Εισαγωγή στην Πληροφορική & στον Προγραμματισμό

Αρχές Προγραμματισμού Η/Υ (με τη γλώσσα C)

Διάλεξη #1

10 & 11 Φεβρουαρίου 2023

Παναγιώτης Παύλου

c-programming-23@allos.gr

Γενικά για το μάθημα

Εσείς και εμείς μαζί

Δομή Μαθήματος

Στο μάθημα υπάρχουν δύο παράλληλες ροές. Για τη ροή του προγραμματισμού με τη γλώσσα C, οι «εργαστηριακές διαλέξεις» πραγματοποιούνται τις Πέμπτες και Παρασκευές στο PC-Lab (Ε.Π.Υ.) σύμφωνα με το πρόγραμμα.

Πριν από κάθε διάλεξη θα πρέπει να έχετε παρακολουθήσει τα **βίντεο προετοιμασίας** που θα αναρτώνται στο helios.

Μετά από κάθε διάλεξη θα δίνονται απλές **εργασίες** για το σπίτι και λίγες ημέρες αργότερα πρέπει να τις **υποβάλλετε ηλεκτρονικά**. Η βαθμολογία τους συμμετέχει στην τελική βαθμολογία του μαθήματος.

Η τελική εξέταση θα γίνεται πάνω σε Η/Υ με ηλεκτρονικό τρόπο, ανάλογα με τα τότε ισχύοντα.

Σκοπός

Ο σκοπός του προγραμματιστικού μέρους του μαθήματος, στο τέλος του εξαμήνου, είναι ο κάθε φοιτητής και η κάθε φοιτήτρια:

- Να έχει **αντίληψη** για τα ζητήματα του προγραμματισμού
- Να μπορεί να **κατανοήσει** τη λειτουργία ενός προγράμματος ή ενός τμήματος κώδικα
- Να μπορεί ο κάθε συμμετέχων να **δημιουργήσει** ένα σχετικά απλό πρόγραμμα
- Να **προετοιμαστεί** για τη χρήση του προγραμματισμού στα υπόλοιπα μαθήματα της Σχολής

Υλικό

Το υλικό το οποίο σας παρέχεται στα πλαίσια του μαθήματος, σας παρέχει όλες τις απαιτούμενες γνώσεις ώστε να το ολοκληρώσετε επιτυχώς, ενώ θα είναι στη διάθεσή σας με διάφορες μορφές:

- Παρουσιάσεις από τις διαλέξεις (σε μορφή PDF)
- Ηλεκτρονική (σε PDF) και έντυπη μορφή του «Εγχειριδίου της Γλώσσας Προγραμματισμού C»
- Αναφορές (links) σε διάφορα site με σχετικό υλικό (ως links μέσα στα PDF)
- Προτεινόμενα σχετικά βιβλία διεθνούς βιβλιογραφίας
  - K & R – The C programming language
  - Deitel-Deitel – C How to Program
- Κοινότητες προγραμματιστών (ενδεικτικά το SO και αυτό)
- Τη σελίδα ερωταποκρίσεων του μαθήματος

Τι θα μάθουμε

Η παραπάνω ύλη θα καλύπτει τουλάχιστον τα εξής αντικείμενα, με τη χρήση της γλώσσας προγραμματισμού C:

- Η πληροφορία στον Η/Υ
  - Αριθμοί
  - Πίνακες / Κείμενα
  - Δομές
- Αλγόριθμοι
  - Διαγράμματα ροής
  - Ροή εκτέλεσης
  - Έλεγχο ροής
- Συναρτήσεις
  - Δημιουργία & χρήση
  - Βιβλιοθήκες συναρτήσεων
- Βασικές (standard) βιβλιοθήκες
  - Διαχείριση μνήμης
  - Ανάγνωση και εγγραφή αρχείων στον Η/Υ
  - Άλλες βιβλιοθήκες συναρτήσεων

Σημεία προσοχής

- Επειδή η περισσότερη ουσιαστική πληροφορία πάνω στον προγραμματισμό είναι στην Αγγλική γλώσσα, παρότι οι διαλέξεις είναι στα Ελληνικά, θα παρουσιάζονται και θα χρησιμοποιούνται οι **Αγγλικοί όροι**
- Ο προγραμματισμός είναι τέχνη γι'αυτό δεν γίνεται να τον μάθει κάποιος διαβάζοντας 1 εβδομάδα πριν τις εξετάσεις. **Απαιτεί παρακολούθηση, εξάσκηση και συζήτηση**
- **Ερωτήσεις** κατά τη διάρκεια του μαθήματος μπορούν να γίνουν σε συγκεκριμένα σημεία.
- Είμαστε **στη διάθεσή σας** για την επίλυση απορίων ή προβλημάτων με την προϋπόθεση ότι αυτή γίνεται αφού έχετε καταβάλει πρώτα τη σχετική προσπάθεια μόνοι σας και αφού έχετε ελέγξει μήπως υπάρχει ήδη απαντημένη η ερώτηση στο σχετικό site.

ΠΡΟΣΟΧΗ! Οι ερωτήσεις αποστέλλονται με e-mail στο:
c-programming-23@allos.gr
ενώ οι απαντήσεις θα δημοσιεύονται στο
https://qna.c-programming.allos.gr

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-23@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε μαζί τον κώδικα και τα μηνύματα λάθους από το CLion ως κείμενα με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Τονίζουμε : Μην στείλετε **ποτέ κώδικα ως εικόνα**, είναι παντελώς άχρηστος!

Γιατί γλώσσες προγραμματισμού

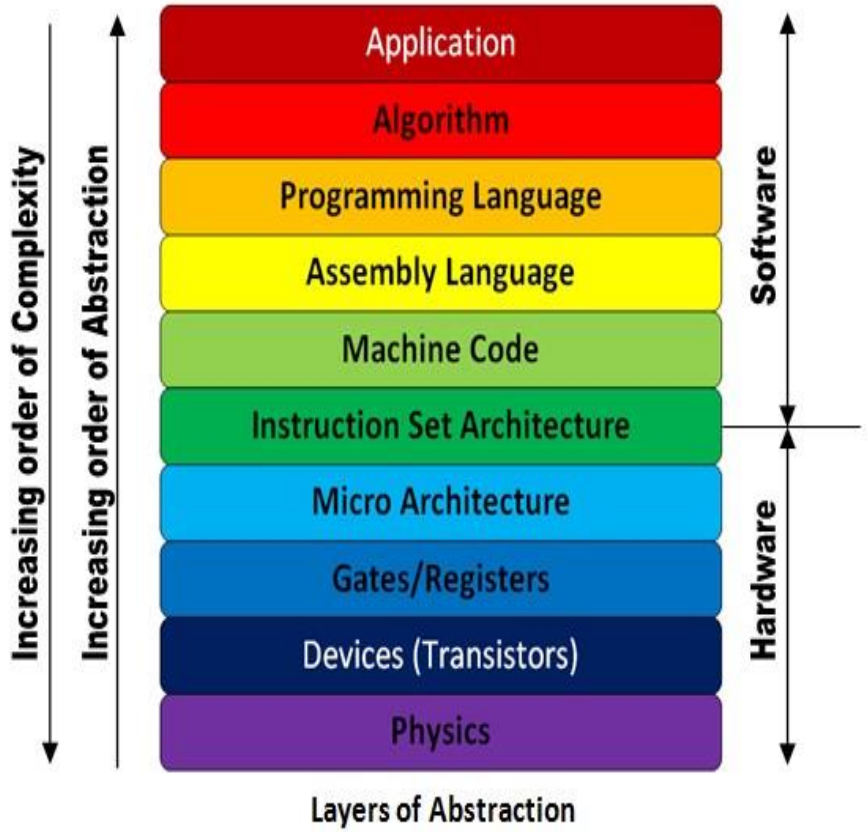
Τι ξεχωρίζει τη γλώσσα C

Αφαιρετικά επίπεδα λειτουργίας

Ο Η/Υ είναι μία συσκευή η οποία εκτελεί εντολές αδιάκοπα. Το λογισμικό (**software**) γράφεται για κάποιο **σκοπό**. Αποτελείται από τις εντολές που καθορίζουν το πώς θα λειτουργήσει η συσκευή (**hardware**), ώστε να επιτευχθεί ο παραπάνω σκοπός.

Τόσο το hardware όσο και το software παραδοσιακά στους Η/Υ οργανώνονται σε διάφορα επίπεδα αφαίρεσης (**abstraction layers**). Η λέξη επίπεδα (layers) δεν χρησιμοποιείται τυχαία αλλά για να τονίσει τη στρωματοποιημένη δομή τους.

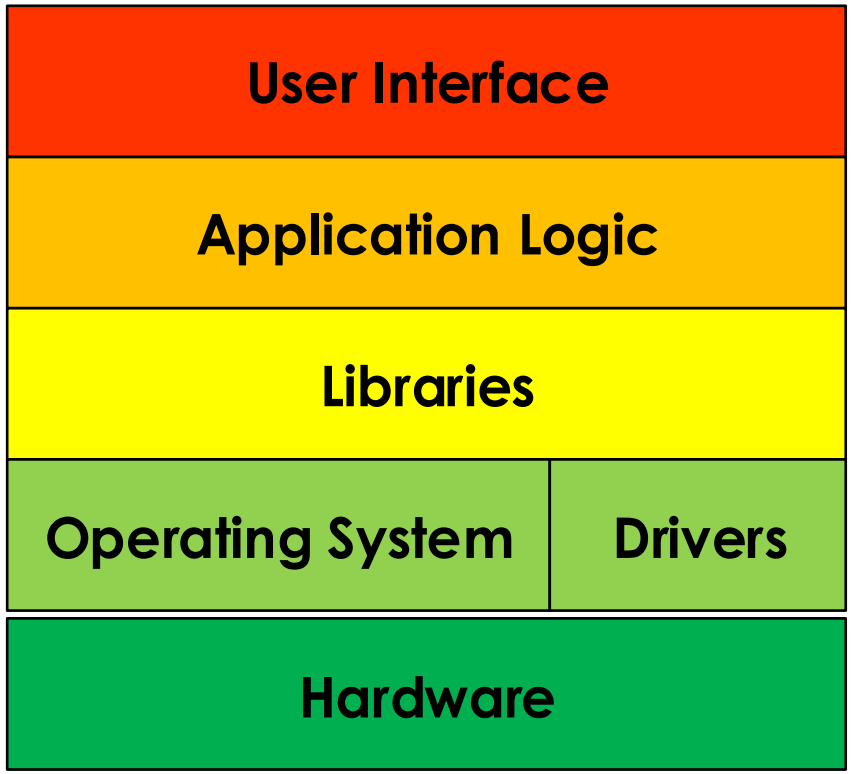
Πιο σημαντικό είναι όμως ο όρος «αφαίρεση» (abstraction) ο οποίος τονίζει ότι κάποιος μπορεί να ασχοληθεί με ένα μόνο layer γνωρίζοντας μόνο την εξωτερική συμπεριφορά των υπολοίπων layers στα οποία βασίζεται. Με τον ίδιο τρόπο που ο οδηγός ενός αυτοκινήτου δεν χρειάζεται να γνωρίζει μηχανολογία.



Αφαιρετικά επίπεδα λειτουργίας

Ως προς τη δομή του software:

- το υψηλότερο είναι η διεπαφή προς τον χρήστη (**user interface**) ή γενικότερα η διεπαφή του με το περιβάλλον
- το χαμηλότερο από αυτά τα επίπεδα είναι το λειτουργικό σύστημα (**operating system**) και οι οδηγοί (**drivers**) που επικοινωνούν απευθείας με το hardware
- όλα τα υπόλοιπα επίπεδα βρίσκονται ανάμεσα σε άλλα επίπεδα αφαίρεσης



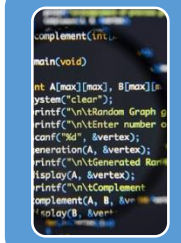
Παραδείγματα

http

- Web σελίδα από τον server στον browser

Αρχεία και φάκελοι

- Το λογισμικό και οι χρήστες ασχολούνται με αρχεία και φακέλους

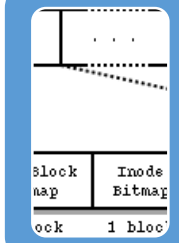


Γλώσσα υψηλού επιπέδου

- Κώδικας σε C

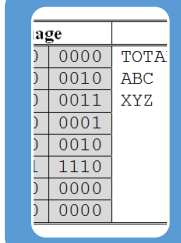
TCP

- Μεταφορά πληροφορίας από Η/Υ σε Η/Υ



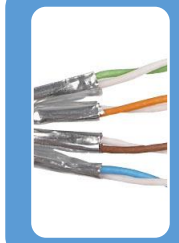
Filesystem

- Η δομή της αποθήκευσης των αρχείων στον δίσκο



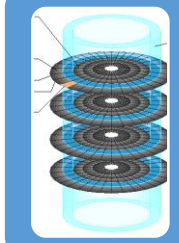
Γλώσσα Μηχανής

- Εντολές «επιπέδου επεξεργαστή»



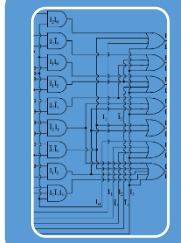
Ethernet

- Αφορά τη φυσική σύνδεση Η/Υ με Η/Υ



Blocks

- Η τμηματοποίηση του δίσκου, ανάλογα με την τεχνολογία του



Λογικά Κυκλώματα

- Λογικές πύλες, Transistors

Γλώσσες προγραμματισμού

Υπάρχουν πάρα πολλές γλώσσες προγραμματισμού. Κάποιες από αυτές βασίζονται σε άλλες προγενέστερες. Κάποιες από αυτές είναι κατάλληλες για συγκεκριμένες εργασίες ενώ άλλες είναι γενικότερης χρήσης. Πασίγνωστες γλώσσες προγραμματισμού είναι οι:

C/C++	Java	JavaScript	Python	FORTRAN
C#	Kotlin	Perl	Dart	Assembly
VisualBasic	Lua	Pascal	PHP	Go

Αυτές που αναφέρονται με κόκκινα γράμματα είναι γλώσσες που έχουν παρόμοιο συντακτικό με τη C. Άρα μαθαίνοντας μία από αυτές γνωρίζουμε το βασικό συντακτικό και για τις υπόλοιπες.

Άλλες γλώσσες του Η/Υ

Υπάρχουν επίσης «γλώσσες» για τον υπολογιστή οι οποίες δεν αφορούν εντολές προς τον επεξεργαστή, άρα δεν αποτελούν γλώσσες προγραμματισμού. Τέτοιες είναι οι:

- HTML, XML και JSON, οι οποίες περιγράφουν δεδομένα
- CSS, που περιγράφει την εμφάνιση δεδομένων (κυρίως HTML)
- XSLT, που περιγράφει μετασχηματισμούς δεδομένων

και πολλές άλλες.

Γλώσσες που ξεχωρίζουν

Ιδιαίτερη σημασία πρέπει να δοθεί στη γλώσσα Assembly η οποία είναι μία «πιο εμφανίσιμη» μορφή της γλώσσας μηχανής. Αυτή είναι η γλώσσα που «μιλάει» ο ίδιος ο επεξεργαστής. **Οποιοσδήποτε κώδικας εκτελείται σε έναν Η/Υ είναι οπωσδήποτε σε γλώσσα μηχανής (όχι assembly).**

Η γλώσσα μηχανής, άρα και η Assembly είναι **διαφορετική** για κάθε τύπο επεξεργαστή. Αυτό καθιστά αρκετά περίπλοκο το να γνωρίζει κάποιος όλες τις **παραλλαγές**. Επίσης δημιουργεί μεγάλο πρόβλημα στο να δημιουργηθεί ένας **ενιαίος** κώδικας που να λειτουργεί σε διαφορετικούς επεξεργαστές και λειτουργικά συστήματα. Να είναι όπως λέμε **cross-platform**.

Τη λύση στα παραπάνω προβλήματα, παραμένοντας ταυτόχρονα όσο το δυνατόν πιο κοντά στη λογική του επεξεργαστή, μας τη δίνει η C.

Γιατί η C?

Η C είναι μία γλώσσα που λίγοι προγραμματιστές δεν θα την μάθουν στη ζωή τους. Μερικά σημαντικά χαρακτηριστικά της είναι:

- Είναι πολύ κοντά στην γλώσσα μηχανής, παρέχει τις ίδιες λειτουργίες
- Αυτό την καθιστά διαχρονική γλώσσα, δεν αναμένεται να εγκαταληφθεί
- Πάνω σε αυτή βασίζονται πολλές σύγχρονες γλώσσες (με πιο γνωστές τις Java, JavaScript, PHP)
- Ένας κώδικας σε C που χρησιμοποιεί τις τυπικές βιβλιοθήκες αναμένεται να λειτουργεί απροβλημάτιστα σε οποιοδήποτε λειτουργικό σύστημα, εφόσον γίνει compile γι'αυτό. Είναι δηλαδή cross-platform.
- Λόγω της ευρείας χρήσης της έχει αρκετές έτοιμες βιβλιοθήκες πέρα από τις τυπικές

Υψηλού επιπέδου γλώσσες προγραμματισμού

Οι γλώσσες προγραμματισμού (εκτός της γλώσσας μηχανής και της Assembly) ονομάζονται και υψηλού επιπέδου γλώσσες με την έννοια ότι αποτελούν ένα abstraction layer **πάνω** από τη γλώσσα μηχανής.

Ο κώδικας σε αυτές τις γλώσσες γράφεται σε απλά αρχεία κειμένου που μπορούν να ανοίξουν ακόμα και με το σημειωματάριο (notepad) των Windows.

Για να εκτελεστούν όμως πρέπει με κάποιον τρόπο να μετατραπούν σε γλώσσα μηχανής. Σε αυτό υπάρχουν τρεις λύσεις που χρησιμοποιούνται:

	Εγκατάσταση στον Η/Υ του		Φορητότητα Κώδικα	Ταχύτητα εκτέλεσης	Ενδεικτικές Γλώσσες
	προγραμματιστή	χρήστη			
Interpreter (διερμηνέας)	Όχι	Ναι	Ναι	Μικρή	JS, PHP, Python, Lua, Perl, Dart
Compiler (μεταφραστής) Ο τελικός κώδικας είναι αυτόνομος	Ναι	Όχι	Όχι	Μέγιστη	C/C++ , Fortran, Pascal, Go
JIT/AOT Compiler (Just-in-time/Ahead-of-time compiler) μίξη των παραπάνω, Compilation σε Bytecode	Ναι	Ναι	Ναι	Μεγάλη	Java, Kotlin, C#, VisualBasic

Παράδειγμα Scripting

Μία πολύ διαδεδομένη **scripting** γλώσσα είναι η PHP. Ακολουθεί ένα πρόγραμμα που εμφανίζει έναν χαιρετισμό στην οθόνη:

Edit

```
<?php

echo 'Hello world!';
sleep(10);
```

Το οποίο εκτελείται (εφόσον είναι εγκατεστημένη η PHP στον υπολογιστή) γράφοντας στη γραμμή εντολών:

Run

```
> php file-name.php
```

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-23@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε μαζί τον κώδικα και τα μηνύματα λάθους από το CLion ως κείμενα με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Τονίζουμε : Μην στείλετε **ποτέ κώδικα ως εικόνα**, είναι παντελώς άχρηστος!

Η γλώσσα C

Πρώτα στοιχεία της και περιβάλλον ανάπτυξης (IDE)

Το 1ο πρόγραμμα σε C

```
#include <stdio.h>


int main() {
    printf("Hello world!\n");
    return 0;
}
```

Δίπλα παρουσιάζεται το αντίστοιχο πρόγραμμα με αυτό της PHP, γραμμένο στη γλώσσα C.

Είναι και πάλι ένα απλό αρχείου κειμένου. Για να εκτελεστεί πρέπει να μετατραπεί σε εκτελέσιμο αρχείο.

Τη διαδικασία αυτή την αποκαλούμε συνήθως compilation, όμως αυτό είναι ανακριβές. Το compilation είναι ένα (το κυριότερο) βήμα της. Όλη η διαδικασία ονομάζεται building και περιλαμβάνει μεταξύ άλλων και το βήμα του linking. Το linking συνδέει τον compiled κώδικα που έχει γραφεί, με όλο τον βοηθητικό κώδικα που απαιτείται για την εκτέλεση του προγράμματος.

IDE : Το περιβάλλον ανάπτυξης

Στη C, αλλά και στις περισσότερες υψηλού επιπέδου γλώσσες (χωρίς να είναι αυτό απαραίτητο), όλη η διαδικασία της ανάπτυξης ενός προγράμματος, πραγματοποιείται μέσα σε ένα λογισμικό το οποίο ονομάζεται Integrated Development Environment (δηλαδή Ολοκληρωμένο Περιβάλλον Ανάπτυξης).

Αυτό περιλαμβάνει τουλάχιστον:

1. τον **editor** (τον επεξεργαστή κειμένου) όπου γράφεται ο κώδικας
2. τον **builder** όπου γίνεται η μετατροπή του κώδικα σε εκτελέσιμο
3. τη γραμμή εντολών όπου γίνεται η εκτέλεση του κώδικα
4. τον **debugger**, όπου βοηθά τον εντοπισμό και την επίλυση σφαλμάτων (bugs) στον κώδικα

Διαθέσιμα IDEs

Για τη C υπάρχουν πολλά διαθέσιμα IDEs. Μερικά από αυτά είναι:

- JetBrains CLion
- Bloodshed C++
- Netbeans
- Codeblocks
- Eclipse
- Microsoft Visual Studio
- VSCode

Από αυτά, στο μάθημα θα χρησιμοποιούμε το **CLion**, το οποίο είναι ένα επαγγελματικό IDE.

Υπάρχει ήδη εγκατεστημένο στο PC-Lab της Σχολής, αλλά μπορείτε (και πρέπει) να το εγκαταστήσετε σε έναν υπολογιστή σας ώστε να μπορείτε να κάνετε τις εργασίες σας και από το σπίτι.

CLion : Εγκατάσταση και 1η χρήση

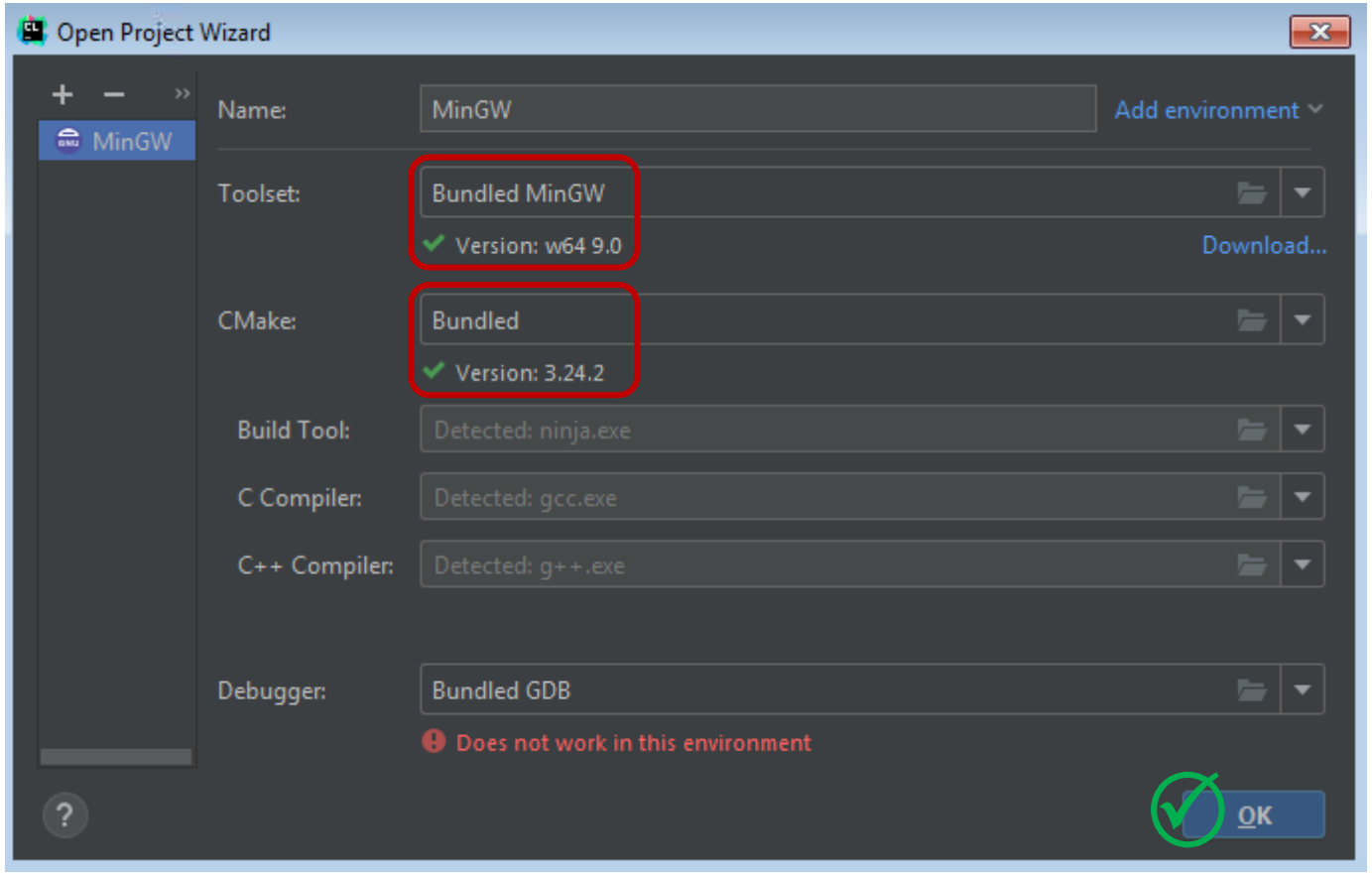
Η εγκατάσταση του CLion γίνεται με τα ακόλουθα βήματα:

1. Δημιουργία κωδικού χρήστη στην JetBrains, χρησιμοποιώντας το e-mail (mc######@mail.ntua.gr) του ΕΜΠ
2. Λήψη και εγκατάσταση
3. Ρύθμιση κατά την πρώτη χρήση (δεν απαιτούνται κάποια plugins)
4. Ενεργοποίηση του στον υπολογιστή σας (επιτρέπεται μόνο ένας Η/Υ ανά φοιτητή)
5. Δεν απαιτείται ρύθμιση του toolchain, μόνο επιβεβαίωση

CLion : Επιβεβαίωση toolchain

Το μόνο που απαιτείται είναι κατά τη δημιουργία του 1ου project στο παράθυρο που φαίνεται και εδώ, να βεβαιωθείτε ότι τα Toolset και Cmake είναι αυτά που γράφουν Bundled και ότι έχουν το πράσινο check από κάτω τους. Ο Debugger δεν πειράζει εάν φαίνεται κόκκινος.

Τέλος πατάτε το OK κάτω δεξιά.



CLion : Οι βασικές περιοχές του IDE

Κεντρικό Menu

Περιλαμβάνει όλες τις επιλογές του IDE. Είναι όλο και πιο χρήσιμο καθώς προοδεύει ο προγραμματιστής

Περιοχή Project

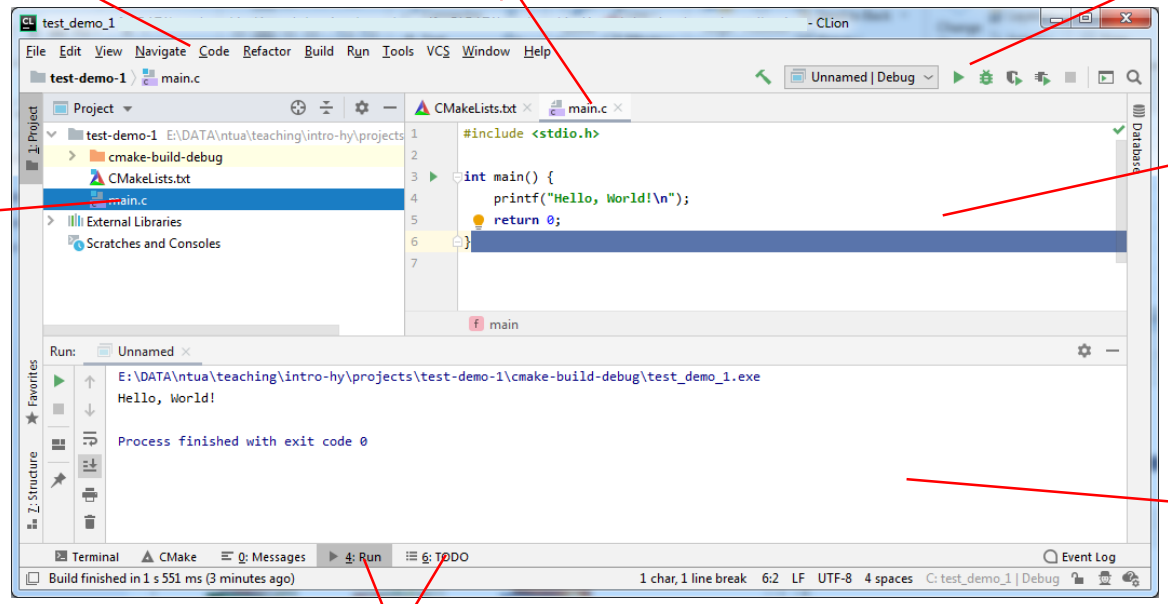
Όλα τα σχετικά και απαραίτητα αρχεία Περιλαμβάνουν και το αρχείο του κώδικα (εδώ main.c)

Καρτέλες Επιλογής

Από αυτές τις καρτέλες επιλέγεται τι εμφανίζει κάθε στιγμή η περιοχή αποτελεσμάτων. Τα δύο βέλη υποδεικνύουν τις δύο πιο συχνά χρησιμοποιούμενες καρτέλες. Η αριστερή εμφανίζει τα μηνύματα κατά τη διάρκεια του build ενώ η δεξιά εμφανίζει τα αποτελέσματα της εκτέλεσης.

Καρτέλες Επεξεργαστή

Εμφανίζουν τα αρχεία που είναι ανοιχτά στον επεξεργαστή. Το τρέχον αρχείο ξεχωρίζει.



Γραμμή εκτέλεσης

Περιέχει επιλογή του χτισίματος, της εκτέλεσης, του debugging, κ.α.

Περιοχή Επεξεργαστή

Εμφανίζεται ο κώδικας του προγράμματος. Έχει διάφορα βοηθητικά χαρακτηριστικά όπως είναι ο χρωματισμός των εντολών (syntax highlighting) και άλλα.

Περιοχή Μηνυμάτων & Αποτελεσμάτων

Εμφανίζει τα μηνύματα κατά το Building ή τα μηνύματα κατά την εκτέλεση κ.α.

CLion : Βασικά στοιχεία του editor

Ο editor (επεξεργαστής) του κώδικα είναι ένας κειμενογράφος απλών κειμένων (όπως το Σημειωματάριο/Notepad των Windows) με αρκετές πρόσθετες δυνατότητες. Για παράδειγμα:

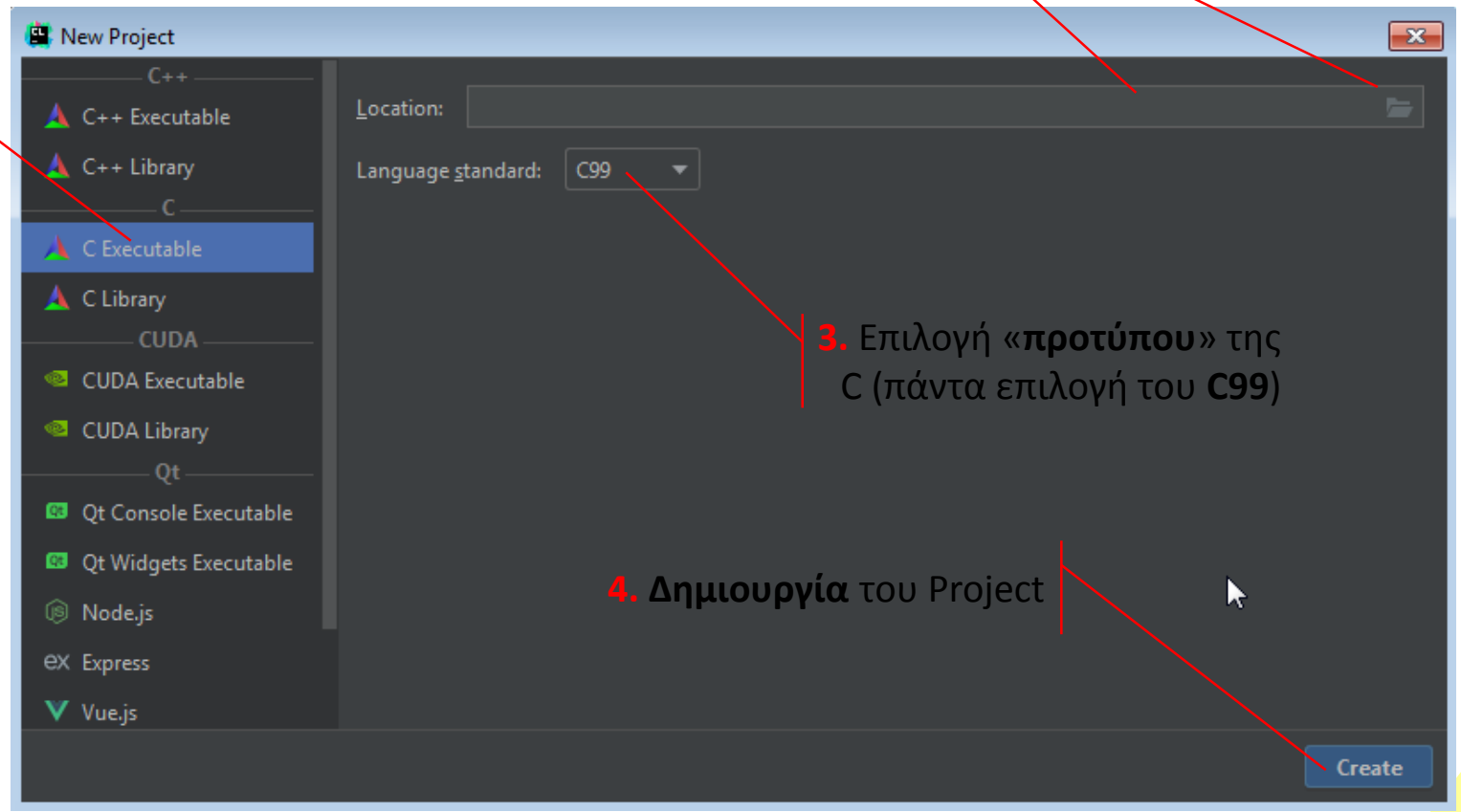
- Syntax highlighting : Χρωματισμός των διαφόρων σημείων του κώδικα ώστε να βοηθά στον εντοπισμό λέξεων κλειδιών, σφαλμάτων, κλπ
- IntelliSense : Αυτόματη συμπλήρωση λέξεων κλειδιών, ονομάτων (identifiers), παραμέτρων συναρτήσεων, κατά την πληκτρολόγηση
- Μετονομασία μεταβλητών, όπου αυτόματα μετονομάζονται όλες οι εμφανίσεις τους
- Προειδοποιήσεις για πιθανά σφάλματα
- Προτάσεις για βελτίωση του κώδικα
- Αυτόματη μορφοποίηση του κώδικα

CLion : Δημιουργία ενός Project

Επιλέγοντας από το μενού του CLion, ***File > New Project*** εμφανίζεται το διπλανό πλαίσιο διαλόγου (**dialog**).

Το κάθε project αποθηκεύεται σε ένα φάκελο στον δίσκο. Εκεί βρίσκονται όλα τα απαραίτητα αρχεία για αυτό το project.

Στο κάθε project αντιστοιχεί τουλάχιστον ένας «στόχος» (**target**) που είναι το αποτέλεσμα του build. Συνήθως αυτός ο στόχος είναι ένα εκτελέσιμο αρχείο.



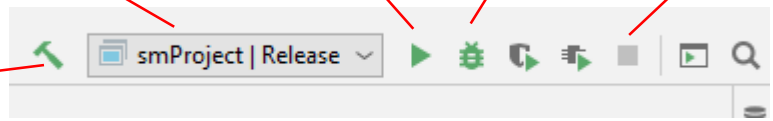
CLion : Χτίσιμο και Εκτέλεση

Για την **εκτέλεση** του προγράμματος πατώ εδώ

Για την εκτέλεση με **debugging** του προγράμματος πατώ εδώ

Όταν υπάρχουν περισσότεροι από έναν εκτελέσιμοι στόχοι επιλέγω από εδώ

Για τον **τερματισμό** ενός ήδη εκτελούμενου προγράμματος πατώ εδώ



Για το **χτίσιμο (build)** του προγράμματος πατώ εδώ

Όταν επιλέγει ο χρήστης **εκτέλεση** (**run**) τότε, εφόσον έχει αλλάξει ο κώδικας από το προηγούμενο build, αυτόματα γίνεται πρώτα **build** το project και κατόπιν ξεκινά η εκτέλεση.
Τα μηνύματα του **build** εμφανίζονται στο κάτω μέρος του IDE στην καρτέλα **Messages**.
Ενώ τα μηνύματα της **εκτέλεσης** του προγράμματος εμφανίζονται στην καρτέλα **Run**.

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-23@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε μαζί τον κώδικα και τα μηνύματα λάθους από το CLion ως κείμενα με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Τονίζουμε : Μην στείλετε **ποτέ κώδικα ως εικόνα**, είναι παντελώς άχρηστος!

Το πρώτο μου πρόγραμμα

Παρουσίαση και επεξήγηση του πρώτου προγράμματος σε C

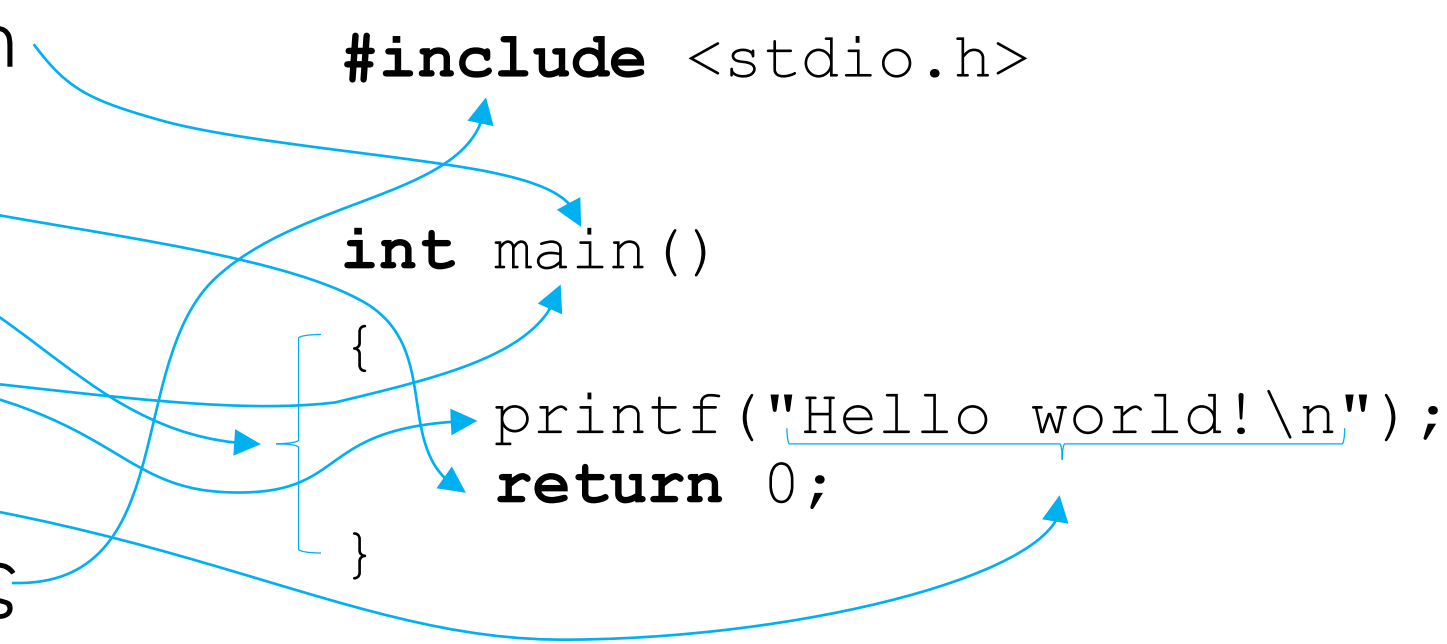
Γνωριμία με τη γλώσσα

Τι είναι ένα πρόγραμμα

- Ένα πρόγραμμα υπολογιστή είναι μια **ακολουθία** εντολών που εκτελούνται:
  - μία κάθε φορά
  - η μία μετά την άλλη με τη σειρά που εμφανίζονται, τοποθετημένες ώστε να επιτελούν ένα συγκεκριμένο επιθυμητό σκοπό.
- Οι εντολές μπορούν να **ομαδοποιούνται** μεταξύ τους ώστε να επιτελούν επιμέρους λειτουργίες
- Οι εντολές και οι λειτουργίες συνήθως είναι **παραμετρικές**
- Το αντικείμενο των εντολών συχνά είναι ο **μετασχηματισμός** της πληροφορίας
- Πολύ συχνά χρησιμοποιούμε **έτοιμες** λειτουργίες με σκοπό την **ταχύτερη δημιουργία** του προγράμματος

Βασικά στοιχεία του 1ου προγράμματος σε C

1. Η συνάρτηση main
2. Εντολές
3. Block εντολών
4. Συνάρτηση
5. Κείμενα
6. Χρήση βιβλιοθήκης



Περισσότερα στοιχεία του 1ου προγράμματος σε C

- Εκτός από την πυρήνα της γλώσσας όλες οι υπόλοιπες συναρτήσεις είναι από **εξωτερικές βιβλιοθήκες** και με την `#include` δηλώνουμε ότι θέλουμε να τις χρησιμοποιήσουμε (περισσότερα αργότερα).
- Η C έχει **ελεύθερη σύνταξη**, δηλαδή ανάμεσα στα σύμβολα και τις λέξεις μπορούν να υπάρχουν κενοί χαρακτήρες (space, tab, enter) οπουδήποτε και οσοιδήποτε. Αυτό επιτρέπει την καλή οπτική τοποθέτηση των εντολών και τον παραμέτρων τους. Ο μόνος περιορισμός είναι ότι εντός των εισαγωγικών (δηλαδή στα κείμενα) δεν ισχύει η ελεύθερη σύνταξη και δεν επιτρέπεται η αλλαγή γραμμής!

Τα στοιχεία του κώδικα

Αναγνωριστικά / Τελεστές / Λέξεις κλειδιά / Σταθερές / Προεπεξεργαστής

Είναι λεκτικά δικής μας επιλογής που χρησιμοποιούμε για να ονομάσουμε συνήθως μεταβλητές και συναρτήσεις.

Είναι σύμβολα, τα οποία περιγράφουν μία πράξη μεταξύ «ποσοτήτων» ή οριοθετούν μια περιοχή του κώδικα.

Είναι ειδικές λέξεις που έχουν συγκεκριμένο ρόλο στη C και φυσικά δεν μπορούν να χρησιμοποιηθούν ως αναγνωριστικά.

```
#include <stdio.h>

int main()
{
    printf("Hello world!\n");
    return 0;
}
```

Παριστάνουν κάποια συγκεκριμένη πληροφορία με σταθερή τιμή.

() vs [] vs {}

ΠΡΟΣΟΧΗ!

Επειδή στα μαθηματικά χρησιμοποιούνται ελεύθερα μεταξύ τους τα παραπάνω σύμβολα, δεν συμβαίνει το ίδιο στον προγραμματισμό! Έχουν τελείως διαφορετικές χρήσεις:

- () οι **παρενθέσεις** ή brackets χρησιμοποιούνται στις παραστάσεις για αλλαγή προτεραιότητας αλλά και στις συναρτήσεις (βλ. `main`)
- [] οι **αγκύλες** ή square brackets χρησιμοποιούνται στους πίνακες
- {} τα **άγκιστρα** ή curly brackets χρησιμοποιούνται για να ορίζουν ομάδες (block) εντολών

ΠΡΟΣΟΧΗ και στις **ονομασίες** τους, ώστε να υπάρχει σωστή κατανόηση!!

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

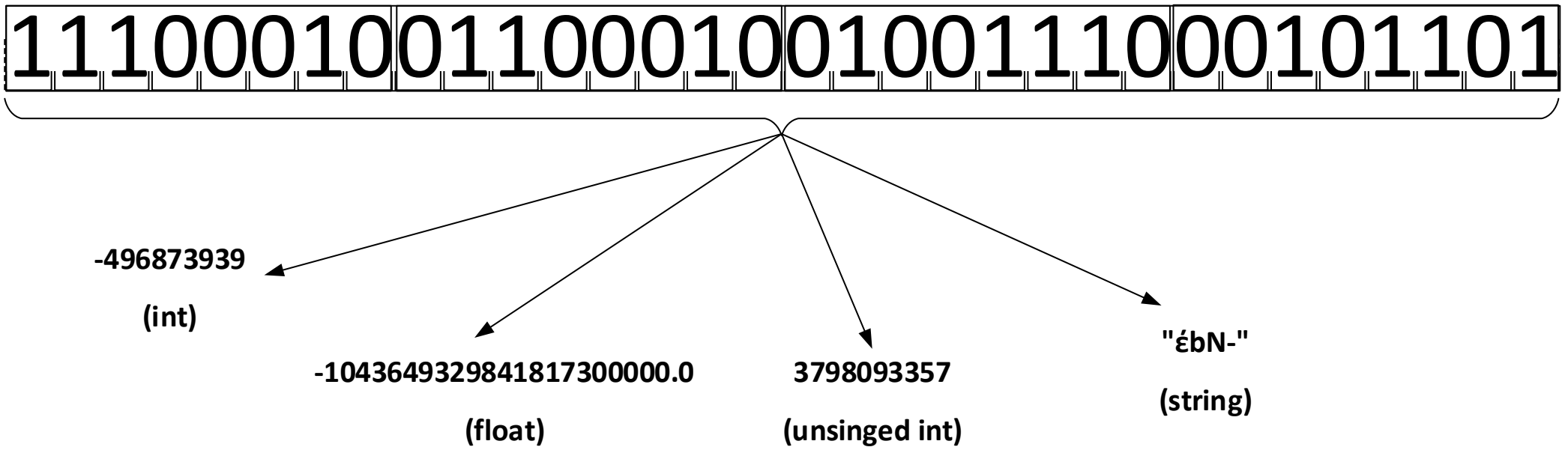
  c-programming-23@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε μαζί τον κώδικα και τα μηνύματα λάθους από το CLion ως κείμενα με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Τονίζουμε : Μην στείλετε **ποτέ κώδικα ως εικόνα**, είναι παντελώς άχρηστος!

Δεδομένα, Παραστάσεις & Μεταβλητές

Πως παριστάνονται στην C

Μετάφραση της δυαδικής πληροφορίας



Η δυαδική πληροφορία που βρίσκεται στη μνήμη του Η/Υ από μόνη της «δεν λέει τίποτα». Χρειάζεται να «γνωρίζει» κάπως ο Η/Υ (ο επεξεργαστής) το πώς πρέπει να την μεταφράσει. Αυτό το καθορίζει ο **τύπος των δεδομένων**.

Εγγενείς (Native) τύποι δεδομένων

Ο τύπος των δεδομένων καθορίζει την κωδικοποίηση μιας πληροφορίας σε δυαδική μορφή. Η κωδικοποίηση των βασικών τύπων δεδομένων που αφορούν αριθμούς (ακέραιους και πραγματικούς) καθορίζεται από τον ίδιο τον επεξεργαστή. Αυτό συμβαίνει επειδή η γλώσσα μηχανής (και κατ'αντιστοιχία τα εσωτερικά κυκλώματα του επεξεργαστή) λειτουργούν με συγκεκριμένο τρόπο/συνδεσμολογία.

Οι τύποι αυτοί που υποστηρίζονται από τον επεξεργαστή ονομάζονται και εγγενείς (native).

Εγγενείς τύποι δεδομένων και όρια

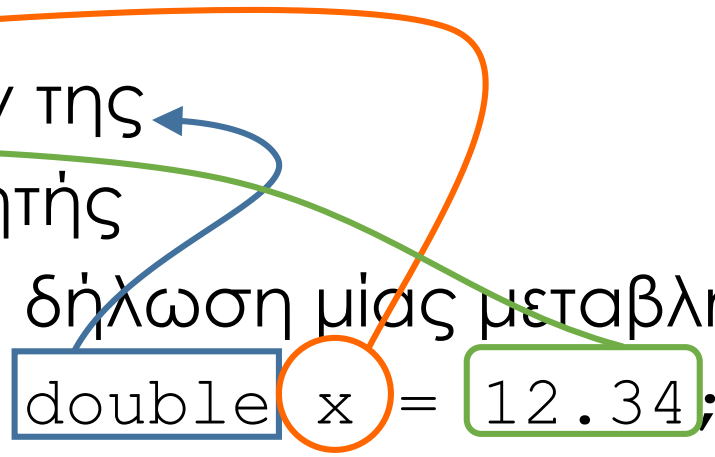
Τύπος Δεδομένων	Bits	Ελάχιστη	Μέγιστη
Ακέραιες **Απρόσημες** Τιμές			
`unsigned char`	8	0	255
`unsigned short`	16	0	65535
`unsigned int`	32	0	4294967295
`unsigned long`	32	0	4294967295
`unsigned long long`	64	0	18446744073709551615
Ακέραιες **Προσημασμένες** Τιμές			
`char`	8	-128	127
`short`	16	-32768	32767
`int`	32	-2147483648	2147483647
`long`	32	-2147483648	2147483647
`long long`	64	-9223372036854775808	9223372036854775807
Δεκαδικές Τιμές (κινητής υποδιαστολής, πάντα προσημασμένες)			
`float`	32	±1.17549e-038	± 3.40282e+038
`double`	64	±2.22507e-308	± 1.79769e+308

Μεταβλητές στη C

Όταν είναι επιθυμητό να αποθηκευθεί στη μνήμη του Η/Υ το αποτέλεσμα μιας πράξης, ένας αριθμός ή κάποια άλλη πληροφορία χρησιμοποιούνται οι μεταβλητές. Αυτές μοιάζουν με τις γνωστές μεταβλητές των μαθηματικών, όμως στους Η/Υ η μεταβλητή συνδέεται με 3 πράγματα:

- Την ονομασία της
- Τον τύπο δεδομένων της
- Την τιμή της μεταβλητής

Αυτό φαίνεται και στη δήλωση μίας μεταβλητής. Π.χ.:



Τιμή της μεταβλητής σε συνδυασμό με τον τύπο δεδομένων της καθορίζει το δυαδικό περιεχόμενο της μνήμης.

Ονομασία & Αναγνωριστικά (Identifiers)

Με τον όρο Αναγνωριστικά (Identifiers) αποκαλούμε τις ονομασίες στη C, τις οποίες επιλέγει ο προγραμματιστής για να ονοματίσει τις μεταβλητές, τις συναρτήσεις, κ.α. στον κώδικα που αναπτύσσει.

Οι περιορισμοί για τα ονόματα αυτά είναι οι παρακάτω:

- Αποτελούνται από **λατινικούς** χαρακτήρες, **αριθμητικούς** χαρακτήρες και την **κάτω παύλα _** (underscore)
- Ο **πρώτος** χαρακτήρας δεν μπορεί να είναι αριθμητικός
- Οι πεζοί χαρακτήρες με τους αντίστοιχους κεφαλαίους θεωρούνται διαφορετικοί (είναι **case sensitive**)
- Δεν επιτρέπεται να συμπίπτουν με **δεσμευμένες λέξεις** της C
- Ενδεχομένως (ανάλογα τον compiler) να υπάρχει ένα όριο στο **πλήθος των χαρακτήρων** επιτρέπεται να είναι ένα αναγνωριστικό

Έτσι το `someName1` , `some_name_1` και το `Somename1` είναι αποδεκτά αναγνωριστικά και διαφορετικά μεταξύ τους, ενώ το `1_name` δεν είναι αποδεκτό επειδή ξεκινά με αριθμητικό χαρακτήρα.

Δεσμευμένες λέξεις στη C

Στη C κάποιες λέξεις χρησιμοποιούνται από την ίδια τη γλώσσα. Αυτές ονομάζονται δεσμευμένες λέξεις. Αυτές είναι:

`auto`	**`register`**	`static`	**`extern`**	`const`	**`volatile`**
`signed`	**`unsigned`**	`short`	**`long`**	`void`	**`char`**
`int`	**`float`**	`double`	**`while`**	`do`	**`for`**
`continue`	**`break`**	`if`	**`else`**	`switch`	**`case`**
`default`	**`goto`**	`return`	**`struct`**	`union`	**`enum`**
`typedef`	**`sizeof`**				

Αυτές, λοιπόν, είναι οι λέξεις που δεν μπορούν να χρησιμοποιηθούν αυτούσιες ως identifiers (αναγνωριστικά).

Δήλωση και χρήση μιας μεταβλητής τοπικής εμβέλειας

Οι μεταβλητές δηλώνονται μέσα σε άγκιστρα, δηλαδή σε **block εντολών**. Από το σημείο της δήλωσή τους και μέχρι το τέλος του block μπορούν να χρησιμοποιηθούν, είτε δίνοντας τιμή σε αυτές, είτε χρησιμοποιώντας τις μεταβλητές μέσα σε παραστάσεις όμοια με τις σταθερές τιμές. Μετά το τέλος του block η μεταβλητή **διαγράφεται** και δεν μπορεί να χρησιμοποιηθεί πλέον.

Πριν την πρώτη χρήση κάθε μεταβλητής θα πρέπει ο κώδικας να της αναθέτει κάποια τιμή. Αυτό συνήθως γίνεται κατά τη δήλωση. Εάν δεν ανατεθεί αρχική τιμή, η μεταβλητή θα έχει τυχαία τιμή και όχι μηδενική. Εκτός της δήλωσης η ανάθεση γίνεται με τη μορφή:

```
myVariable = 0.0;
```

Φυσικά εκτός από συγκεκριμένη τιμή, μπορεί να δοθεί και ολόκληρη παράσταση στα δεξιά του =.

Προσέξτε ότι το = έχει την έννοια της ανάθεσης και **όχι της εξίσωσης**, οπότε το παρακάτω είναι σωστή σύνταξη και αυξάνει την τιμή της μεταβλητής κατά A.

```
myVariable = myVariable + A;
```

Τιμές στη C για τους native τύπους δεδομένων

Μέσα σε ένα πρόγραμμα C οι τιμές των native τύπων δεδομένων μπορούν να γραφούν με έναν από τους ακόλουθους τρόπους.

- Ακέραιοι αριθμοί (προσημασμένοι ή απρόσημοι)
  - **Δεκαδικό** σύστημα αρίθμησης : **[πρόσημο]ψηφία(0-9)**
    π.χ. **+123 1952384 -2435 0**
  - **Οκταδικό** σύστημα αρίθμησης : **[πρόσημο]0ψηφία(0-7)**
    π.χ. **+0123 0153425 -05234 0**
  - **Δεκαεξαδικό** σύστημα αρίθμησης : **[πρόσημο]0xψηφία(0-9 ή A-F ή a-f)**
    π.χ. **+0x53A4F -0x123 +0x0FFF 0x0**

Σημειώστε ότι αυτό που αλλάζει είναι ο τρόπος έκφρασης και όχι οι το δυαδικό περιεχόμενο που αντιστοιχεί στον ίδιο αριθμό. Π.χ. το `0x100` και το `256` που είναι ο ίδιος αριθμός έχουν την ίδια δυαδική αναπαράσταση, ανεξαρτήτως εάν δηλώθηκε ως δεκαδικός ή δεκαεξαδικός αριθμός.

Τιμές στη C για τους native τύπους δεδομένων

- Κινητής υποδιαστολής (πάντα προσημασμένοι)
  - Απλή γραφή :

    [πρόσημο]ψηφία(0-9).ψηφία(0-9)

    υποχρεωτικά υπάρχουν η υποδιαστολή (που είναι τελεία, όχι κόμμα) και ένα ψηφίο

  - Επιστημονική ή εκθετική γραφή :

    [πρόσημο] ψηφία(0-9).ψηφία(0-9)e[πρόσημο]ψηφία(0-9)

    υποχρεωτικά υπάρχουν το **e** (ή **E**), στα αριστερά του η βάση γράφεται σε απλή γραφή και στα δεξιά ο εκθέτης ως προσημασμένος ακέραιος. Σε όλη την έκταση του αριθμού δεν υπάρχουν καθόλου κενά.

Σωστό	Λάθος
1234.5678	1234
-1000.0001	1234,5678
+123.345	12 .34
-.34	11.22.33
123.	.

Σωστό	Λάθος
-12.34e10	12e 3
13.1002e-30	1e8.2
12.32E+4	
.132E+2	
1e3	

ΣΗΜΕΙΩΣΗ : Οι αγκύλες [ ] εδώ συμβολίζουν ότι το πρόσημο είναι προαιρετικό το να μπει (για τους θετικούς αριθμούς). Δεν πληκτρολογούνται και δεν εμφανίζονται πουθενά όπως βλέπετε και στα παραδείγματα στα δεξιά!

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-23@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε μαζί τον κώδικα και τα μηνύματα λάθους από το CLion ως κείμενα με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Τονίζουμε : Μην στείλετε **ποτέ κώδικα ως εικόνα**, είναι παντελώς άχρηστος!

Μορφοποιημένη έξοδος (`printf`)

Για την εμφάνιση τιμών στην οθόνη χρησιμοποιείται η `printf`. Το κείμενο της πρώτης παραμέτρου της `printf` εμφανίζεται στην οθόνη, όχι όμως αυτούσιο. Υπάρχουν δύο περιπτώσεις όπου το κείμενο διαφοροποιείται. Η πρώτη περίπτωση προκύπτει όταν συναντάται ο χαρακτήρας `%` και η δεύτερη όταν συναντάται ο χαρακτήρας `\`.

Έτσι για παράδειγμα όταν η `printf` κληθεί ως εξής:

```
printf("Please wait for %d more seconds\n", 123);
```

Στο σημείο που υπάρχει ο χαρακτήρας `%`, σημαίνει ότι εκεί πρέπει να εμφανιστεί μία τιμή (εδώ η 123). Για κάθε χαρακτήρα `%` που υπάρχει στο κείμενο μία ακόμα τιμή πρέπει να δίνεται ως παράμετρος στην `printf`. Το πώς θα εμφανιστεί η τιμή εξαρτάται από τον χαρακτήρα που ακολουθεί το σύμβολο `%`. Εδώ το `%d` υποδεικνύει ότι η τιμή πρέπει να τυπωθεί η τιμή ως ακέραιος στο δεκαδικό σύστημα αρίθμησης.

Προσέξτε ότι η μορφοποίηση που θα επιλεγεί πρέπει να ταιριάζει με τον τύπο δεδομένων της ποσότητας που δίνεται, αλλιώς το εμφανιζόμενο αποτέλεσμα μπορεί να είναι παραπλανητικό (άρα και λανθασμένο).

Μορφοποιημένη έξοδος (`printf`)

Οι διαθέσιμες και κατάλληλες μορφοποιήσεις υπάρχουν στον παράρτημα III (αλλά και στο παράρτημα II) των σημειώσεων. Αυτές οι ακολουθίες χαρακτήρων αφορούν μόνο την `printf`, αντίθετα οι παρακάτω ακολουθίες αφορούν οποιαδήποτε κείμενα

Στο σημείο που υπάρχει ο χαρακτήρας ** (backslash), σημαίνει ότι ο επόμενος χαρακτήρας αποκτά διαφορετική σημασία. Αυτές οι ακολουθίες χαρακτήρων ονομάζονται **escape sequence και ο χαρακτήρας **** ονομάζεται **escape character**. Εδώ η ειδική ακολουθία χαρακτήρων η `\n` υποδεικνύει μια αλλαγή γραμμής.

Χωρίς την αλλαγή γραμμής η επόμενη `printf` θα εμφάνιζε το αποτέλεσμά της δίπλα από αυτό της όποιας `printf` είχε τυχόν προηγηθεί. Οι περισσότεροι από αυτούς τους χαρακτήρες αναφέρονται στον πίνακα του παραρτήματος I των σημειώσεων.

Σημειώστε ότι αφού οι χαρακτήρες **%** και **** έχουν ειδική χρήση και συνδυάζονται με τους χαρακτήρες που τους ακολουθούν, τότε πως θα μπορούσαμε να εμφανίσουμε ένα μήνυμα που να τους περιέχει; Η λύση είναι ότι απλά τους γράφουμε δύο φορές δίπλα-δίπλα δηλαδή ως **%%** ή ως ****.

Επίσης παρόμοιο πρόβλημα εμφανίζεται και με τον χαρακτήρα " ο οποίος ανοίγει και τερματίζει ένα κείμενο. Για να τον περιλάβουμε σε ένα κείμενο γράφουμε **\"**

Προσοχή! Στη περίπτωση του **%%** εξυπακούεται ότι θα πρέπει να μην δοθεί κάποια αντίστοιχη παράμετρος στην printf!

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-23@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε μαζί τον κώδικα και τα μηνύματα λάθους από το CLion ως κείμενα με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Τονίζουμε : Μην στείλετε **ποτέ κώδικα ως εικόνα**, είναι παντελώς άχρηστος!

Αριθμητικές πράξεις μεταξύ των τιμών

Με δεδομένες τις αριθμητικές ποσότητες το επόμενο βήμα είναι η πραγματοποίηση αριθμητικών πράξεων μεταξύ τους. Στη C είναι διαθέσιμα τα σύμβολα των 4ων πράξεων (**+**, **-**, *****, **/**) αλλά και το σύμβολο **%** το οποίο παριστάνει τον υπολογισμό του υπολοίπου της ακέραιας διαίρεσης. Η προτεραιότητα των πράξεων είναι η γνωστή από τα μαθηματικά. Όπου χρειάζεται να τροποποιηθεί η προτεραιότητα χρησιμοποιούνται αποκλειστικά και μόνο παρενθέσεις. Οι πράξεις πραγματοποιούνται όπως συναντώνται από τα αριστερά προς τα δεξιά.

Πλήρη λίστα των τελεστών με τις προτεραιότητές τους θα βρείτε στις σημειώσεις στην ενότητα «Τελεστές & Παραστάσεις».

Δοκιμάστε

```
5 + 3
12.34 * 1e-2
7.7 / 2
44 % 5
12 / 3 / 4
(1 + 8)/(5 + 4)
5 / 8
4.0 / 2.0 * 2.0
4.4 % 3.3
```

Μετατροπή τύπων δεδομένων

Οι αριθμητικές πράξεις μεταξύ δύο τιμών του ίδιου τύπου δεδομένων δίνουν αποτέλεσμα του ίδιου τύπου. Έτσι η πράξη 5/8 δίνει αποτέλεσμα 0 αφού και οι δύο αριθμοί είναι ακέραιοι, οπότε και το αποτέλεσμα είναι ακέραιο.

Για να γίνουν αριθμητικές πράξεις μεταξύ δύο τιμών διαφορετικών τύπων, μετατρέπονται πρώτα σε τιμές του ίδιου τύπου και μάλιστα προς τον τύπο που καλύπτει μεγαλύτερο εύρος τιμών. Για παράδειγμα πράξη μεταξύ `int` και `double` μετατρέπει πρώτα και τις δύο ποσότητες σε double. Κατόπιν γίνεται η πράξη.

Όταν ο προγραμματιστής θέλει να μετατραπεί μία ποσότητα σε άλλο τύπο δεδομένων, τότε μπορεί να γράψει ακριβώς πριν την ποσότητα τον επιθυμητό τύπο μέσα σε παρενθέσεις. Η διαδικασία αυτή λέγεται type casting ή απλά casting.

Το δεκαδικό αποτέλεσμα στο παραπάνω παράδειγμα προκύπτει με δύο τρόπους:

```
5.0/8          5/(double)8
```

Overflow & Underflow

Επειδή το εύρος των τιμών κάθε τύπου δεδομένων είναι πεπερασμένο, μπορεί το αποτέλεσμα κάποιας πράξης να είναι πολύ μεγάλο σε απόλυτη τιμή για να παρασταθεί από αυτό τον τύπο. Αυτό ονομάζεται υπερχείλιση (overflow). Δείτε και εδώ.

Ειδικά για τις τιμές κινητής υποδιαστολής μπορεί να συμβεί η τιμή του αποτελέσματος κάποιας πράξης να είναι πολύ μικρή σε μέτρο για να παρασταθεί με αυτό τον τύπο δεδομένων. Αυτό ονομάζεται υποχείλιση (underflow). Δείτε και εδώ.

Σημειώστε ότι ως underflow για ακέραιες μεταβλητές ορίζεται η παράσταση τιμής μικρότερης από την ελάχιστη δυνατή.

Ο προγραμματιστής πρέπει να προβλέπει τέτοιες περιπτώσεις καθώς δεν τις ανιχνεύει η C άμεσα.

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-23@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε μαζί τον κώδικα και τα μηνύματα λάθους από το CLion ως κείμενα με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Τονίζουμε : Μην στείλετε **ποτέ κώδικα ως εικόνα**, είναι παντελώς άχρηστος!

Παράδειγμα

Ένας κώδικας που υπολογίζει εμβαδόν του διπλανού σχήματος.

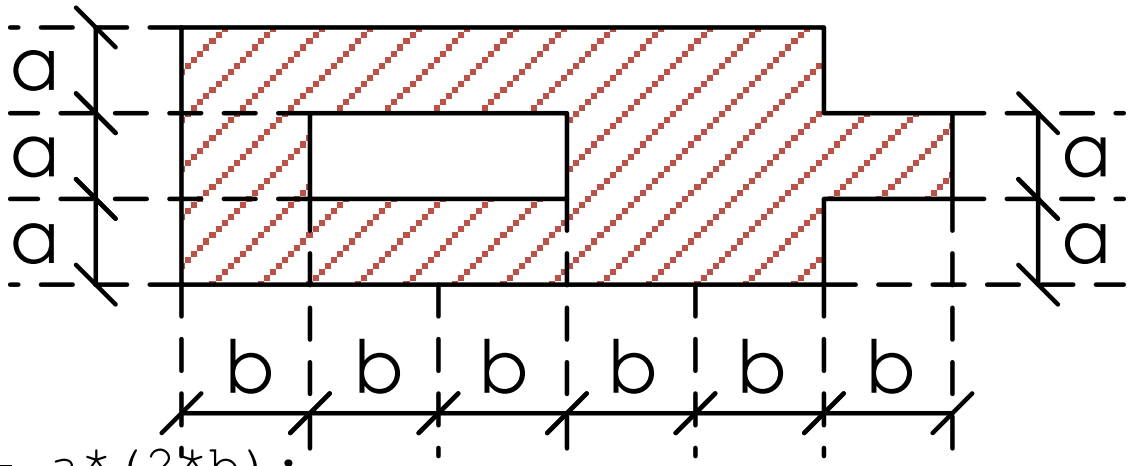
```
#include <stdio.h>
int main() {
    double a = 1.2, b = 2.1;

    double vSide = 3*a;
    double hSide = 5*b;

    double E = vSide * hSide + a*b - a*(2*b);

    printf("To embado eina %lf\n", E);

    return 0;
}
```



Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-23@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε μαζί τον κώδικα και τα μηνύματα λάθους από το CLion ως κείμενα με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Τονίζουμε : Μην στείλετε **ποτέ κώδικα ως εικόνα**, είναι παντελώς άχρηστος!

Σημαντικά σημεία

Μετά από τη σημερινή διάλεξη θα πρέπει να γνωρίζετε:

- Πως να δημιουργήσετε και να εκτελέσετε ένα στοιχειώδες πρόγραμμα στη C
- Πως θα εκτυπώνετε μηνύματα στην οθόνη με την printf
- Ποιοι είναι οι εγγενείς τύποι δεδομένων
- Πως να δηλώνετε μεταβλητές και να τους αναθέτετε τιμές
- Πως να κάνετε αριθμητικές πράξεις με τιμές και μεταβλητές και την προτεραιότητά τους
- Πως να εκτυπώνετε μηνύματα που περιέχουν και τιμές μεταβλητών

Και θα πρέπει να έχετε εγκαταστήσει το CLion στον υπολογιστή σας.

Πρακτικά ζητήματα

smProject – Γιατί σας βοηθά να παραδώσετε σωστότερο αποτέλεσμα

Χρήση smProject

Η κάθε εργασία, εκτός από την εκφώνησή της, θα συνοδεύεται και από ένα CLion project το οποίο έχει σαν σκοπό να σας βοηθά να παραδίδετε πιο σωστές εργασίες, επειδή:

1. Περιλαμβάνει κάποιες δοκιμές για κάποια συνηθισμένα σφάλματα πάνω στα ζητούμενα. Έτσι θα έχετε την ευκαιρία να δείτε μόνοι σας εάν ο κώδικάς σας έχει κάποιο τέτοιο σφάλμα.
   Η λίστα των ελέγχων είναι ενδεικτική και όχι εξαντλητική, έτσι εάν πετύχουν όλοι οι έλεγχοι, πιθανώς να εξακολουθούν να υπάρχουν αρκετά και σοβαρά λογικά σφάλματα στον κώδικά σας. Και τα σταματημένα ρολόγια 2 φορές τη μέρα δείχνουν τη σωστή ώρα.
2. Έχει έτοιμες τις δηλώσεις των συναρτήσεων που ζητούνται, οπότε έχετε μία ευκολία παραπάνω. Σωστό όνομα συνάρτησης, σωστές παραμέτρους και τύπο δεδομένων του αποτελέσματος.
3. Έχει οριοθετημένο τον κώδικα με ένα σχόλιο στην αρχή και ένα στο τέλος. Εφόσον εσείς γράψετε όλο τον κώδικά σας ανάμεσα σε αυτά τα σχόλια, τότε σε συνδυασμό με το αναβαθμισμένο σύστημα υποβολής, εάν δεν έχετε αντιγράψει ολόκληρο τον κώδικα θα εμφανιστεί μία ειδοποίηση γι'αυτό.

Από τα παραπάνω είναι προφανές ότι για κάθε άσκηση θα πρέπει να κατεβάζετε το αντίστοιχο smProject αυτής της άσκησης και σε αυτό το project να γράφετε τον κώδικα.

Για να είναι δυνατή η λειτουργία του smProject, χρειάζεται η παραδοχή ότι αντί της `main`, η κύρια συνάρτηση του προγράμματος θα είναι η `smMain` με την ίδια ακριβώς σύνταξη που έχει η `main`.

Στην επόμενη διαφάνεια θα δείτε πως, όταν ανοίξετε το smProject στο CLion, θα επιλέγετε την εκτέλεση της κανονικής λειτουργίας ή των δοκιμαστικών ελέγχων.

Κανονική εκτέλεση και δοκιμές

Στην γραμμή εκτέλεσης εμφανίζονται πλέον δύο **στόχοι**.

- **smProject** , που αφορά την κανονική εκτέλεση του κώδικα
- **RunTests** , που αφορά την εκτέλεση των δοκιμών

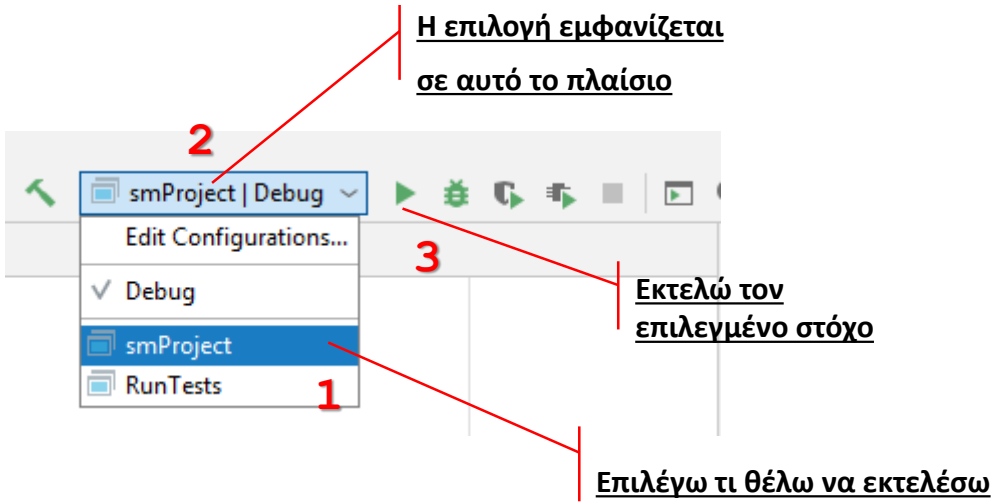
Θα πρέπει λοιπόν ο χρήστης, ανάλογα με την επιθυμία του, να επιλέξει τον αντίστοιχο στόχο.

Αφού έχει γίνει η επιλογή αυτή τα υπόλοιπα κουμπιά της γραμμής λειτουργούν κατά τα γνωστά.

Ως συνήθης πρακτική προτείνεται να γίνονται τα πάντα σε κανονική εκτέλεση μέχρι να θεωρήσει ο προγραμματιστής ότι ο κώδικάς είναι έτοιμος.

Κατόπιν να γίνεται αλλαγή του στόχου σε RunTests για να βεβαιωθεί ότι όλα εκτελούνται καλά.

Ενδεικτικό αποτέλεσμα ελέγχων φαίνεται δίπλα. *Το **ζητούμενο** είναι να μην υπάρχει η λέξη FAILED, να εμφανίζεται μόνο το Ok και η επιστρεφόμενη τιμή να είναι 0.*




```
TESTING MODE!


Test ARC_NORMAL_VALUE_TESTS :
  VALUE_1_2 FAILED : arc(1.2) returns unexpected result!
ARC_NORMAL_VALUE_TESTS FAILED!
Test aktina_0 : Ok
Test aktina_PI : Ok
Process finished with exit code 1
```

Ερωτήσεις?

- Διαβάστε τις σημειώσεις, διαβάστε τις διαφάνειες και δείτε τα videos **πριν** ρωτήσετε
- **Συμβουλευτείτε** τη σελίδα ερωταποκρίσεων του μαθήματος

  https://qna.c-programming.allos.gr
- **Στείλτε** τις ερωτήσεις σας πριν και μετά το μάθημα στο

  c-programming-23@allos.gr

- Εάν έχετε **πρόβλημα** με κάποιο κώδικα στείλτε μαζί τον κώδικα και τα μηνύματα λάθους από το CLion ως κείμενα με copy/paste. Εάν θεωρείτε ότι επιπλέον βοηθά και ένα στιγμιότυπο οθόνης, είναι καλοδεχούμενο.
- Τονίζουμε : Μην στείλετε **ποτέ κώδικα ως εικόνα**, είναι παντελώς άχρηστος!