση** κάθε προγράμματος είναι η συνάντηση **main()**, την οποία καλεί το λειτουργικό σύστημα του υπολογιστή μας όταν θέλουμε να τρέξουμε το πρόγραμμα μας. Γενικά, υπάρχουν **2 είδη συναρτήσεων**:

- **έτοιμες συναρτήσεις**: τον κωδικά τους τον έχει γράψει κάποιος άλλος προγραμματιστής και βρίσκεται ενσωματωμένος σε κάποια **βιβλιοθήκη**. Για να την χρησιμοποιήσουμε θα πρέπει να εισάγουμε την βιβλιοθήκη στο πρόγραμμα μας με την εντολή **#include <...>**. Για παράδειγμα, εάν θέλουμε να εκτυπώσουμε κάτι στην οθόνη χρειαζόμαστε την συνάρτηση printf() η οποία ανήκει στην βιβλιοθήκη stdio.h. Για να την χρησιμοποιήσουμε θα πρέπει στην αρχή του προγράμματος μας να γράψουμε

  #include <stdio.h>

- **δικές μας συναρτήσεις**: είναι συναρτήσεις για τις οποίες θα πρέπει να γράψουμε εμείς τον κωδικά τους. Κάθε τέτοια συνάρτηση πρέπει να την εισάγουμε στα έξης 3 σημεία του προγράμματος μας:

  1. **πρωτότυπο συνάρτησης**: το γράφουμε πριν την main() και αποτελείται από τον τύπο επιστροφής, τον όνομα συνάρτησης κ μέσα σε παρενθέσεις τον τύπο των ορισμάτων που χρειάζεται η συνάρτηση για να δουλέψει. Εάν δεν χρειάζεται ορίσματα οι παρενθέσεις είναι κενές κ εάν δεν επιστρέφει κάτι τότε ο τύπος επιστροφής είναι void. Στο τέλος έχει πάντα τον χαρακτήρα ; (ελληνικό ερωτηματικό).

  2. **δήλωση συνάρτησης**: την γράφουμε πάντα μετά την main() και είναι το ίδιο με το πρωτότυπο άλλα βάζουμε τα ονόματα των παραμέτρων (εάν έχει), βγάζουμε το ερωτηματικό και ανοίγουμε άγκιστρα. Μέσα στα άγκιστρα γράφουμε τον κωδικά της συνάρτησης.

  3. **κλήση συνάρτησης**: όταν θέλουμε στην main() να καλέσουμε (στην ουσ ία να χρησιμοποιήσουμε) την συνάρτηση τότε θα πρέπει να την καλέσουμε στο σημείο που πρέπει. Κατά την κλήση της συνάρτησης περνάμε τιμές στις εάν υπάρχουν παράμετροι (χωρίς τον

τύπο) και εάν η συνάρτησα επιστρέφει κάποια τιμή την εκχωρούμε σε μεταβλητή για να μπορέσουμε να την χρησιμοποιήσουμε στην συνεχεία του κωδικά μας.

Παράδειγμα

Έστω ότι θέλαμε να υπολογίσουμε ένα πρόγραμμα στο οποίο ο χρήστης θα δίνει 2 ακέραιους και το πρόγραμμα θα έβρισκε και θα εκτύπωνε τον μεγαλύτερο τους. Ας δούμε το πρόγραμμα αρχικά χωρίς επιπλέον συναρτήσεις και στη συνεχεία με συναρτήσεις.

Κώδικας χωρίς επιπλέον συναρτήσεις

```c
#include <stdio.h>

int main() {

    int a, b, max;

    printf("Give a:");
    scanf("%d", &a);

    printf("Give b:");
    scanf("%d", &b);


    if (a > b)
        max = a;
    else
        max = b;

    printf("Max: %d\n\n", max);


    return 0;
}
```

Αρχικά θα μπορούσαμε να εισάγουμε μια συνάρτηση η οποία θα ζητάει έναν ακέραιο από τον χρήστη και θα το επιστρέφει στο πρόγραμμα. Αυτή η συνάρτηση θα κληθεί 2 φορές μιας κ χρειαζόμαστε 2 ακέραιους από τον χρήστη. Ο κώδικας θα είναι ο έξης:

```c
#include <stdio.h>

int insertNumber();

int main() {

    int a, b, max;

//    printf("Give a:");
//    scanf("%d", &a);
    a = insertNumber();

//    printf("Give b:");
//    scanf("%d", &b);
    b = insertNumber();


    if (a > b)
        max = a;
    else
        max = b;

    printf("Max: %d\n\n", max);


    return 0;
}

int insertNumber() {
    int num;

    printf("Give a number:");
    scanf("%d", &num);

    return num;
}
```

Στη συνεχεία, θα μπορούσαμε να εισάγουμε ακόμη 2 συναρτήσεις:

• findMax(): παιρνει 2 ακέραιους και βρίσκει ποιος είναι ο μεγαλύτερος. Στο τέλος τον επιστρέφει.

• printMax(): παιρνει ως παράμετρο έναν ακέραιο και τον εκτυπώνει στην οθόνη.

Το πρόγραμμα μας θα ήταν το έξης:

```c
#include <stdio.h>

int insertNumber();
int findMax(int, int);
void printMax(int);

int main() {

    int a, b, max;

    a = insertNumber();

    b = insertNumber();

    max = findMax(a, b);

    printMax(max);

    return 0;
}

int insertNumber() {
    int num;

    printf("Give a number:");
    scanf("%d", &num);

    return num;
}

int findMax(int x, int y) {

    int megistos;

    if (x > y)
        megistos = x;
    else
        megistos = y;

    return megistos;
}

void printMax(int m) {
    printf("Max: %d\n\n", m);
}
```

2. Κλήση συνάρτησης

Μία συνάρτηση την καλούμε όταν χρειαζόμαστε να εκτελεστεί ο κώδικας της. Ο τρόπος κλήσης εξαρτάται από το πως την έχουμε δηλώσει:

- **συνάρτηση με τύπο επιστροφής void**: όταν την καλούμε δεν χρειάζεται να αποθηκεύσουμε κάτι όταν τερματίσει, οπότε **σύνταξή** τους φαίνεται παρακάτω:

```
printMax(max);
```

- **συνάρτηση που επιστρέφει κάποια τιμή**: όταν την καλούμε πρέπει να αποθηκεύσουμε σε μια μεταβλητή την τιμή που επέστρεψε, άρα αριστερά της κλήσης της θα πρέπει να εισάγουμε εκχώρηση τιμής:

```
max = findMax(a, b);
```

2. Αναδρομική συνάρτηση

Αναδρομική ονομάζεται μια συνάρτηση που μέσα στο σώμα της καλεί το ίδιο της τον εαυτό με άμεσο ή έμμεσο τρόπο. Γενικότερα, στα μαθηματικά υπάρχουν πολλές συναρτήσεις που ορίζονται αναδρομικά, για παράδειγμα το παραγοντικό ενός αριθμού μπορεί να οριστεί και χωρίς αναδρομή:

$$N! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot N$$

αλλά και αναδρομικά:

$$N! = \begin{cases} N \cdot (N-1)!, \forall N > 1 \\ 1, N = 0,1 \end{cases}$$

Παρατηρούμε ότι ο αναδρομικός ορισμός

- περιέχει μέσα του την αναδρομή
- περιέχει το σημείο τερματισμού της αναδρομής

Παράδειγμα - Παραγοντικό αριθμού

Να γίνει πρόγραμμα στο οποίο ο χρήστης εισάγει έναν θετικό ακέραιο αριθμό κ το πρόγραμμα υπολογίζει και εκτυπώνει το **παραγοντικό** του αριθμού που εισήγαγε ο χρήστης .

Λύση

```
#include <stdio.h>

int paragontiko(int);

int main(int argc, const char * argv[]) {

   int N, par;

   do {
   printf("Give positive N:");
   scanf("%d", &N);
   } while(N<1);

   par = paragontiko(N);

   printf("%d! = %d\n\n", N, par);

   return 0;
}

int paragontiko(int n) {

   if (n <= 1)
      return 1;
   else
      return n * paragontiko(n-1);
}
```

Παράδειγμα - Ακολουθία αριθμών Fibonacci

Να γίνει πρόγραμμα στο οποίο ο χρήστης εισάγει τον τελικό όρο Fibonacci και το πρόγραμμα θα τον εκτυπώνει.

Λύση

```
#include <stdio.h>

int fibon(int);

int main(int argc, const char * argv[]) {

    int N, oros;

    do {
        printf("Τελικός όρος:");
        scanf("%d", &N);
    } while (N<1);

    oros = fibon(N);

    printf("Ο %dος όρος είναι ο %d\n\n", N, oros);

    return 0;
}

int fibon(int n) {
    if (n<=2)
        return 1;
    else
        return fibon(n-1) + fibon(n-2);
}
```