Π. Σταθοπούλου
pstath@ece.upatras.gr ή
pstath@upatras.gr

Ομάδα Α΄ (Φοιτητές με μονό αριθμό Μητρώου)

- Διδασκαλία : Παρασκευή 11πμ-13μμ ΗΛ7
- Φροντιστήριο : Δευτέρα 11πμ-12πμ ΗΛ4

Προηγούμενη Διάλεξη

Προτάσεις, Εκφράσεις & Τελεστές

- *είδη προτάσεων, προτάσεις δομής συνθήκης*
  (if, switch-case)
- *προτάσεις επαναληπτικής δομής ελέγχου*
  (for, while, do ..... while)
- *μαθηματικούς τελεστές & η σειρά προτεραιότητας αυτών*

Σήμερα

Μαθηματικοί Τελεστές
- *Είδη τελεστών & προτεραιότητα*

Έννοια της συνάρτησης
- *Ορισμός, σύνταξη & λειτουργία*

Μαθηματικοί Τελεστές

Προτεραιότητα μαθηματικών **τελεστών** & **παρενθέσεων**

Μαθηματικοί Τελεστές
- Μοναδιαίους
  - **++** τελεστής αύξησης,
  - **--** τελεστής μείωσης,

Μαθηματικοί Τελεστές
- Δυαδικούς
  - **+** πρόσθεση,
  - **–** αφαίρεση,
  - **/** διαίρεση,
  - **%** υπόλοιπο διαίρεσης
  - ***** πολλαπλασιασμός

Μαθηματικοί Τελεστές

- **Οι Μοναδιαίοι Μαθηματικοί Τελεστές** *ονομάζονται έτσι διότι δέχονται μόνο ένα τελεστέο (μεταβλητή).*
- *Χρησιμοποιούνται μόνο με* ***μεταβλητές*** *και όχι με* ***σταθερές***

Σύμβολο	Ενέργεια	Τελεστής	Παράδειγμα
++	Αυξάνει τον τελεστέο κατά ένα	αύξησης	**++x, x++**
--	Μειώνει τον τελεστέο κατά ένα	μείωσης	**--x, x--**

```
++x   x=x+1;
--y   y=y+1;
```

```
x=10;   } y=11
y=x++   } x=10
```

```
x=10;   } y=11
y=++x   } x=11
```


Μαθηματικοί Τελεστές

- **Οι Δυαδικοί Μαθηματικοί Τελεστές** *ονομάζονται έτσι διότι δέχονται δύο τελεστέους (μεταβλητές).*

Προτεραιότητα μαθηματικών τελεστών & παρενθέσεων

Προτεραιότητα μαθηματικών τελεστών	
Τελεστές	**Σχετική προτεραιότητα**
++ -- ←	**1**
*** / %**	**2**
+ -	**3**

Μαθηματικοί Τελεστές

Προτεραιότητα μαθηματικών τελεστών
& **παρενθέσεων**

Παράδειγμα 1ο

12%5*2 **4**

γιατί?

12%5 είναι 2

2*2 είναι 4

Παράδειγμα 2ο

x=4 + 5*3 19 ή 27

Τι τιμή ?

if x=(4+5)*3 είναι 27

if x=4+5*3 είναι 19

Μοναδιαίοι τελεστές

Μοναδιαίους Τελεστές

++	τελεστής αύξησης,
--	τελεστής μείωσης,
sizeof()	δίνει το μέγεθος σε bytes
&	δίνει την διεύθυνση της μεταβλητής στην μνήμη του υπολογιστή

Τελεστές μνήμης

& * [] . ->

πίνακες

δομές

Λογικοί τελεστές

Οι **Λογικοί τελεστές** επιτρέπουν να συνδυάζονται δύο ή περισσότερες σχεσιακές εκφράσεις σε μία έκφραση που εκτιμάται σε **αληθή** ή **ψευδής**

Τελεστής	Σύμβολο	Παράδειγμα
AND	&&	expr1 && expr2
OR	||	expr1 || expr2
NOT	!	expr1! expr2

Προτεραιότητα λογικών τελεστών	
Τελεστές	Σχετική προτεραιότητα
!	1
||	2
&&	3

Λογικοί τελεστές

Προτεραιότητα λογικών τελεστών & παρενθέσεων

Παράδειγμα 1ο

(5==5) && (6!=2) true
(5 >1) || (6 <1) true
(2==1) && (5==5) false
!(5==4) true

Παράδειγμα 2ο

(x>1) && (x<5)
Χ μεταξύ 1 & 5
(x>=2) && (x <=4)
x μεταξύ 2 & 4
(x==2)||(x==3)||(x==4)
x ίσον ή του 2 ή του 3 ή του 4

Συνδυαστικοί τελεστές

Οι **συνδυαστικοί τελεστές εκχώρησης** παρέχουν συντομευμένη μέθοδο για τον συνδυασμό μιας **δυαδικής μαθηματικής πράξης** & της **πράξης εκχώρησης**

Έστω ότι

x=x+5; → ***x+=5;***

γενική σύνταξη

exp1 op=exp2;

Συνδυαστικοί τελεστές

Παραδείγματα συνδυαστικών τελεστών	
x *= y	x = x*y
y -= z+1	y = y-z+1
a /= b	a = a/b
y %= 3	y = y%3
x += y/8	x = x + y/8

Τελεστές

Τελεστής συνθηκών = τριαδικός τελεστής

?

Διαχειρίζεται τρεις τελεστέους (μεταβλητές)

Η σύνταξη του → ***expr1 ? exp2 : expr3***

Παραδείγματα ***x = y ? 1 : 100***

```
if (x>y)
z=x;
else   z=y;
```

z = (x >y) ? x : y

τελεστές

Τελεστής κόμμα

,

- *διαχωρίζει τις μεταβλητές, τα ορίσματα*
- *διαχωρίζει δευτερεύουσες προτάσεις (**for**)*

- *Οι δύο εκφράσεις υπολογίζονται με το να προηγείται η αριστερή έκφραση*
- *Ολόκληρη η έκφραση εκτιμάται στην τιμή της δεξιάς*

τελεστές

Παράδειγμα

x=(a++, b++);

- *εκχωρεί στην μεταβλητή **x** την τιμή της **b***
- *αυξάνει την μεταβλητή **a***
- *αυξάνει την μεταβλητή **b***

Γενική παρουσίαση προτεραιότητας τελεστών

επίπεδο	τελεστές
1	(), [], - >.
2	!, ~, ++, --
3	*, /, %
4	+, -
5	<< >>
6	<, <=, >, >=
7	==, !=
8	&
9	^

επίπεδο	τελεστές
10	|
11	&&
12	||
13	?, :
14	=,+=, -=, *=, /=,%=, &=, ^=, |=,<<=, >>=
15	==, !=
()	τελεστής συνάρτησης
[]	τελεστής πίνακα

Έννοια της συνάρτησης

- *Ορισμός συνάρτησης*
- *Πως λειτουργεί μία συνάρτηση*
- *Συναρτήσεις & δομημένος προγραμματισμός*
- *Δήλωση συνάρτησης*
- *Σώμα συνάρτησης*
- *Αρχέτυπη συνάρτηση*
- *Διοχέτευση ορισμάτων σε συνάρτηση*
- *Κλήση συναρτήσεων*
- *Αναδρομή*
- *Θέση συναρτήσεων*

Έννοια της συνάρτησης

Η συνάρτηση *είναι το βασικό δομικό στοιχείο της C που μας επιτρέπει να εφαρμόσουμε την* ***αφαιρετικότητα*** *στον προγραμματισμό*

Η αφαιρετικότητα *είναι μία από τις βασικότερες τεχνικές που ο άνθρωπος εισήγαγε για να διαχειριστεί* ***πολύπλοκες διαδικασίες***

Έννοια της συνάρτησης

Αφαιρετικότητα *είναι διεργασία γενίκευσης μιας έννοιας ή ενός φαινόμενου μέσα από την* ***απλοποίηση*** *της περιγραφής τους με τελικό στόχο να αναδειχθεί μόνο η* ***απαραίτητη πληροφορία*** *για τον συγκεκριμένο σκοπό.*

Έννοια της συνάρτησης

2 μορφές αφαιρετικότητας

- ***Αφαιρετικότητα στις διαδικασίες*** *(procedural abstraction)*
- ***Αφαιρετικότητα στα δεδομένα*** *(data abstraction)*

Αφαιρετικότητα στις διαδικασίες → *Συναρτήσεις*

Αφαιρετικότητα στα δεδομένα → *Δομές*

Ορισμός συνάρτησης

Μία συνάρτηση *είναι:*

- ***μια ονομαστική***
- ***ανεξάρτητη*** *ενότητα κώδικα C που*
- ***εκτελεί μία συγκεκριμένη εργασία*** *και*
- ***προαιρετικά επιστρέφει μία τιμή*** *στο πρόγραμμα την κάλεσε*

Ορισμός συνάρτησης

Μία συνάρτηση είναι ονομαστική

*Έχει ένα **μοναδικό** όνομα που της επιτρέπει την επικοινωνία με το πρόγραμμα που είναι συνδεδεμένη*

Μία συνάρτηση είναι ανεξάρτητη

*Εκτελεί την εργασία της **χωρίς παρεμβολή** από ή με άλλα τμήματα του προγράμματος*

Ορισμός συνάρτησης

Μία συνάρτηση εκτελεί συγκεκριμένη εργασία

- *η ταξινόμηση ενός πίνακα,*
- *η αποστολή μηνύματος στην οθόνη ή στον εκτυπωτή,*
- *ο υπολογισμός της κυβικής ρίζας*

Μία συνάρτηση επιστρέφει μια τιμή

- *το πρόγραμμα καλεί μία συνάρτηση*
- *οι προτάσεις που περιέχονται στην συνάρτηση εκτελούνται*
- *οι προτάσεις μπορούν να διοχετεύουν ή όχι στο πρόγραμμα που τις κάλεσε*

Ορισμός συνάρτησης - παράδειγμα

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

int cube(int x);
long input, answer;

int main(int argc, char *argv[])
{
printf("\n\t\tEnter an integer value for the length of the cube side: ");
scanf("%d", &input);
answer=cube(input);
printf("\n\t\tThe cube volume of side %d is %d\n\n",input,answer);

  system("PAUSE");
  return 0;
}

//function cube()
Int cube(int x)
{
int x_cubed;
x_cubed=x*x*x;
return x_cubed;
}
```

Δήλωση συνάρτησης

κλήση συνάρτησης

Σώμα συνάρτησης

Ορισμός συνάρτησης - παράδειγμα

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

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

  system("PAUSE");
  return 0;
}
```

Δήλωση αρχέτυπης συνάρτησης

```
int cube(int x);
```

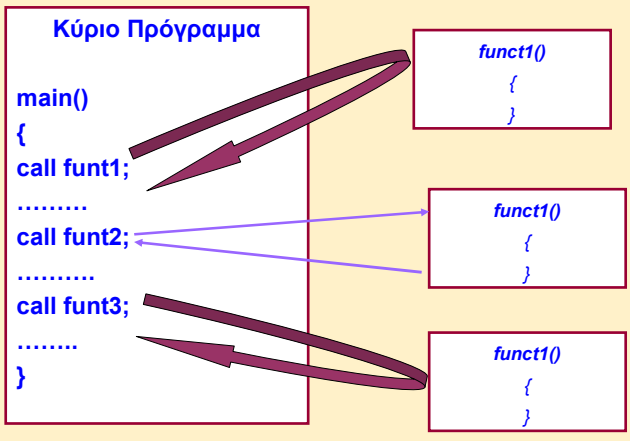
κλήση συνάρτησης

```
answer=cube(input);
```

Σώμα συνάρτησης

```
//function cube()
Int cube(int x)
{
int x_cubed;
x_cubed=x*x*x;
return x_cubed;
}
```


Λειτουργία συνάρτησης



Σύνταξη της συνάρτησης

Δήλωση αρχέτυπης (πρότυπο) συνάρτησης

<**τύπος**> **όνομα συνάρτησης (**<**τύπος**> *όνομα_μεταβλητής*$_1$, ... ,<**τύπος**> *όνομα_μεταβλητής*$_n$**);**

Ορισμός συνάρτησης **= κώδικας της συνάρτησης**

```
<τύπος>  όνομα συνάρτησης (<τύπος> όνομα_μεταβλητής1, .
                          . . . . . ,<τύπος> όνομα_μεταβλητήςn)
{
      προτάσεις;

}
```


Παραδείγματα συναρτήσεων

Όνομα συνάρτησης

τύπος μεταβλητής

1ο Παράδειγμα	
δήλωση	double squared (double number);

ορισμός	double squared (double number)
	{
	return(number*number);
	}

όνομα μεταβλητής

Παραδείγματα συναρτήσεων

2ο Παράδειγμα	
δήλωση	int athroisma_tetr (int n);

ορισμός	int athroisma_tetr (int n)
	{
	int i, athroisma= 0;
	for (i=1; i <= n; ++i)
	athroisma += i*i;
	return (athroisma);
	}

athroisma = athroisma + i*i;

Συναρτήσεις & δομημένος προγραματισμός

Χρήση συναρτήσεων

↓

Δομημένος προγραμματισμός

Πλεονεκτήματα του δομημένου προγραμματισμού

- *Εύκολο γράψιμο κώδικα*
- *Εύκολη διόρθωση ή τροποποίηση*

→ *Εξοικονόμηση χρόνου*

Συναρτήσεις & δομημένος προγραματισμός

Δομημένος προγραμματισμός

↓

Σχεδιασμό της επίλυσης του προβλήματος

ΠΩΣ ?

Καταρχήν με λεκτική περιγραφή

Έστω το πρόβλημα διαχείρισης μιας λίστας ονομάτων και διευθύνσεων

Συναρτήσεις & δομημένος προγραματισμός

Γενική ιδέα λειτουργίας του προγράμματος

- *να εισάγει νέα ονόματα & διευθύνσεις*
- *να τροποποιεί υπαρκτές καταχωρήσεις*
- *να ταξινομεί καταχωρήσεις*
- *να εκτυπώνει ετικέτες αλληλογραφίας*

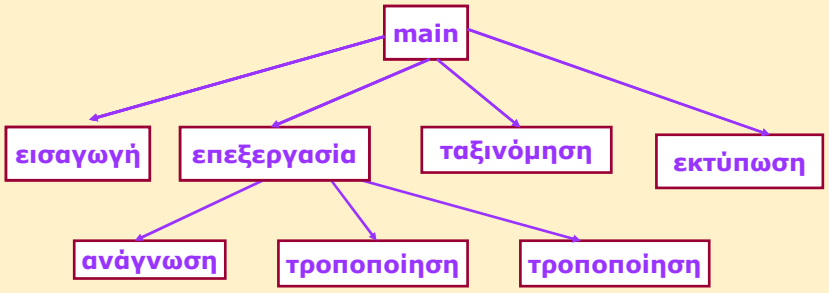
↳ **Τέσσερες κύριες εργασίες**

Συναρτήσεις & δομημένος προγραματισμός

Ο σχεδιασμός οδηγεί σε μια ιεραρχική

ή

διαστρωματωμένη δομή



Συναρτήσεις & δομημένος προγραματισμός

Σχεδίαση του δομημένου προγραμματισμού

Καθοδική προσέγγιση

- *μικρό όγκο κώδικα στο κυρίως πρόγραμμα (**main**)*
- *το σημαντικότερο μέρος του προγράμματος εκτελείται στις συναρτήσεις*