Η ΓΛΩΣΣΑ C

Η C είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου η οποία αναπτύχθηκε στις αρχές της δεκαετίας του ’70 από τον Dennis Ritchie στα Bell Labs.
Η σημερινή μορφή της γλώσσας ακολουθεί το πρότυπο ANSI (American National standards Institute) γι αυτό και ονομάζεται “ANSI C”
Ως γλώσσα υψηλού επιπέδου παρουσιάζει τα εξής πλεονεκτήματα:

•
Αναγνωσιμότητα
Τα προγράμματα διαβάζονται εύκολα.
•
Συντήρηση
Τα προγράμματα είναι εύκολο να συντηρηθούν.
•
Μεταφερσιμότητα
Τα προγράμματα μεταφέρονται εύκολα σε διαφορετικά λειτουργικά συστήματα

Κάθε γλώσσα υψηλού επιπέδου χρειάζεται έναν μεταγλωττιστή (compiler) ο οποίος μεταφράζει τις γραμμές ενός προγράμματος σε γλώσσα μηχανής ώστε να γίνεται κατανοητό από τον υπολογιστή και να εκτελείται

Ένα απλό πρόγραμμα σε C.

```
/* This is my first C program */

#include <stdio.h>

int main()
{
      printf("This is a C program\n");
      return(0);
}
```

Επεξήγηση του προγράμματος

1 Πως γράφω σχόλια.

Η πρώτη γραμμή περιέχει ένα σχόλιο:
/* This is my first C program */

2 Η εντολή #include και τα αρχεία επικεφαλίδων.

Η δεύτερη γραμμή του προγράμματος είναι η ακόλουθη: #include<stdio.h>
Η γραμμή αυτή ξεκινά με ένα #, το οποίο ακολουθείται από την εντολή include. Το include είναι μια εντολή του προεπεξεργαστή η οποία ψάχνει να βρει το αρχείο stdio.h. Επί πλέον η εντολή include ζητά από τον προεπεξεργαστή να τοποθετήσει το αρχείο stdio.h στη θέση της εντολής μέσα στο πρόγραμμα.

Τα αρχεία τα οποία περιλαμβάνονται με την εντολή #include, όπως το stdio.h, ονομάζονται αρχεία επικεφαλίδων και περιέχουν συναρτήσεις έτοιμων προς χρήση για τον προγραμματιστή.. Εκτός από το stdio.h υπάρχουν και άλλα αρχεία επικεφαλίδων, όπως τα

math.h, stdlib.h, string.h κτλ. Κάθε φορά ο προγραμματιστής πρέπει να περιλαμβάνει τα απαραίτητα αρχεία επικεφαλίδων στην αρχή του κώδικά του.

3 Η συνάρτηση main() αποτελεί μια ειδική συνάρτηση της C.
Κάθε πρόγραμμα πρέπει να περιέχει μία και μόνο μία συνάρτηση main() η οποία εκτελείται πάντα πρώτη ακόμη και εάν είναι στο κάτω μέρος του προγράμματος. Στο συγκεκριμένο παράδειγμα η συνάρτηση main έχει δηλωθεί ως ακέραιου τύπου (int) ενώ δεν περιέχει κάποιο όρισμα

4 Η συνάρτηση printf() εμφανίζει και τυπώνει μηνύματα στην οθόνη μας. Ο χαρακτήρας \n ο οποίος εμφανίζεται στο τέλος του μηνύματος λέει στον υπολογιστή να μετακινήσει τον δρομέα στην αρχή της επόμενης γραμμής.

Η συνάρτηση printf() ορίζεται στο αρχείο επικεφαλίδας stdio.h.

5 Η εντολή return.
Όλες οι συναρτήσεις της C μπορούν να επιστρέφουν τιμές. Στο συγκεκριμένο παράδειγμα η συνάρτηση main η οποία έχει δηλωθεί ως ακέραια συνάρτηση επιστέφει την τιμή μηδέν και με αυτόν τον τρόπο το πρόγραμμα τερματίζεται κανονικά.

ΑΝΑΛΥΤΙΚΑ ΓΙΑ ΤΗΝ ΣΥΝΑΡΤΗΣΗ printf()

printf():
Η συνάρτηση **printf()** εμφανίζει στην οθόνη μορφοποιημένα δεδομένα ή/και συνοδευτικό κείμενο.
Η γενική μορφή της συνάρτησης είναι:

printf (ΣΕΙΡΑ_ΕΛΕΓΧΟΥ, v1, v2,... , vn)

Τα " **v1, v2,... , vn** ", είναι ονόματα μεταβλητών, σταθερές ή εκφράσεις.

Η ΣΕΙΡΑ_ΕΛΕΓΧΟΥ περικλείεται μέσα σε διπλά εισαγωγικά (") και καθορίζει τον τρόπο εμφάνισης των δεδομένων.
Η ΣΕΙΡΑ_ΕΛΕΓΧΟΥ μπορεί να περιλαμβάνει:

- **επεξηγηματικό κείμενο**, οτιδήποτε κείμενο θέλουμε
- **προσδιοριστές**, που **μπαίνουν απαραίτητα** μέσα στην ΣΕΙΡΑ_ΕΛΕΓΧΟΥ, μόνο όταν θέλουμε να εμφανίσουμε καποιο στοιχείο (μεταβλητές, σταθερές ή εκφράσεις)
- **χαρακτήρες** ελέγχου, που τους χρησιμοποιούμε για την διαμόρφωση της εμφάνισης

Η **ΣΕΙΡΑ_ΕΛΕΓΧΟΥ** δεν είναι απαραίτητο να περιλαμβάνει όλα τα παραπάνω. Εξαρτάται από το τι θέλουμε να κάνουμε με την **printf()**.

Οι **προσδιοριστές** των στοιχείων τοποθετούνται μέσα στην ΣΕΙΡΑ_ΕΛΕΓΧΟΥ όπου θέλουμε να εμφανιστούν τα δεδομένα, έχουν σχέση με τον τύπο των δεδομένων που εμφανίζονται στην **printf()** και αρχίζουν με τον χαρακτήρα **%**. Ο αριθμός των προσδιοριστών είναι ίσος με τον αριθμό των στοιχείων.

Για κάθε τύπο δεδομένων έχουμε τους εξής προσδιοριστές:

%d	Ακέραιος
%c	Χαρακτήρας/char (1 μονο)
%s	Σειρά χαρακτήρων (συμβολοσειρά)
%f	Κινητής υποδιαστολής, δηλαδή float ή double
%e	Κινητής υποδιαστολής, δηλαδή float ή double σε εκθετική μορφή
%g	Κινητής υποδιαστολής σαν %e ή %f (όποιο είναι μικρότερο)
%u	Ακέραιος χωρίς πρόσημο (unsigned int)
%o	Ακεραιος σε οκταδικό σύστημα (χωρίς πρόσημο)
%x	Ακέραιος σε δεκαεξαδικό σύστημα

Συχνά οι προσδιοριστές συνοδεύονται με αριθμούς που ορίζουν το εύρος της εμφάνισης του δεδομένου. Για παράδειγμα το **%6d**, σημαίνει ότι το δεδομένο είναι ακέραιος και ότι το εύρος του είναι 6 ψηφία (χαρακτήρες), ενώ το **%7.2f,** σημαίνει ότι το δεδομένο είναι κινητής υποδιαστολής και έχει εύρος 7 ψηφία με 2 ψηφία μετά την υποδιαστολή.

Οι **χαρακτήρες ελέγχου** χρησιμοποιούνται για τον διαμόρφωση της εμφάνισης και για την εκτύπωση των δεσμευμένων από την ΣΕΙΡΑ_ΕΛΕΓΧΟΥ χαρακτήρων (όπως τα εισαγωγικά). Ξεκινούν με την ανάποδη πλάγια κάθετο \ (backslash) και είναι:

\a	Ηχητικό σήμα (<BELL>)
\b	Ο χαρακτήρας <BACKSPACE> (διάστημα πίσω)
\f	Ο χαρακτήρας νέας σελίδας (<FORM FEED>)
\n	Νέας γραμμής (<LINE FEED>)
\r	Επιστροφής (<CR>)
\t	Οριζοντίου προκαθορισμένου διαστήματος (<TAB>)
\v	Κατακορύφου διαστήματος(<VTAB>)
\'	Εμφάνιση του απλού εισαγωγικού
\"	Εμφάνιση του διπλόυ εισαγωγικού
\\	Εμφάνιση της ανάποδης πλαγίας καθέτου
\?	Εμφάνιση του λατινικού ερωτηματικού
\xhhh	Εμφάνιση του χαρακτήρα hhh σε δεκαεξαδικό αριθμό
\ooo	Εμφάνιση του χαρακτήρα ooo σε δεκαεξαδικό αριθμό

Η **printf()** σαν συνάρτηση που είναι, επιστρέφει το αριθμό των χαρακτήρων που τυπώνονται.

<u>Παραδείγματα:</u>

1)Δίνονται οι κάτωθι εντολές .Τι τυπώνει το συγκεκριμένο κομμάτι του προγράμματος ?
```
metavliti=5;
printf(" %d", metavliti);
```

<u>Τυπώνει:</u>
5

2)Δίνονται οι κάτωθι εντολές .Τι τυπώνει το συγκεκριμένο κομμάτι του προγράμματος ?

```
metavliti=5;
printf("The value of variable metavliti is: %d", metavliti);
```

Τυπώνει:

The value of variable metavliti is:5

3)Δίνονται οι κάτωθι εντολές .Τι τυπώνει το συγκεκριμένο κομμάτι του προγράμματος ?

```
metavliti=5;
printf("The value of variable metavliti is:\n %d", metavliti);
```

Τυπώνει:

The value of variable metavliti is:
5

4)Δίνονται οι κάτωθι εντολές .Τι τυπώνει το συγκεκριμένο κομμάτι του προγράμματος ?

```
x=5;
y=3;
printf("%d + %d = %d", x, y, (x+y));
```

Τυπώνει:

5+3 = 8

5)Δίνονται οι κάτωθι εντολές .Τι τυπώνει το συγκεκριμένο κομμάτι του προγράμματος ?

```
x=5;
y=3;
printf("x=%d /n y=%d /n x+y= %d", x, y, (x+y));
```

Τυπώνει:

x=5
y=3
x+y=8

6)Δημιουργήστε ένα πρόγραμμα που θα εμφανίζει στην οθόνη του υπολογιστή τα παρακάτω κατά σειρά:

- Τον ακέραιο αριθμό 5
- Τον πραγματικό αριθμό 5
- Τον πραγματικό αριθμό 5.01234567890123456789 σε απλή γραφή
- Τον πραγματικό αριθμό 5.01234567890123456789 σε απλή γραφή με δύο δεκαδικά ψηφία

- Τον χαρακτήρα N

Και η εξοδος να είναι η εξής:

O akeraios arithmos x exei timh: 5

O pragmatikos arithmos y exei timh: 5.000000

O pragmatikos arithmos z exei timh: 5.012346

O pragmatikos arithmos z exei timh: 5.01

O xaraktiras einai o N

ΠΡΟΓΡΑΜΜΑ

```
#include <stdio.h>
void main()
{
int x=5;
float y=5;
float z=5.01234567890123456789;
char w='N';
printf("O akeraios arithmos x exei timh: %d\n",x);
printf("O pragmatikos arithmos y exei timh: %f\n",y);
printf("O pragmatikos arithmos z exei timh: %f\n",z);
printf("O pragmatikos arithmos z exei timh: %.2f\n",z);
printf("O xaraktiras einai o %c\n", w);
}
```

7)Δημιουργήστε ένα πρόγραμμα που θα ζητά από τον χρήστη δύο ακέραιους αριθμούς και θα εκτυπώνει στην οθόνη:

- Το άθροισμά τους
- Την διαφορά τους
- Το γινόμενό τους.

ΠΡΟΓΡΑΜΜΑ

```
#include <stdio.h>
void main()
{
int x;
int y;
printf("Dwse enan akeraio arithmo x:\n");
scanf("%d", &x);
printf("Dwse enan akomi akeraio arithmo y:\n");
scanf("%d", &y);
printf("x+y= %d\n",x+y);
printf("x-y= %d\n",x-y);
```

printf("x*y= %d\n",x*y);

ΑΝΑΛΥΤΙΚΑ ΓΙΑ ΤΗΝ ΣΥΝΑΡΤΗΣΗ scanf ()

scanf():
Διαβάζει από το πληκτρολόγιο μορφοποιημένες τιμές μεταβλητών.
scanf (ΣΕΙΡΑ_ΕΛΕΓΧΟΥ, &v1, &v2,... , &vn)

Η συνάρτηση **scanf()** μοιάζει πολύ με την **printf()** με μιά μεγάλη διαφορά. Όπως βλέπετε στην περιγραφή της δεν αναφέρεται σε πραγματικές μεταβλητές, αλλά σε. διευθύνσεις μεταβλητών (Όταν δηλώνεται μία μεταβλητή σ' ένα πρόγραμμα, ουσιαστικά δεσμεύεται στην μνήμη του υπολογιστή μία περιοχή, η οποία θα φιλοξενήσει κατά την διάρκεια της εκτέλεσης του προγράμματος τις τιμές που θα δώσουμε για αυτή την μεταβλητή. Αυτή η περιοχή μνήμης έχει μία συγκεκριμένη διεύθυνση -πρόκειται για κάποιον συνήθως δεκαεξαδικό αριθμό- που είναι η διεύθυνση της μεταβλητής.)

Στην γλώσσα προγραμματισμού C, γιά ν' αναφερθούμε στην διεύθυνση μιάς μεταβλητής χρησιμοποιούμε το όνομα της μεταβλητής προσθέτοντας μπροστά το σύμβολο "**&**". Δηλαδή η έκφραση **&v1** εννοεί την διεύθυνση της μεταβλητής **v1**

Προσέξτε όμως για το διάβασμα μιάς συμβολοσειράς, δεν χρησιμοποιούμε &, διότι το όνομα της μεταβλητής_συμβολοσειράς είναι ουσιαστικά η διεύθυνση του πρώτου χαρακτήρα

Παραδείγματα

1) Διαβάζει από το πληκτρολόγιο την ακέραια μεταβλητή x
printf("Enter first number: ");
scanf("%d", &x);

καλό είναι κάθε κλήση της scanf() να συνοδεύεται από μία κληση της printf(), όπως στο προηγούμενο παράδειγμα (η printf να μπαίνει πρίν την scanf), ώστε ο χρήστης γνωρίζει τι δεδομένο να εισάγει με το πληκτρολόγιο.

2)Διαβάζει από το πληκτρολόγιο μία μεταβλητή κινητής υποδιαστολής (k) και μία ακέραια μεταβλητή (l)

printf ("Give me 1 arithmo kinitis ipodiastolis kai 1 arithmo akereo:");
scanf("%f %d", &k, &l);

3) Τι τιμες παίρνουν τα i,,x όταν γράφω

int i;
float x;

scanf("%d%f", &i, &x);
και δίνω ως γραμμή εισαγωγής : **25 54.324**

Απάντηση
η μεταβλητή **i** να πάρει την τιμή 25 και η μεταβλητή **x** να πάρει την τιμή 54.324

ΑΝΑΛΥΤΙΚΑ ΓΙΑ ΤΗΝ ΣΥΝΑΡΤΗΣΗ puts ()

puts():

Εμφανίζει στην οθόνη μία σειρά χαρακτήρων (συμβολοσειρά).
puts(<συμβολοσειρά>|<μεταβλητή_συμβολοσειράς>);

Είναι πιό απλή και πιό γρήγορη από την **printf()** αλλά χρησιμοποιείται μόνο για συμβολοσειρές (όχι για τις μεταβλητές ή τις σταθερές). Η **puts()** μετά την εμφάνιση της συμβολοσειράς, τυπώνει τον χαρακτήρα **\n**, δηλαδή επιστρέφει στην επόμενη σειρά. Ο χαρακτήρας **\0** που υποδεικνύει το τέλος της συμβολοσειράς χρησιμοποιείται από την **puts()** και καλό θα έιναι να προσέχετε, ώστε η συμβολοσειρά που έχετε δώσει για εκτύπωση να περιέχει στο τέλος τον χαρακτήρα **\0.**

ΑΝΑΛΥΤΙΚΑ ΓΙΑ ΤΗΝ ΣΥΝΑΡΤΗΣΗ gets ()

gets():

Διαβάζει από το πληκτρολόγιο (είσοδος -stdin) μία σειρά χαρακτήρων (συμβολοσειρά).
gets(<μεταβλητή_συμβολοσειράς>);

Ιδιαίτερα χρήσιμη για διαλογική συζήτηση με τον υπολογιστή.

ΠΑΡΑΔΕΙΓΜΑ

```
char name[20];

printf("What's your name:"); gets(name);
printf("Hello "); puts(name);
```

Η **gets()** διαβάζει από το πληκτρολόγιο μέχρι να συναντήσει τον χαρακτήρα **\n** (πράγμα που συμβαίνει όταν πατήσουμε το <ENTER>). Ο χαρακτήρας **\0** μπαίνει αυτόματα στο τέλος της συμβολοσειράς που πληκτρολογείτε.

Παραδείγμα με τις **gets()** και **puts()**:

```
#include <stdio.h>
main()
{
char aaa[20];
char bbb[10] = "Ritsa";
printf("Hello my name is "); puts(bbb);
printf("What's your name:"); gets(aaa);
printf("\n");
printf("%s you 're welcome to our class...",aaa);
}
```

putchar():

Εμφανίζει στην οθόνη ένα μόνο χαρακτήρα.

getchar():

Διαβάζει από το πληκτρολόγιο ένα χαρακτήρα. Η συνάρτηση δεν δέχεται παράμετρους, απλά παίρνει έναν χαρακτήρα από το πληκτρολόγιο και δεν χρειάζεται να δώσουμε <ENTER>.

Και άλλο παράδειγμα με χρήση putchar() και getchar()

Στο πρόγραμμα που ακολουθεί εισάγουμε μια πρόταση η οποία τερματίζεται με enter

και στη συνέχεια εμφανίζεται η πρόταση αυτή και μετράμε τους χαρακτήρες της

```
#include <stdio.h>

void main()

{

        int c,count=0;
```

```
        printf("Dose protasi\n");


        while ((c=getchar())!='\n')//διαβάζουμε τους χαρακτήρες της πρότασης ένα προς
                                                                //ένα μέχρι να δώσουμε το
enter
        {
                        putchar(c);//τυπώνεται ο χαρακτήρας αυτός
                count++;//αυξάνεται ο μετρητής των χαρακτήρων της πρότασης αυτής
        }


        printf("\nPlithos xaraktiron protashs = %d\n",count);
}
```

ΤΕΛΕΣΤΕΣ

Η C όπως και κάθε γλώσσα προγραμματισμού χρησιμοποιεί διάφορους τελεστές για να εκτελέσει πράξεις, να αναπαραστήσει αριθμητικές και λογικές εκφράσεις και να δημιουργήσεις δομές ελέγχου. Ετσι λοιπόν στην C ανα κατηγορία έχουμε τους τελεστές που ακολουθούν.

<u>Ο τελεστής καταχώρησης:</u>

Για το τελεστή καταχώρησης (ή καλύτερα ανάθεσης σε μεταβλητή -assigment) χρησιμοποιείται το σύμβολο του ίσον (=) που έχει βέβαια διαφορετική έννοια απο ότι στα μαθηματικά. Η εντολή:

```
var2 = 52;
```

σημαίνει ότι δίνω την τιμή **52** στην μεταβλητή **var2**. Αν δώσω παρακάτω την εντολή:

```
var2 = var2 +1;
```

που ενώ μαθηματικώς δεν "στέκει", "αυξάνω" κατα μία μονάδα την τιμή που ήδη υπάρχει καταχωρημένη στην μεταβλητή **var2**, δηλαδή η νέα τιμή της **var2** μετά την εκτέλεση της συγκεκριμένης εντολής θα είναι 53!

Φυσικά δεν μπορείτε να καταχωρήσετε τιμή σε μία σταθερά, ενώ στην C μπορώ να κάνω πολλαπλή καταχώρηση. Δηλαδή ισχύει το:

```
x=y=z=54;
```

Εδώ οι καταχωρήσεις γίνονται από δεξιά προς τα αριστερά, δηλαδή πρώτα πάιρνει την τιμή **54** η μεταβλητή **z,** μετά η μεταβλητή **y** και τέλος η **x**.

<u>Οι αριθμητικοί τελεστές:</u>

Προκειται για τους:

+	Πρόσθεση
-	Αφαίρεση
*	Πολαπλασιασμός
/	Διαίρεση
%	Υπόλοιπο (πχ., 25 % 4 μας δίνει 1), καλείται "mod" και εφαρμόζεται μόνο σε ακεραίους.

Για κάθε τελεστή απαιτούνται 2 όροι (εκτός από τον τελεστή "-", που είναι δυνατόν να χρησιμοποιηθεί και σαν πρόσημο), οι οποίοι μπορεί να είναι σταθερές ή μεταβλητές.

```
ilikia=etos1 - etos2;
printf("%d", 95-55);
celsious=(-25);
```

Τα κενά διαστήματα δεν παίζουν ρόλο και μπορείτε να χρησιμοποιήστε παρενθέσεις για μαθηματικές παραστάσεις:

```
result = -(b+2)*a + (c +3*(a-b);
```

Η C ακολουθεί την αλγεβρική προτεραιότητα στους αριθμητικούς τελεστές. Δηλαδή μεγαλύτερη προτεραιότητα έχουν οι παρενθέσεις, μετά το προσημο (δηλαδή το μείον), ακολουθούν ο πολλαπλασιασμός, η διαίρεση και το mod και τέλος οι τελεστές πρόσθεσης και αφαίρεσης. Η εντολή καταχώρησης έχει μικρότερη προτεραιότητα από όλες τις πράξεις. Η φορά των πράξεων είναι από αριστερά προς δεξιά για όλους τους αριθμητικούς τελεστές, εκτός αν χρησιμοποιήστε το μείον σαν πρόσημο, οπότε η φορά είναι από δεξιά προς αριστερά!

Οι τελεστές συσχετισμού:

Τελεστές που τους χρησιμοποιούμε για την δημιουργία απλών λογικών εκφράσεων συσχετισμού (σύγκρισης). Αυτοί είναι:

<	Μικρότερο από (πχ. αν i<j επιστρέφει 1 αν το i είναι μικρότερο από το j)
>	Μεγαλύτερο από
<=	Μικρότερο από ή ίσο
>=	Μεγαλύτερο από ή ίσο
= =	Λογικό ισον
!=	Διάφορο (όχι ίσο)

Το αποτέλεσμα μιάς λογικής έκφρασης είναι είτε **1** (αληθές) είτε **0** (ψευδές). Για παράδειγμα αν έχω **x=2;** και **y=3;** τότε η έκφραση **(x > y)** επιστρέφει **0**. Οι λογικές εκφράσεις χρησιμοποιούνται σαν συνθήκες στις εντολές ελέγχου (και όχι μόνο). Προσέξτε το σύμβολο της ισότητας που είναι ένα διπλό ισον (= =)! Επίσης θα πρέπει να γνωρίζετε ότι στην C η

προτεραιότητα των τελεστών συσχέτισμού είναι μικρότερη από αυτή των αριθμητικών τελεστών, ενώ είναι μεγαλύτερη από αυτή του τελεστή καταχώρησης. Τι τιμή θα πάρει η μεταβλητή **i** στο κώδικα που ακολουθεί;

```
j = k =13;
i = j = = k;
```

Θα πάρει την τιμή **1**, διότι ο τελεστής της ισότητας (λογικό ίσον = =) έχει μεγαλύτερη προτεραιότητα από τον τελεστή καταχώρησης (ένα απλό ίσον =). Αρα λοιπόν, καθώς οι μεταβλητές **j** και **k** είναι ίσες το αποτέλεσμα της σύγκρισης αυτών (j = = k) θα είναι αληθές, δηλαδή **1,** τιμή που θα πάρει η μεταβλητή **i**