Εισαγωγή στον δομημένο προγραμματισμό

Ενότητα 8η: Συναρτήσεις

Αν. καθηγητής Στεργίου Κώστας
e-mail: kstergiou@uowm.gr

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα στο Πανεπιστήμιο Δυτικής Μακεδονίας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Στόχοι της διάλεξης

- Εισαγωγή στην έννοια του αρθρωτού προγραμματισμού και της αφαίρεσης, καθώς και στην υλοποίηση αυτών των εννοιών στη C μέσω των συναρτήσεων.
- Ανάλυση του τρόπου χρήσης των συναρτήσεων στη C.
- Ανάλυση των τρόπων κλήσης συναρτήσεων (με τιμή και με αναφορά).
- Περιγραφή σχετικών παραδειγμάτων.
- Σύντομη περιγραφή ορισμένων βασικών βιβλιοθηκών της C.

Περιεχόμενα

- Αρθρωτός προγραμματισμός και αφαίρεση.
- Συναρτήσεις.
  - Δήλωση.
  - Ορισμός.
  - Κλήση.
  - Επιστροφή από συνάρτηση.
- Οι βασικές βιβλιοθήκες της C.
- Μεταβίβαση παραμέτρων:
  - Με τιμή.
  - Με αναφορά.

Αρθρωτός Σχεδιασμός (1/3)

- Ως τώρα όλα τα προγράμματα που είδαμε ήταν γραμμένα μέσα στην main.
- Τι θα κάνουμε όμως αν:
  - Το πρόγραμμα είναι τεράστιο (π.χ. 100000 γραμμές);
  - Υπάρχουν κομμάτια κώδικα τα οποία θέλουμε να εκτελέσουμε πολλές φορές;
- Θα χωρίσουμε το πρόγραμμα σε κομμάτια.
  - Φιλοσοφία του “διαίρει και βασίλευε”.
  - Το κάθε κομμάτι σε ένα επιμέρους υποπρόβλημα.
- Τι θα κερδίσουμε;

Αρθρωτός Σχεδιασμός (2/3)

- Τα υποπροβλήματα είναι μικρότερα και συνεπώς πιο εύκολα στη διαχείριση.
- Δημιουργούμε μια συνάρτηση για κάθε υποπρόβλημα
  - Ο κώδικας είναι πιο ευανάγνωστος και ευκολότερος στη συντήρηση.
  - Μπορούμε να χρησιμοποιήσουμε συναρτήσεις που έχουμε γράψει στο παρελθόν για την επίλυση νέων προβλημάτων.
  - Αποφεύγουμε την επανάληψη κώδικα στο ίδιο πρόγραμμα.
- Επιτυγχάνεται **αφαίρεση**. Δηλ. απόκρυψη μη απαραίτητων λεπτομερειών από τον προγραμματιστή.
  - Σκεφτείτε ομάδες προγραμματιστών που υλοποιούν ένα μεγάλο σύστημα.

Αρθρωτός Σχεδιασμός (3/3)

- Βασίζει την ανάπτυξη σύνθετων συστημάτων στην τμηματοποίηση των σύνθετων διεργασιών σε επιμέρους απλούστερες διεργασίες.
- Τμηματοποίηση ενός προγράμματος σε συναρτήσεις (functions).
- Κάθε συνάρτηση πρέπει να είναι:
  - Σχετικά μικρή.
  - Αν είναι δυνατό να έχει μία είσοδο και μία έξοδο.
  - Να είναι αυτόνομη.
  - Να εκτελεί σαφώς ορισμένο έργο.
  - Να έχει καλά προσδιορισμένη διεπαφή.

Παράδειγμα: Σύγκριση μήκους δύο αλφαριθμητικών a και b

- **Διεργασία 1:** Βρες το μήκος του αλφαριθμητικού a → a_length.
- **Διεργασία 2:** Βρες το μήκος του αλφαριθμητικού b → b_length.
- **Διεργασία 3:** Σύγκρινε τους ακέραιους a_length και b_length και βρες το μεγαλύτερο.

Συναρτήσεις (1/2)

- Μία αυτόνομη μονάδα κώδικα που επιτελεί ορισμένο έργο:
  - Αποφυγή της επανάληψης.
  - Αύξηση της επαναχρησιμοποίησης.
  - Βελτίωση της αναγνωσιμότητας του κώδικα.
  - Βελτίωση της συντήρησης του κώδικα.
- Κάθε πρόγραμμα C αποτελείται από τη συνάρτηση main και (προαιρετικά) από ένα αριθμό άλλων συναρτήσεων.

Συναρτήσεις (2/2)

- Η βιβλιοθήκη της C παρέχει πολλές έτοιμες συναρτήσεις:
  - `printf()` – συνάρτηση εκτύπωσης.
  - `scanf()` – συνάρτηση εισόδου.
  - `gets()` – συνάρτηση εισόδου αλφαριθμητικών.
  - `strcpy()` – συνάρτηση αντιγραφής αλφαριθμητικών.
  - …
- Αν για κάτι που θέλουμε να κάνει το πρόγραμμα μας, βοηθάει μια έτοιμη συνάρτηση τότε την χρησιμοποιούμε.
  - Δεν ξαναγράφουμε τον κώδικα από την αρχή.
  - Χρησιμοποιήστε τις συναρτήσεις της βιβλιοθήκης όσο περισσότερο μπορείτε.

Δήλωση Συνάρτησης (1/2)

- <τύπος> <όνομα συνάρτησης>(
  <τύπος1> [<παράμετρος1>], ...,
  <τύποςN> [<παράμετροςN>]);
  - Οι παράμετροι της συνάρτησης υποστηρίζουν το πέρασμα εισόδων στη συνάρτηση.
  - Η επιστρεφόμενη τιμή υποστηρίζει την επιστροφή του αποτελέσματος.
  - Η δήλωση μιας συνάρτησης ονομάζεται και **πρωτότυπο ή αρχέτυπο.**

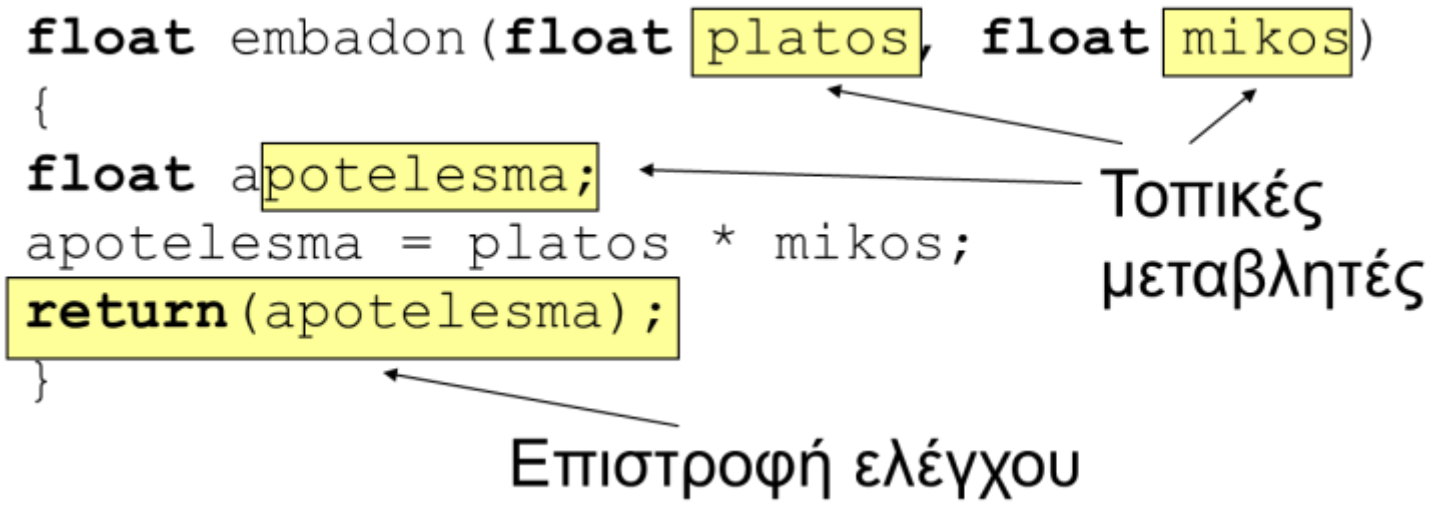
Δήλωση Συνάρτησης (2/2)

```
int max(int a, int b);
int max(int, int);
int string_length(char str[]);
char *getptr(char *str, char ch);
void draw_circle(double x,double y, doubler)
```

- **void**: δεν επιστρέφεται καμία τιμή.
- Οι δηλώσεις των printf και scanf βρίσκονται στο αρχείο stdio.h.

Ορισμός Συνάρτησης

- Αποτελείται από το πρωτότυπο της συνάρτησης και το σύνολο των προτάσεων που επιτελούν το έργο της συνάρτησης.
- Η επιστροφή τιμής γίνεται με τη λέξη-κλειδί **return**
  - π.χ. Συνάρτηση που υπολογίζει το εμβαδόν ορθογωνίου:

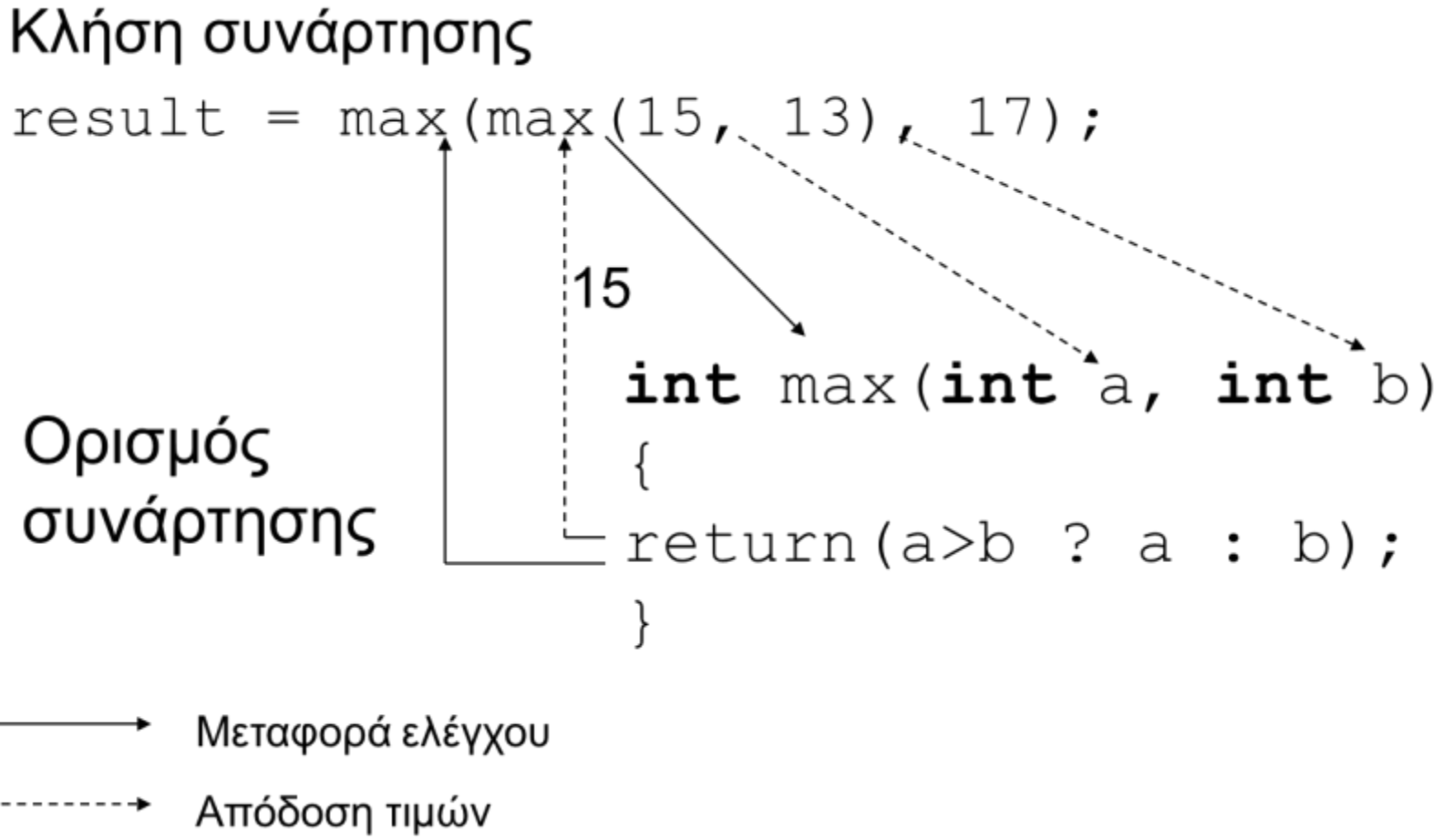


Κλήση Συνάρτησης

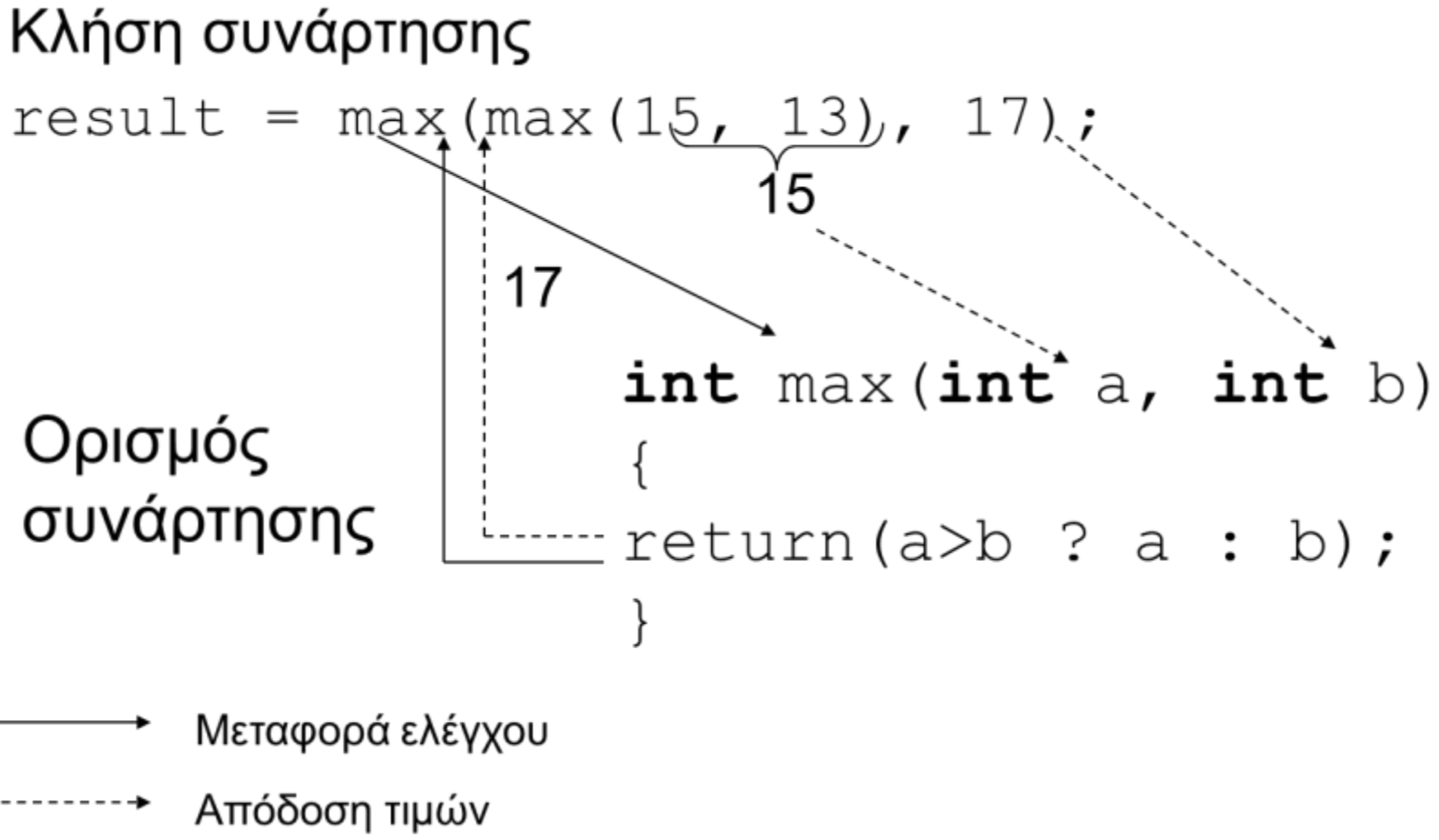
- Θυμηθείτε τις κλήσεις των συναρτήσεων printf και scanf.

- <όνομα συνάρτησης>(όρισμα1, ..., όρισμαΝ)
  - `draw_circle(2.5, 3.0, 3.2);`
  - `draw_circle(2.5*c, 3.0*b, 3.2);`
  - `report_error();`
  - `x=max(num1, num2);`
  - `result = num1 + max(num1, num2);`
  - `printf("%d", max(num1, num2));`

Παράδειγμα: Κλήση Συνάρτησης (1/2)



Παράδειγμα: Κλήση Συνάρτησης (2/2)



Παράδειγμα: Πρωτότυπο, Κλήση, Ορισμός Συνάρτησης

```
#include <stdio.h>
int max(int a, int b);                          /* Πρωτότυπο */
int main(void)
{
  int num1, num2;
  printf("dwse duo arithmous"\n);
  scanf("%d %d",&num1,&num2);
  printf("o megaliteros anamesa stous %d kai %d einai o %d",
          num1, num2, max(num1,num2)); /* Κλήση */
  return 0;
}
int max(int a, int b)                           /* Ορισμός */
{  return (a>b ? a : b);
}
```

Επιστροφή από συνάρτηση

- Με τη δεσμευμένη λέξη **return** τερματίζει μια συνάρτηση.
  - Αν υπάρχει κώδικας παρακάτω, δεν θα εκτελεστεί.

```
int max(int a, int b, int c)
{
  int temp;
  temp = a;
  if (temp < b)
     temp = b;
  if (temp < c)
     temp = c;
  return temp;
  printf("o megaliteros einai o %d", temp);

  return 0;
}
```

Αυτό ισχύει ακόμα κι η συνάρτηση δηλωθεί να μην επιστρέφει τίποτα (**void**).

Δεν εκτελούνται

void συναρτήσεις

- Η δεσμευμένη λέξη **void** υποδηλώνει ότι η συνάρτηση δεν επιστρέφει τίποτα.

```
void print_stars(int n)
{
  int i;

  for (i=0; i<n; i++)
        putchar('*');
}
```

- Η συνάρτηση τυπώνει n αστερίσκους και δεν επιστρέφει τίποτα.

Τοπικές Μεταβλητές

- Η παρακάτω συνάρτηση έχει 4 τοπικές μεταβλητές:
  - οι παράμετροι `a`, `b`, `c` και η μεταβλητή `temp`.

```
int max(int a, int b, int c)
{
  int temp;
  temp = a;
  if (temp < b)
     temp = b;
  if (temp < c)
     temp = c;
  return temp;
  printf("o megaliteros einai o %d",
   temp);
  return 0;
}
```

Οι μεταβλητές αυτές είναι γνωστές μόνο μέσα στην συνάρτηση.

Άρα δεν μπορούν να προσπελαστούν π.χ. από την main.

Ροή Προγράμματος

- Όταν κατά την εκτέλεση ενός προγράμματος συναντηθεί η κλήση μιας συνάρτησης η ροή εκτέλεσης αλλάζει.
  - Έστω η κλήση `f(num1,num2,num3.`

1. Υπολογισμός των εκφράσεων `num1,num2, num3.`
2. Αντιγραφή τους στις παραμέτρους της συνάρτησης.
3. Εκτελείται η πρώτη εντολή της `f`, μετά η δεύτερη, κ.ο.κ.
   - Μέχρι να συναντηθεί **return** (ή } αν είναι **void**).
4. Γίνεται αντικατάσταση της κλήσης `f(num1,num2,num3)` με την τιμή που επέστρεψε η `f.`
5. Η εκτέλεση του κώδικα συνεχίζεται με την εντολή που ακολουθεί την κλήση `f(num1,num2,num3).`

Ροή Προγράμματος - Παράδειγμα

```
#include <stdio.h>
int max(int a, int b);
int main(void)
{   int num1, num2, num3,
    megaliteros;
   printf("dwse treis
    arithmous"\n);
   scanf("%d %d
    %d",&num1,&num2,&num3);
   megaliteros =
    max(num1,num2,num3);
   printf("max is
    %d",megaliteros);
   return 0; }
int max(int a, int b, int c) {
   int temp = a;
   if (temp < b) temp = b;
   if (temp < c) temp = c;
   return temp;
}
```

- Πρώτα εκτελούνται το printf και το scanf της main.
- Μετά έχουμε κλήση συνάρτησης.
- Υπολογίζονται οι τιμές των num1,num2,num3.
- Ανατίθεται στην μεταβλητή η τιμή που επέστρεψε η max και εκτελείται το printf.
- Οι τρεις τιμές αντιγράφονται στις παραμέτρους της max a,b,c.
- Εκτελούνται οι εντολές της max και η συνάρτηση επιστρέφει έχοντας υπολογίσει το temp.

Κλήση συνάρτησης από συνάρτηση

- Μπορούμε να καλέσουμε μια συνάρτηση μέσα από μια άλλη συνάρτηση.

  Ποια είναι η ροή του προγράμματος σε αυτό το παράδειγμα;

```
#include <stdio.h>
int max(int a, int b, int c);
int print_max(int);
int main(void)
{
   int num1, num2, num3,
    megaliteros;
   printf("dwse treis arithmous"\n);
   scanf("%d %d
    %d",&num1,&num2,&num3);
   megaliteros =
    max(num1,num2,num3);
   return 0;
}
int max(int a, int b, int c) {
   int temp = a;
   if (temp < b) temp = b;
   if (temp < c) temp = c;
   print_max(temp);
   return temp;
}
void print_max(int num) {
   printf("max is %d",num);
}
```

Πρωτότυπα συναρτήσεων

```
#include <stdio.h>
int max(int a, int b);
int main(void)
{
  int num1, num2, num3, megaliteros;
  printf("dwse treis arithmous"\n);
  scanf("%d %d
    %d",&num1,&num2,&num3);
  megaliteros = max(num1,num2,num3);
  printf("max is %d",megaliteros);
  return 0;
}
int max(int a, int b, int c) {
  int temp = a;
  if (temp < b) temp = b;
  if (temp < c) temp = c;
  return temp;
}
```

- Το πρωτότυπο μας δίνει τη δυνατότητα να καλέσουμε μια συνάρτηση χωρίς να την έχουμε ορίσει πλήρως.
- Ο ορισμός της συνάρτησης max ακολουθεί την κλήση της.
  - Χωρίς πρωτότυπο θα είχαμε πρόβλημα.

Κανόνες Ευανάγνωστου Προγράμματος (1/2)

- Οργανώστε τον κώδικά σας σύμφωνα με τα παρακάτω:
  - Εντολές προεπεξεργαστή (#include, #define).
  - Πρωτότυπα συναρτήσεων.
  - Δηλώσεις γενικών μεταβλητών.
  - Συνάρτηση main.
  - Ορισμοί λοιπών συναρτήσεων.
- Μεταξύ των παραπάνω ομάδων αφήνετε 2-3 κενές γραμμές.

Κανόνες Ευανάγνωστου Προγράμματος (2/2)

```
/*Παρεμβάλετε σχόλια όπου κρίνετε απαραίτητο */
/* δεν επιτρέπονται /* επικαλυπτόμενα */ σχόλια*/
```

- Χρησιμοποιήστε εκφραστικά ονόματα μεταβλητών.
- Μία πρόταση ανά γραμμή.
- Όλες οι δηλώσεις μεταβλητών σε μία θέση.
- **ΠΡΙΝ** αρχίσετε να γράφετε κώδικα σχεδιάστε καλά το πρόγραμμά σας.

Πίνακες και Συναρτήσεις

- Όταν ένα όρισμα μιας συνάρτησης είναι μονοδιάστατος πίνακας δεν είναι απαραίτητο να ορίζουμε τις διαστάσεις του:
  - `float func(int x, float list[])`
- Όταν ένα όρισμα μιας συνάρτησης είναι δισδιάστατος πίνακας πρέπει να ορίζουμε τη δεύτερη διάστασή του (στήλες):
  - `float func(int x, float[][5])`

Παράδειγμα

```
/* Εύρεση μέσου όρου στοιχείων πίνακα */
float av1(int x, float a[])
{  int i;
   float sum=0.0;
   for (i=0; i<x; i++)
        sum += a[i];
   return(sum/x);
}
/* Εκτύπωση στοιχείων δισδιάστατου πίνακα */
void print1(int x, int y, float b[][5])
{  int i,j;
   for (i=0; i<x; i++) {
        for (j=0; j<y; j++)
              printf("%f", b[i][j]);
        printf("\n"); }
}
```

Σύγκριση Αλφαριθμητικών

/* strcmp: επιστρέφει <0 αν s<t, 0 αν s=t, και >0 s>t */

Χωρίς δείκτες

```
int strcmp(char s[], char t[])
{       int i;
        for (i=0; s[i]==t[i]; i++)
                if (s[i]=='\0')
                        return 0;
        return s[i]-t[i];
}
```

Με δείκτες

```
int strcmp(char *s, char *t)
{       int i;
        for ( ; *s==*t; s++, t++)
                if (*s=='\0')
                        return 0;
        return *s-*t;
}
```

Υπολογισμός $1^k+2^k+...+n^k$

```
int athroisma(int k, int n)
{
   int i, athroisma=0;
   for (i=1; i<=n; ++i)
       athroisma += dynami(i, k);
   return athroisma;
}

int dynami(int m, int n)
{
   int i, ginomeno=1;
   for (i=1; i<=n; ++i)
       ginomeno *= m;
   return ginomeno;
}
```

Υπολογισμός Αριθμών Fibonacci (1/2)

```
#include <stdio.h>
int fibonacci(int n);
int main (void)
{
int n;
printf("Dose ena thetiko arithmo:");
scanf("%d",&n);
printf("O %d-os arithmos fibonacci einai: %d", n,
   fibonacci(n));
return 0;
}
```

Υπολογισμός Αριθμών Fibonacci (2/2)

```
/* fibonacci: υπολογίζει τον ν-στο αριθμό */
int fibonacci(int n)
{
int protos=1, deuteros=1, metritis=3, arithmos;
if (n==1)arithmos=protos;
else if (n==2) arithmos=deuteros;
while (metritis<=n)
 {
   arithmos=protos+deuteros;
   protos=deuteros;
   deuteros=arithmos;
   metritis++;
 }
return arithmos;
}
```

Εύρεση της θέσης του αλφαριθμητικού t μέσα στο s (-1 αν δεν υπάρχει)

```
int strindex(char s[], char t[])
{
int i, j, k;
for (i=0; s[i] != '\0'; i++)
 {
   for (j=i, k=0; t[k]!='\0' && s[j]==t[k]; j++, k++)
       ;
   if (k>0 && t[k] == '\0')
       return i;
 }
return -1;
}
```

? Πώς μπορούμε να βρούμε την δεξιότερη εμφάνιση του t στο s ?

Βασική Βιβλιοθήκη της C (1/2)

- Σχεδόν σε όλα τα παραδείγματα λέμε στον προ-επεξεργαστή να συμπεριλάβει το **αρχείο επικεφαλίδων** `stdio.h`.
  -
```
#include <stdio.h>
int main ()
{
 …
 }
```
- Αυτό το αρχείο βρίσκεται αποθηκευμένο στο σύστημα (σε διαφορετικό σημείο ανάλογα με το λειτουργικό) και περιέχει πρωτότυπα συναρτήσεων για Είσοδο/Έξοδο (I/O).
  - Υπάρχουν πολλά άλλα τέτοια αρχεία για την υποβοήθηση διάφορων λειτουργιών.

Βασική Βιβλιοθήκη της C (2/2)

- Ένα σύνολο από συναρτήσεις που επιτελούν βασικές διεργασίες.
  - “Έτοιμος” μεταγλωττισμένος κώδικας.
- Η χρήση τους:
  - Μειώνει σημαντικά το χρόνο ανάπτυξης.
  - Αυξάνει την αξιοπιστία του προγράμματος.
- **Βασικές κατηγορίες:**
  - Διαχείρισης αλφαριθμητικών.
  - Μαθηματικές.
  - Ταξινόμησης χαρακτήρων.
  - Εισόδου/εξόδου.
  - Διαχείρισης αρχείων.

Βασικά αρχεία επικεφαλίδων

<stdio.h>	αρχέτυπα συναρτήσεων για είσοδο/έξοδο
<stdlib.h>	αρχέτυπα συναρτήσεων για μετατροπή αριθμών σε κείμενο και αντίστροφα, κατανομή μνήμης, τυχαίους αριθμούς και άλλες βοηθητικές συναρτήσεις
<math.h>	αρχέτυπα συναρτήσεων για μαθηματικές συναρτήσεις
<assert.h>	προσθήκη διαγνωστικών για διόρθωση προγράμματος
<ctype.h>	έλεγχος ιδιοτήτων χαρακτήρων και μετατροπές από πεζά γράμματα σε κεφαλαία και αντιστρόφως

Μαθηματικές συναρτήσεις

Η βιβλιοθήκη της C ορίζει μέσω του <math.h> διάφορες μαθηματικές συναρτήσεις, όπως

```
double sqrt( double x );

double exp( double x );

double log( double x );

double pow( double x, double y );

double sin ( double x );
```

Πολλές από αυτές υπάρχουν και για τον τύπο float.

Μετατροπή Αλφαριθμητικού σε Ακέραιο

```
/* atoi: “  125”=125, “ -125”=-125 */
#include <ctype.h>
int atoi(char s[])
{
   int i, n, sign;
   for (i=0; isspace(s[i]); i++) ;
   if (s[i]==‘-’)
    {  sign = -1;
       i++; }
   else sign = 1;
   for (n=0; isdigit(s[i]); i++)
       n=10*n+(s[i]-‘0’);
   return sign*n;
}
```

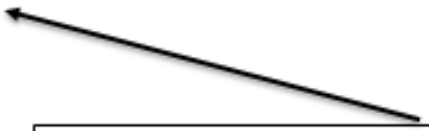
Μετατροπή Αλφαριθμητικού σε Πραγματικό

```
/* atof: "  125.33"=125.33, " -125"=-125 */
#include <ctype.h>
double atof(char s[])
{  double val, power;
   int i, sign;
   for (i=0; isspace(s[i]); i++) ;
   if (s[i]=='-')
    {   sign = -1;
        i++; }
   else sign = 1;
   for (val=0.0; isdigit(s[i]); i++)
        val=10.0*val+(s[i]-'0');
   if (s[i]=='.') i++;
   for (power=1.0; isdigit(s[i]); i++)
        {val = 10.0*val+(s[i]-'0');
        power *= 10.0; }
   return sign*val/power;
}
```

atoi με χρήση της atof

```
/* Μετατροπή αλφαριθμητικού σε ακέραιο με
   χρήση της atof */

   int atoi(char s[])
   {
      return (int) atof(s);
   }
```



Αν δεν δηλωθεί ρητώς,
η μετατροπή αυτή μπορεί να εμφανίσει
warning κατά την μεταγλώττιση

Μεταβίβαση Παραμέτρων

- Όταν καλούμε μία συνάρτηση, η μεταβίβαση παραμέτρων μπορεί να γίνει με δύο τρόπους:
  - **Κλήση κατά τιμή** (call by value):
    Η συνάρτηση δουλεύει σε αντίγραφα των πραγματικών παραμέτρων.
  - **Κλήση κατ' αναφορά** (call by reference):
    Η συνάρτηση εφαρμόζεται στις πραγματικές παραμέτρους.

Κλήση κατά τιμή

- Στην κλήση κατά τιμή, πριν κληθεί μια συνάρτηση δημιουργείται ένα αντίγραφο όλων των μεταβλητών που έχουν δοθεί ως παράμετροι.
- Η συνάρτηση καλείται με παραμέτρους αυτά τα αντίγραφα.
- Με το τέλος της συνάρτησης τα αντίγραφα αυτά σβήνονται από την μνήμη.

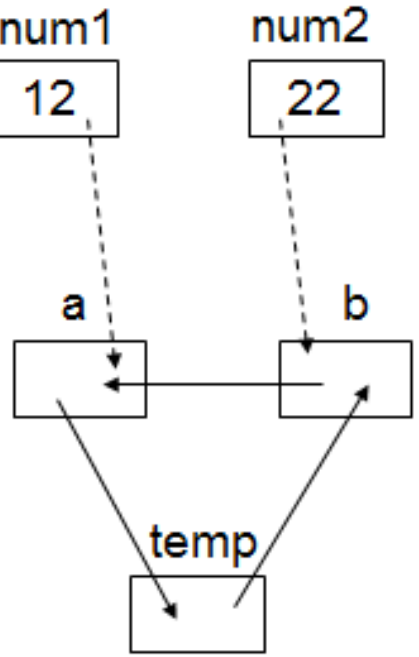
```
#include <stdio.h>
int max(int a, int b);
int main(void)
{
  int num1, num2, num3
  int max_num;
  …
  max_num =
   max(num1,num2,num3);
  …
  return 0;
}

int max(int a, int b, int c) {
  int temp = a;
  if (temp < b) temp = b;
  if (temp < c) temp = c;
  return temp;
}
```

Παράδειγμα Κλήσης Κατά Τιμή

```
/* Ορισμός Συνάρτησης */
void swap(int a, int b)
{
   int temp;
   temp = a;
   a = b;
   b = temp;
}
/* Κλήση συνάρτησης */
num1=12; num2=22;
swap(num1, num2);
```

Οι τιμές των a και b αλλάζουν
αλλά
οι τιμές των num1 και num2 όχι



Κλήση κατ’ αναφορά

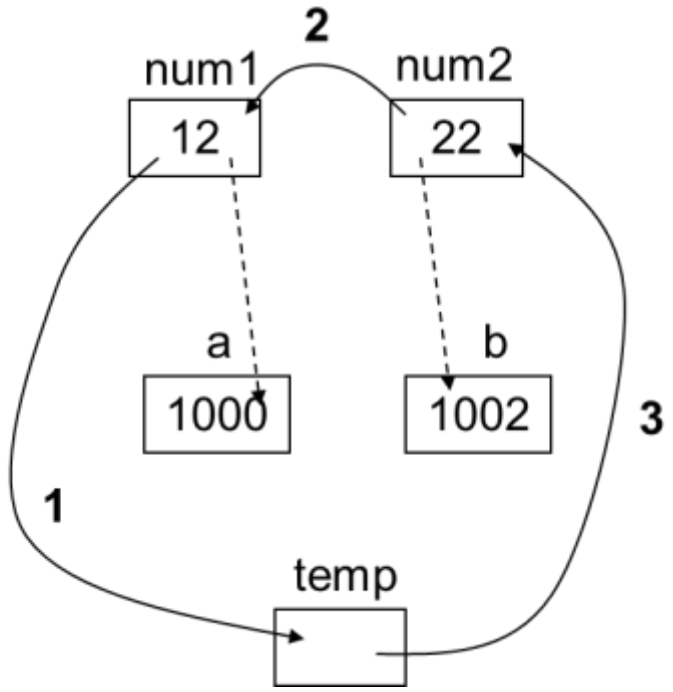
- Στην κλήση μέσω αναφοράς η συνάρτηση καλείται με τις πραγματικές μεταβλητές ως παραμέτρους.

- Η γλώσσα C δεν υποστηρίζει κλήση μέσω αναφοράς, αλλά μπορεί να την προσομοιώσει μέσω της χρήσης δεικτών.

Παράδειγμα Κλήσης Κατ' Αναφορά

```
/* Ορισμός Συνάρτησης */
void swap(int *a, int *b)
{
   int temp;
   temp = *a;
   *a = *b;
   *b = temp;
}
/* Κλήση συνάρτησης */
num1=12; num2=22;
swap(&num1, &num2);
```

Οι δείκτες a και b δείχνουν
στις num1, num2.
Οι τιμές των num1 και num2 αλλάζουν



Τι θα τυπωθεί στις 2 περιπτώσεις;

```
#include <stdio.h>
void increment (int x);
int main (void)
{  int x1=1;
   printf("\nx1=%d", x1);
   increment(x1);
   printf("\nx1=%d",x1);
   return 0;
}

void increment(int x)
{
   x=x+1;
   printf("\nx=%d",x);
}
```

```
#include <stdio.h>
void increment (int *x);
int main (void)
{       int x1=1;
        printf("\nx1=%d",
x1);
        increment(&x1);
        printf("\nx1=%d",x1);
        return 0;
}

void increment(int *x)
{
        *x=*x+1;
        printf("\nx=%d",*x);
}
```

Αντιστροφή Αλφαριθμητικού

```
/* reverse: αντιστροφή αλφαριθμητικού s στη θέση του
   */
#include <string.h>
void reverse(char s[])
{
   int c, i, j;
   for (i=0, j=strlen(s)-1; i<j; i++, j--)
   {
       c=s[i];
       s[i]=s[j];
       s[j]=c;
   }
}
```

Τέλος Ενότητας

Σημείωμα Αναφοράς

- Copyright Πανεπιστήμιο Δυτικής Μακεδονίας, Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών, Στεργίου Κωνσταντίνος. «Εισαγωγή στον Δομημένο Προγραμματισμό». Έκδοση: 1.0. Κοζάνη 2015. Διαθέσιμο από τη δικτυακή διεύθυνση:

  https: //eclass.uowm.gr/courses/ICTE258/

Σημείωμα Αδειοδότησης

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.