Τύποι, Τελεστές και παραστάσεις στη C

Μάθημα 2

*** Types, operators and expressions**

Απλοί Τύποι Δεδομένων
Ακέραιοι

	Μέγεθος – Bytes	min	max
char	1	-128	127
unsigned char	1	0	255
int	4	-2.147.483.648	2.147.483.647
unsigned int	4	0	4.294.967.295
short int	2	-32.768	32.767
unsigned short int	2	0	65536
long int	4	-2.147.483.648	2.147.483.647

Εξαρτάται από το σύστημα του Η/Υ

- Ο τύπος `char` μπορεί να απεικονιστεί στον πίνακα χαρακτήρων ASCII και να δηλώνει ένα σύμβολο
- Μπορεί να χρησιμοποιηθεί ο προσδιορισμός "`signed`" για να δηλώσει ότι ο τύπος είναι προσημασμένος (για τους περισσότερους μεταγλωττιστές δεν απαιτείται)
- Όταν ένας ακέραιος δηλώνεται άμεσα στο πρόγραμμα ως σταθερά, τότε θεωρείται τύπου int. Αν θέλουμε να χρησιμοποιηθεί από τη γλώσσα ως unsigned τότε προσθέτουμε το χαρακτήρα u ή U μετά τον αριθμό (πχ 100u), για long χρησιμοποιούμε το l ή L.

Απλοί Τύποι Δεδομένων
Μορφοποίηση και έξοδος με την printf

`printf`: συνάρτηση της **`stdio`**

Σύνταξη : `printf` (*κείμενο με σύμβολα μορφοποίησης, μεταβλητές*)

Format code: %«τύπος αριθμού»

```
char a='W';
int k=-100;
unsigned int l=1500;
printf("character %c \n",a);
printf("signed integer %d \n",k);
printf("unsigned integer %u \n",l);
printf("the symbol %% \n");
printf("All numbers  a=%d \t k=%d \t l=%u \n",a,k,l);
```

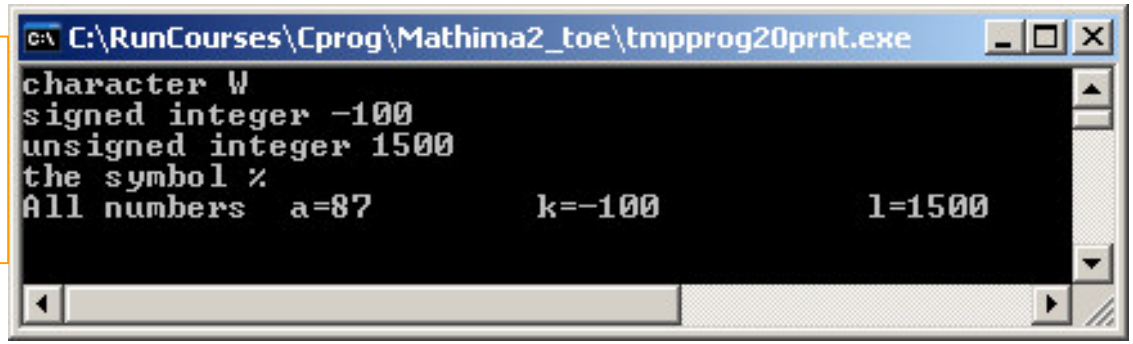
Για ακεραίους

%c : χαρακτήρας

%d : προσημασμένος ακέραιος

%u : απρόσημος

*Format code: %**n**«τύπος αριθμού»
n: ελάχιστος αριθμός ψηφίων
(γέμισμα με κενά πριν τον αριθμό μέχρι να συμπληρωθεί ο αριθμός ψηφίων n)



Δες κώδικα prog20prnt.c

Απλοί Τύποι Δεδομένων
Μορφοποιημένη είσοδος με την scanf

`scanf`: συνάρτηση της **`stdio*`**

Σύνταξη : `scanf` (*σύμβολο μορφοποίησης*, *διεύθυνση μεταβλητής*)

Format code: %«τύπος αριθμού», &: Διεύθυνση μεταβλητής

```
int a;
printf("Give an integer number :");
scanf ("%d",&a)
```

```
/* prog20scanf.c , ver 2.*/
/* Μορφοποίηση και είσοδος ακεραίων */

#include <stdio.h>

main()
{
        int k;
        unsigned int m;
        printf("Give an integer : ");
        scanf("%d",&k);
        printf("Give an unsigned integer : ");
        scanf("%u",&m);
        printf("\n You gave the numbers %d and %u\n",k,m);
        fflush(stdin); getchar();
}
```

*** Περισσότερα για την printf και scanf στο 6ο μάθημα**

Απλοί Τύποι Δεδομένων
Αριθμοί κινητής υποδιαστολής

Μορφή : δεκαδική (πχ 123.456) ή εκθετική (1.23456E02)

	Μέγεθος – Bytes	min	max	Απόλυτο min (≠0)
float	4	-3.402823E38	3.402823E38F	1.175494E-38
double	8	-1.797693E308	1.797693E308	2.225074E-308
long double	12	-1.189731E4932	1.189731E4932	3.362103E-4932

Εξαρτάται από το σύστημα του Η/Υ

(το μηδέν αναγνωρίζεται)

float : αριθμός απλής ακρίβειας (8 σημαντικά ψηφία)
double : αριθμός διπλής ακρίβειας (16 σημαντικά ψηφία)
long double : αριθμός εκτεταμένης ακρίβειας (20 σημαντικά ψηφία)

▪ Όταν ένας πραγματικός δηλώνεται άμεσα στο πρόγραμμα ως σταθερά, τότε θεωρείται τύπου `double`. Αν θέλουμε να χρησιμοποιηθεί από τη γλώσσα ως `float` τότε προσθέτουμε το χαρακτήρα `f` ή `F` μετά τον αριθμό (πχ `1.2f`), για long double χρησιμοποιούμε το `l` ή `L`.

Απλοί Τύποι Δεδομένων
Μορφοποίηση-έξοδος πραγματικών αριθμών

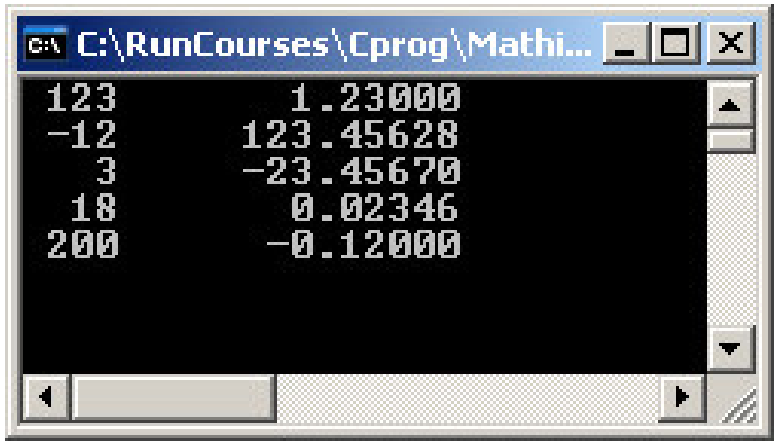
`printf` (*κείμενο με σύμβολα μορφοποίησης*, *μεταβλητές*)

Format code: `%n1.n2f` ή `%n1.n2e`

n1: συνολικός αριθμός ψηφίων, **n2**: αριθμός δεκαδικών
(Προαιρετικά και τα δύο ή το ένα από τα δύο)

Δες κώδικα prog21prnt.c

```
int a1=123, a2=-12, a3=3, a4=18, a5=200;
float x1=1.23, x2=123.4562789, x3=-23.4567,
x4=0.0234578, x5=-0.12;
printf("%4d    %10.5f\n",a1,x1);
printf("%4d    %10.5f\n",a2,x2);
printf("%4d    %10.5f\n",a3,x3);
printf("%4d    %10.5f\n",a4,x4);
printf("%4d    %10.5f\n",a5,x5);
```



Απλοί Τύποι Δεδομένων

Μορφοποιημένη είσοδος Πραγματικών με την scanf

Σύνταξη : λειτουργεί όπως και με τους ακεραίους αλλά με το σωστό σύμβολο μορφοποίησης

Format code: %«τύπος αριθμού», &: Διεύθυνση μεταβλητής

```
float a; double y
Scanf("%f",&x); Scanf("%lf",&y);
```

Για float Για double

```
/* prog20Fscanf.c , ver 2.*/
/* Μορφοποίηση και είσοδος πραγματικών */

#include <stdio.h>
/* δεν έχει σημασία η μορφοποίηση f ή e για floats*/
main()
{
        float a,b;
        double c;
        printf("Give a float : ");
        scanf("%f",&a);
        printf("Give a float : ");
        scanf("%e",&b);
        printf("Give a double : ");
        scanf("%lf",&c);
        printf("\n You gave the numbers %f, %f  and %f\n",a,b,c);
        fflush(stdin); getchar();
```

* Περισσότερα για την printf και scanf στο 6ο μάθημα

Απλοί Τύποι Δεδομένων - Δήλωση σταθερών

1. Χρήση της οδηγίας του προ-μεταγλωττιστή #define, πχ

```
#define Pi 3.14159
```

Κατά την πρώτη ανάγνωση (pre-compiling) κάθε όνομα ``μεταβλητής'' `Pi` μέσα στο πρόγραμμα αντικαθίσταται με το 3.14159

2. Χρήση του προσδιορισμού **const**, πχ

```
const float Pi=3.14159
```

Το σύμβολο `Pi` δηλώνεται ως μεταβλητή τύπου float, παίρνει τιμή κατά την δήλωσή της και η οποία δεν μπορεί να αλλάξει άμεσα (με τελεστή ανάθεσης τιμής στη μεταβλητή

3. Σταθερές απαρίθμησης (enumeration constants) : λίστα μεταβλητών με ακέραιες τιμές ή χαρακτήρες

```
enum studentId {Nick=15, Anna=18, John=10, Maria=60};
```

*αν δεν δηλώνονται τιμές δίνονται με αύξηση κατά 1 από την προηγούμενη δήλωση (αν δεν υπάρχει καμία δήλωση τιμής δίνονται τιμές αρχίζοντας από το μηδέν), πχ

```
enum Color {red=1, green, blue, brown, yellow};
```

(green=2, blue=3 κλπ)

•Οι σταθεροί ακέραιοι αριθμοί στο ***οκταδικό σύστημα*** δίνονται αρχίζοντας με **0**, πχ 0123 (=83)
•Οι σταθεροί ακέραιοι αριθμοί στο ***δεκαεξαδικό σύστημα*** δίνονται αρχίζοντας με **0x**, πχ 0x012A (=298)

•Κωδικοί μορφοποίησης για την printf : `%o` (για οκταδικό) `%x` για δεκαεξαδικό

Δες κώδικα prog22const.c

Μετονομασία-Αλλαγή τύπου

Μετονομασία τύπου : `typedef` *type_name new_name*

Αλλαγή τύπου τιμής μεταβλητής μέσα στο πρόγραμμα (**casting**)
- (type)Μεταβλητή -

```
/* prog23chtyp.c */

/* new type definition and casting */

#include <stdio.h>

typedef double  MyType;
typedef unsigned char BYTE;

main()
{
        MyType a=123.65;
        BYTE b=200;
        printf("a=%f\n",a);
        printf("b=%d\n",b);
        printf("integer a = %d\n",(int)a);
        getchar();
}
```

Casting πραγματικού σε ακέραιο = αποκοπή δεκαδικού μέρους

* Πως μπορούμε να κάνουμε στρογγυλοποίηση στον κοντινότερο ακέραιο με casting;

Αριθμητικοί τελεστές

(μεταβλητή **a**) **Τελ** (μεταβλητή **b**) → τιμή **c**

+ , - , * , /, %

- Αν οι a και b είναι ίδιου τύπου τότε και η c είναι του ίδιου τύπου
- Αν οι a και b είναι διαφορετικού τύπου, η c είναι του «ευρύτερου τύπου» από τις a και b

```
char<int<long int<float<double<long double
```

- %: πράξη ακέραιο υπόλοιπο (πρέπει a,b να είναι ακέραιου τύπου)
- Αν a,b είναι ακέραιου τύπου η διαίρεση / είναι ακέραια διαίρεση, δηλαδή δίνει το πηλίκο.
- Αν θέλουμε να διαιρέσουμε τους δύο ακέραιους a,b με κανονική διαίρεση τότε μπορούμε να εφαρμόσουμε τις ακόλουθες εκφράσεις

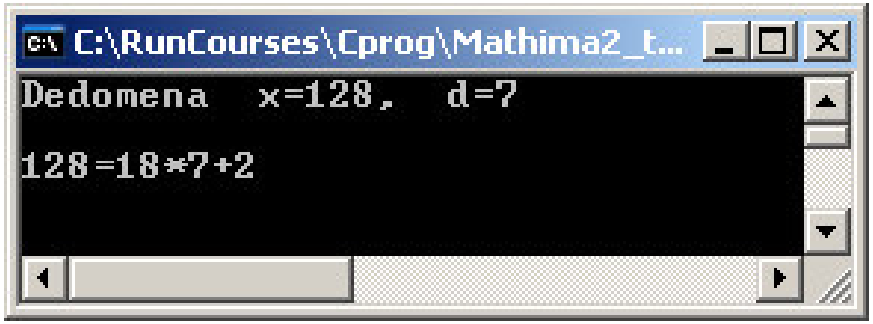
```
(1.0f*a)/b  ή  (float)a/b
```

Δες κώδικα prog24oper.c

Round off errors : παράδειγμα στον κώδικα ***prog25roff.c***

Αριθμητικοί τελεστές – Ασκήσεις

<u>Άσκηση 3.1</u>: Κάντε ένα πρόγραμμα στο οποίο να δίνεται μια τιμή στην ακέραια μεταβλητή **`x`** (διαιρετέος) και μια τιμή στην ακέραια μεταβλητή **`d`**. Θέλουμε το πρόγραμμα να αναλύει τον **`x`** και να τον εκτυπώνει στη μορφή **`x=π*d+υ`**, όπου **`π`** το πηλίκο και **`υ`** το υπόλοιπο.



<u>Άσκηση 3.2</u>

Συμπληρώστε το πρόγραμμα ***prog25roff.c*** με μερικά ακόμα βήματα υπολογισμών και υπολογίστε σε κάθε βήμα πως αυξάνει η διαφορά μεταξύ των τριών τιμών, υπολογίστε δηλαδή τις διαφορές (z-x) και (z-y). Αν υποθέσουμε ότι το z (long double) είναι το σωστό αποτέλεσμα των υπολογισμών, τότε οι παραπάνω διαφορές θα δείχνουν το σφάλμα όταν χρησιμοποιούμε αριθμούς float και double που είναι μικρότερης ακρίβειας.

Τελεστές μείωσης/αύξησης

`--` , `++` , `+=` , `-=` , `*=` , `/=`

* Έστω `x` μεταβλητή και `a` μια τιμή (*σταθερά, μεταβλητή, ή παράσταση*)

`x+=a` : αύξηση της μεταβλητής `x` κατά `a` `(x=x+a)`
`x-=a` : μείωση της μεταβλητής `x` κατά `a` `(x=x-a)`
`x*=a` : πολλαπλασιασμός της μεταβλητής `x` με το `a` `(x=x*a)`
`x/=a` : πολλαπλασιασμός της μεταβλητής `x` με το `a` `(x=x/a)`

Μοναδιαίοι τελεστές μείωσης/αύξησης

`x++` ή `++x` : αύξηση της μεταβλητής `x` κατά 1 `(x=x+1)`
`x--` ή `--x` : μείωση της μεταβλητής `x` κατά ΄1 `(x=x-1)`

- Όταν το ++ ή -- είναι μετά την μεταβλητή, η αύξηση γίνεται μετά τους υπολογισμούς της παράστασης και την εκτέλεση της τρέχουσας εντολής.
- Όταν το ++ ή -- είναι πριν την μεταβλητή, η αύξηση γίνεται πριν γίνουν οι υπόλοιποι υπολογισμοί της παράστασης και η εκτέλεση της τρέχουσας εντολής.

Δες κώδικα prog24pp.c

Συσχετιστικοί και λογικοί τελεστές

*Τιμές **boolean*** : **FALSE** or **TRUE** ή αντίστοιχα **0** ή **1**

- Μια τιμή ***bool*** αντιστοιχεί σε τύπο int (4 bytes!)
- Κάθε μη μηδενικός ακέραιος, ο οποίος αντιμετωπίζεται μέσα στο πρόγραμμα σαν bool, αντιστοιχεί σε τιμή TRUE.

Συσχετιστικοί τελεστές : >, >=, <, <=, = =, != →αποτέλεσμα ***bool***

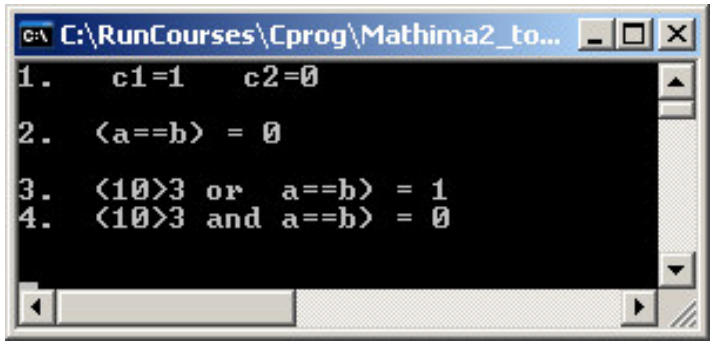
Λογικοί τελεστές : OR (||), AND (&&) και NOT (!)

$!1 \rightarrow 0$
$!0 \rightarrow 1$

$0 \| 0 \rightarrow 0$ $\quad 0 \&\& 0 \rightarrow 0$
$0 \| 1 \rightarrow 1$ $\quad 0 \&\& 1 \rightarrow 0$
$1 \| 0 \rightarrow 1$ $\quad 1 \&\& 0 \rightarrow 0$
$1 \| 1 \rightarrow 1$ $\quad 1 \&\& 1 \rightarrow 1$

```c
#include <stdio.h>

main()
{
        int a=100, b=-30;
        int c1,c2;   /* use as boolean */
        c1=a>b;
        c2=!c1;
        printf("1.    c1=%d    c2=%d\n\n",c1,c2);
        printf("2.   (a==b)  = %d\n\n",a==b);
        printf("3.   (10>3 or   a==b)  = %d \n",10>3||a==b);
        printf("4.   (10>3 and a==b)  = %d \n\n",10>3&&a==b);
        getchar();
}
```



Δες κώδικα prog26bool.c

Παραστάσεις

Τελεστές (λογικοί και αριθμητικοί), σταθερές και μεταβλητές μπορούν να σχηματίσουν σύνθετες παραστάσεις οι οποίες αντιστοιχούν σε μία τιμή (αριθμητική ή bool).

Οι υπολογισμοί μέσα στην παράσταση γίνονται με συγκεκριμένη προτεραιότητα

- Ισχύει γενικά η προσεταιριστικότητα από τα αριστερά (αλλά με εξαιρέσεις για τους τελεστές !, ++,--, +=,- =, *=, /=)
- Πρώτα εκτελούνται οι παρενθέσεις
- Για τις αριθμητικές παραστάσεις προτεραιότητα έχουν οι *, / και % και μετά οι + και -.
- Για τις λογικές παραστάσεις προηγούνται οι συσχετικοί τελεστές <,<=,>,>= των ==, != και έπονται οι λογικοί τελεστές && και ||.

•Σε κάποιες περιπτώσεις (ειδικών τελεστών) η προτεραιότητα μπορεί να εξαρτάται και από τον μεταγλωττιστή ή το ίδιο το hardware.
•**Συνίσταται** όπου υπάρχει αμφιβολία προτεραιότητας να χρησιμοποιούνται παρενθέσεις ή/και περισσότερες γραμμές κώδικα πχ.

Αντί του

```
int x,y,a=9,b=10;
y=a>5+3&&a+1==b++;
```

γράφουμε

```
int x,y,a=9,b=10;
y=(a>(5+3))&&((a+1)==b);
b++;
```

Δες κώδικα prog27expr.c

Παραστάσεις υπό συνθήκη

Τριαδικός τελεστής «**?:**»

Μεταβλητή = [λογική συνθήκη bool] ? παράσταση 1 : παράσταση 2

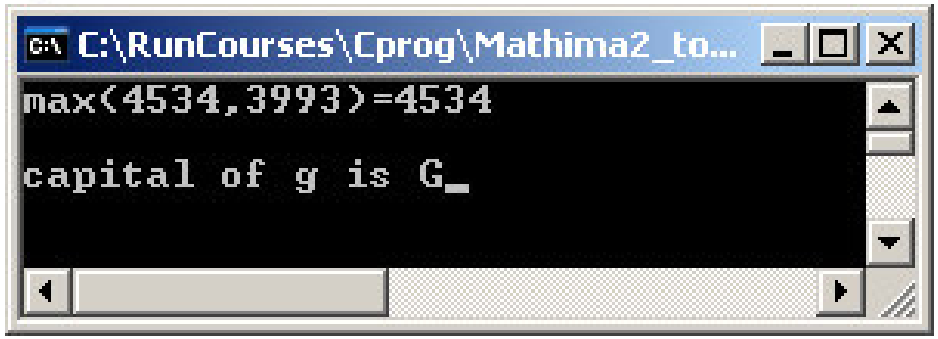
Αν η λογική συνθήκη έχει τιμή TRUE τότε η μεταβλητή παίρνει τη τιμή της παράστασης 1 αλλιώς παίρνει τη τιμή της παράστασης 2. Κάθε φορά μία από τις παραστάσεις υπολογίζεται.

(prog28cond.c)

```
/* find max */
int a, b, max;

a=4534; b=3993;
max=(a>=b)?a:b;
printf("max(%d,%d)=%d\n\n",a,b,max);

/* change lower characters to capital */
const int d=97-65;
char c='g';
printf("capital of %c is ",c);
c=(c<=90)?c:c-d;
printf("%c",c);
```



Συνήθεις μαθηματικές συναρτήσεις

Οι μαθηματικές συναρτήσεις βρίσκονται στην καθιερωμένη βιβλιοθήκη της C και δηλώνονται στη κεφαλίδα `<math.h>`

	Δυνάμεις - Λογάριθμοι					
sqrt	Τετραγωνική ρίζα					
pow	Δύναμη		**exp**	Εκθετική συνάρτηση		
log10	Δεκαδικός Λογάριθμος		**log**	Φυσικός λογάριθμος		
	Τριγωνομετρικές συναρτήσεις (γωνία σε rad)					
cos	συνημίτονο	*(τόξα)*	**acos**	$[0,\pi]$		
sin	ημίτονο		**asin**	$[-\pi/2,\pi/2]$		
tan	εφαπτομένη		**atan**	$[-\pi/2,\pi/2]$		
			atan2	$[0,2\pi)$		
cosh, sinh, tanh : υπερβολικές τριγωνομετρικές συναρτήσεις						
fabs	απόλυτη τιμή πραγματικού αριθμού					
fmod	Υπόλοιπο (για πραγματικούς αριθμούς)					
floor, ceil	συναρτήσεις στρογγυλοποίησης προς τον μικρότερο/μεγαλύτερο ακέραιο					

Συνάρτηση(x)

- x: πραγματική μεταβλητή ή παράσταση (αν δεν είναι τύπου double, μετατρέπεται αυτόματα σε double)
- Δίνει τιμή πραγματικό αριθμό τύπου double

`pow(x,y)` = x^y

`atan2(x,y)=atan(y/x)` με πληροφορία τεταρτημορίου

`fmod(x,y)=R`, ($x=y*k+R$, k:integer)

Ασκήσεις

3.3. Ορίστε μια πραγματική μεταβλητή x και υπολογίστε τις παραστάσεις

$$y = \sin^2(x) + \cos^2(x) \qquad y = e^{9\ln|x|} - x^9, \qquad y = 1 + \frac{x}{2^3} + \frac{|x|^{1/3}}{\sqrt{2}}\left(e^x - \frac{1}{x^2 + \frac{1}{x}}\right)$$

3.4. Δώστε σε ένα πρόγραμμα τις καρτεσιανές συντεταγμένες x,y και υπολογίστε τις πολικές r και θ, όπου θ σε μοίρες. Και το αντίστροφο, δώστε τις πολικές και υπολογίστε τις καρτεσιανές

3.5. Δώστε τρεις μεταβλητές a,b,c και γράψτε μια παράσταση (υπό συνθήκη) που να υπολογίζει τον μεγαλύτερο

3.6. Γράψτε μια παράσταση που να υπολογίζει την στρογγυλοποίηση ενός πραγματικού αριθμού x με δύο δεκαδικά (πχ για τον x=3.14678 να παίρνουμε τον αριθμό y=3.15).