ΕΠΛ232 – Προγραμματιστικές Τεχνικές και Εργαλεία

Εισαγωγή: C για Προγραμματιστές JAVA
(Κεφάλαια 1-2, KNK-2ED)

Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου

http://www.cs.ucy.ac.cy/courses/EPL232

Dr. Andreas Aristidou a.aristidou@ieee.org

Στοιχεία Διδάσκοντα

- **Διδάσκων:** Ανδρέας Αριστείδου
- **Επικοινωνία (email):** a.aristidou@ieee.org
- **Ώρες γραφείου:** Μόνο με ραντεβού
- **Γραφείο:** ΘΕΕ01 Β113

- **Βοηθοί Διδάσκοντες:** Παύλος Αντωνίου (~csp5pa1) και
  Πύρρος Μπράσκας (~bratskas)

Στόχοι Μαθήματος

- **ΕΠΛ131:** Αρχές Προγραμματισμού Ι
  - Ανάπτυξη δεξιοτήτων στην **επίλυση προβλημάτων** με αλγοριθμικό τρόπο, μέσω **διαδικαστικού** και **αντικειμενοστρεφούς προγραμματισμού**
  - Θεμελίωση της **αλγοριθμικής σκέψης** και των **βασικών αρχών προγραμματισμού**

- **ΕΠΛ232:** Προγραμματιστικές Τεχνικές και Εργαλεία
  - Ενδιάμεσες και **προχωρημένες έννοιες και τεχνικές προγραμματισμού** μέσω μιας χαμηλού επιπέδου γλώσσας
  - Ανάπτυξη μεγάλων **εύρωστων προγραμμάτων / βιβλιοθηκών** τα οποία θα επιλύνουν **πολύπλοκα προβλήματα.**
  - Προχωρημένα θέματα **διαχείρισης της κύριας** και **δευτερεύουσας** μνήμης από τη γλώσσα προγραμματισμού, θέματα **μεταγλώττισης**, **ολοκληρωμένα εργαλεία ανάπτυξης**, μεθόδους **αποσφαλμάτωσης** και **βελτιστοποίησης** του κώδικα

Dr. Andreas Aristidou a.aristidou@ieee.org

Συμβόλαιο Μαθήματος

- **Επίπεδο:** Προπτυχιακό
  - Υποχρεωτικό Μάθημα
- **Πίστωση:** 7.5 μονάδες ECTS
- **Προαπαιτούμενα:**
  - ΕΠΛ131: Αρχές Προγραμματισμού Ι
- **Μέθοδοι Διδασκαλίας**
  - **Διαλέξεις & Φροντιστήρια** (4 ώρες εβδομαδιαίως): Συνδυασμένη Παράδοση Διδακτέας Ύλης και Θεωρητική Εμπέδωση
  - **Εργαστήριο** (2 ώρες εβδομαδιαίως): **Εμπέδωση διαλέξεων μέσω ασκήσεων,** εκμάθηση εργαλείων, πρακτική εξάσκηση, ευκαιρία για πιο προσωπική επίλυση απορίων.

Dr. Andreas Aristidou a.aristidou@ieee.org

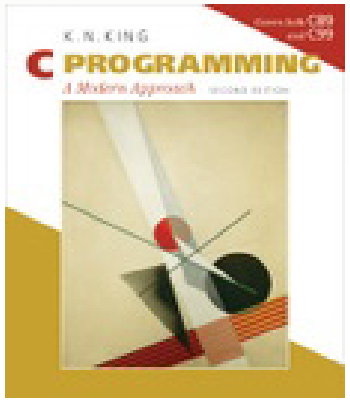
Συμβόλαιο Μαθήματος

- **Αξιολόγηση**
  - **55% Τελική Εξέταση (1)**
  - **20% Ενδιάμεση Εξέταση (1)**
    - Ημερομηνία: **25/10/2024**
  - **25% Ασκήσεις**
    - Προγραμματιστικές Ασκήσεις (2) – 7.5%
    - Ομαδική Εργασία (1) – 10%

Dr. Andreas Aristidou a.aristidou@ieee.org

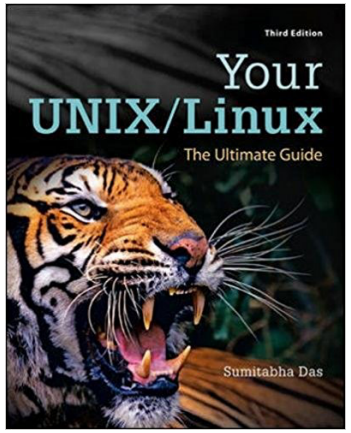
Βιβλιογραφία

Βασική Βιβλιογραφία



- K.N. King, *C Programming: A Modern Approach*, Second Edition, ISBN-10: 0393979504, ISBN-13: 978-0393979503, 832 , W. W. Norton & Company, 2008.
- Your UNIX/Linux: The Ultimate Guide, 3rd Edition, Sumitabha Das, McGraw Hill, ISBN-13 9780073376202, 800 pages, 2013.

Βοηθητική Βιβλιογραφία



- Σημειώσεις Διαλέξεων Μαθήματος
- Programming in C, 4th Edition, Stephen G. Kochan, ISBN-10: 0321776410, ISBN-13: 9780321776419, Addison-Wesley Professional, 600 pp, 2015.
- Η Γλώσσα C σε Βάθος, Νίκος Χατζηγιαννάκης, Τρίτη Έκδοση, 978-960-461-208-6, Κλειδάριθμος, 2009.

Ιστ

- Όλ
  UP

Dr. Andreas Aristidou
University of Cyprus

EPL232: Programming Techniques and Tools

Instructor: Andreas Aristidou
Type: Undergraduate (Compulsory)
Prerequisite: EPL131 - Programming Principles I
Lectures A: Tuesday & Friday, 10:30-12:00 (ΧΩΔ02 #B205)
Recitations A: Wednesday, 11:00-12:00 (ΧΩΔ02 #B210)
Teaching Assistants: Pavlos Antoniou and Pyrros Bratskas

This lecture has been structured based on the notes of the Associate Professor Demetris Zeinalipour.

Overview

The course teaches intermediate and advanced programming concepts, techniques and tools through a language that compiles to machine code. The course familiarizes the students with advanced programming constructs utilized for handling memory and files. Advanced topics in compilation, debugging, documentation and optimization of software. Methodological aspects in developing large-scale system software that addresses complex problems. Basic commands for programmers in the UNIX operating system.

News

Sign-up now to Moodle using code handed out in class!

Content

- Introduction to C for Programmers: types x86/x64, loops, selections, expressions, arrays, functions, IO, basic program organization,
- Advanced C programming constructs: program anatomy and processes, memory and addresses (pointers, pointers and arrays, strings and examples), structures, unions and enumerations. Linear and non-linear programming data structures (dynamic memory allocation, lists, queues, doubly-linked lists, trees, applications and examples).
- Advanced Compilation Topics and Tools: preprocessor directives, compiling multiple files with makefiles, static (.a) and dynamic (.so) linking of object files (.o), error handling (assert.h), static and dynamic code analysis (valgrind and gprof).
- low-level programming (binary operators and examples, binary files and hexdump).
- Basic commands for programmers in the UNIX operating system: file system, redirection and pipes, permissions and basic filters.

» You can download the syllabus of the course here...

Course Schedule and Lectures

1. **Introduction: Syllabus & Basics** Course Objectives and Syllabus, C Advantages and Drawbacks, Comparing C with JAVA, First Program, Directives, Functions, Statements, Compiling C Programs with GCC, Native vs. Intermed. Compile [C/C++/Objective-C vs. JAVA/Android/C#], Basic Data Types, basic comments, x86/x64 data types basics, reserved identifiers. [PDF | 3.98 MB]
2. **Fundamentals I: Formatted I/O, Expressions, Selection Statements, Loops** The printf() and scanf() functions, Arithmetic Operators, Assignment Operators lvalues), Incr/Decr Operators, Expression Evaluation (precedence and order), Logical Expressions and _BOOL (C99), if..else, dangling-else, switch ... break, Loops (while, do, for, break, continue, goto). [PDF | 0.63 MB]
3. **Fundamentals II: Basic Types, Arrays and Functions** Integer, Float, Character Types on x86 and x64 platforms, Type overflow and explanation, Type Conversion (Implicit and Casting), Type Definitions with typedef, The sizeof operator, 1-d arrays and m-d arrays: subscripting, initialization, sizeof. [PDF | 2.21 MB]

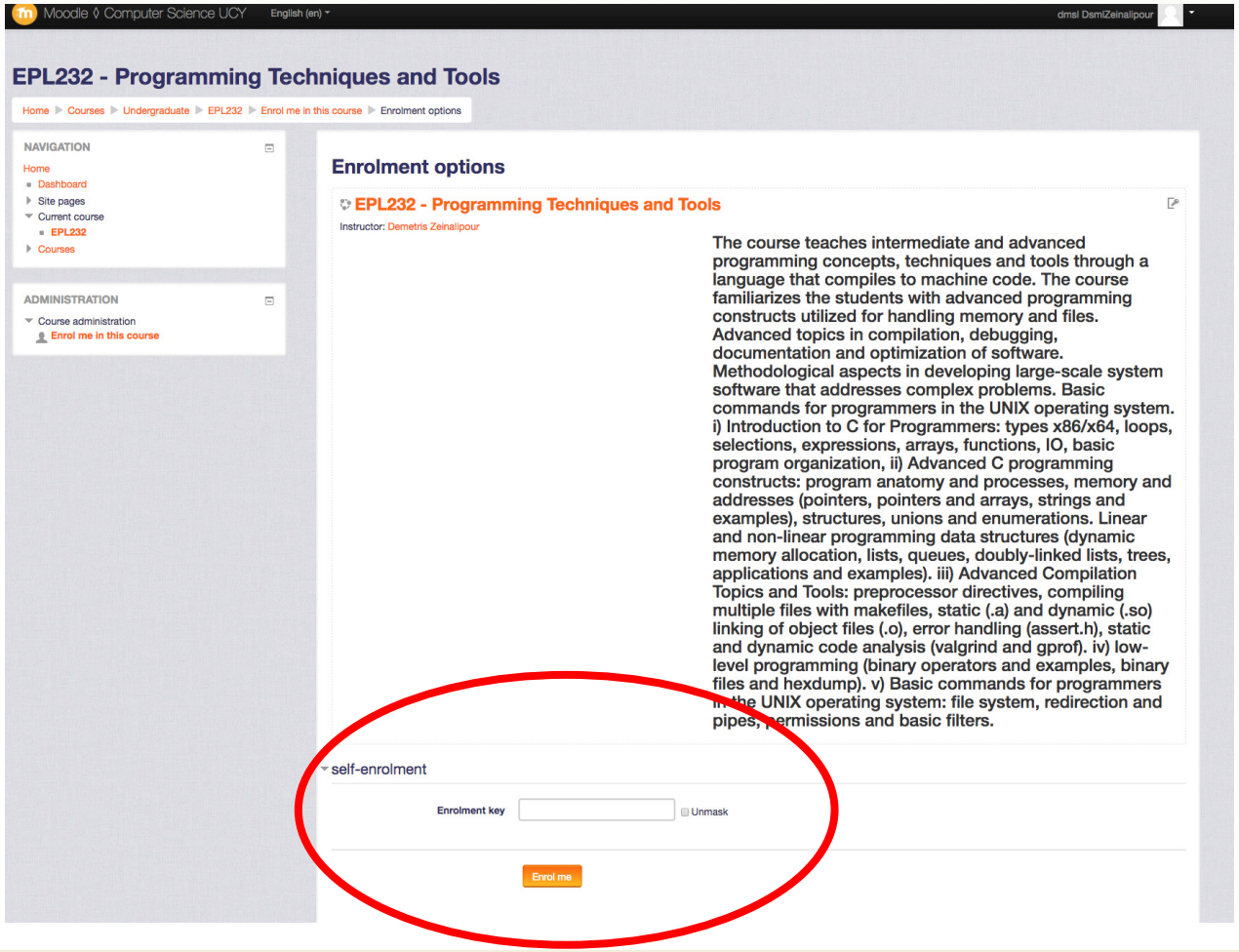
stidou@ieee.org

Ιστοσελίδα μαθήματος

- Όλες οι πληροφορίες του μαθήματος θα βρίσκονται στο ακόλουθο URL: http://www.cs.ucy.ac.cy/courses/EPL232

- Για τις εκπαιδευτικές δραστηριότητες του μαθήματος (**υποβολή εργασιών**, **φόρουμ** ανακοινώσεων, **ερωτηματολόγια**, **βαθμολογίες εργασιών**, κτλ.) θα χρησιμοποιηθεί το Moodle: https://moodle.cs.ucy.ac.cy/

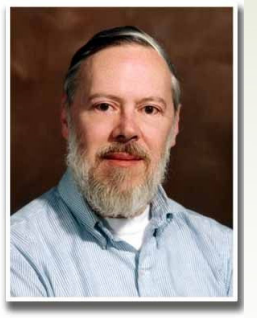
Πλατφόρμα Τηλεκπαίδευσης



Εγγραφείτε κάνοντας χρήση του **Κλειδιού Εγγραφής** που θα δοθεί στην τάξη!

Η γλώσσα προγραμματισμού C


KEN THOMPSON


DENNIS RITCHIE

- Η C είναι ένα υποπροϊόν του UNIX, που αναπτύχθηκε στα εργαστήρια της Bell από τους Ken Thompson, Dennis Ritchie, κ.α.
- Ο Thompson αρχικά ανέπτυξε μια γλώσσα προγραμματισμού την οποία ονόμασε B.
- Το 1971, ο Ritchie άρχισε να αναπτύσσει μια εκτεταμένη έκδοση της B, την οποία αρχικά ονόμασε NB (“New B”).
- Καθώς η γλώσσα άρχισε να αποκλίνει περισσότερο από την B, την μετονόμασε σε C.
- Η γλώσσα ήταν αρκετά σταθερή από το 1973, οπότε πλέον το UNIX μπορούσε να ξαναγραφεί σε C.

Κανονικοποίηση της C

- *K&R C*
  - Περιγράφεται στο Kernighan and Ritchie, *The C Programming Language* (1978)
  - De facto πρότυπο
- *C89/C90*
  - ANSI πρότυπο X3.159-1989 (ολοκληρώθηκε το 1988; εγκρίθηκε και τυπικά από τον Δεκέμβριο του 1989)
  - Διεθνές πρότυποISO/IEC 9899:1990
- *C99*
  - Διεθνές πρότυπο ISO/IEC 9899:1999
  - Ενσωματώνει τις αλλαγές από την τροπολογία 1 (1995)

Γιατί μαθαίνουμε C;

Δημοτικότητα Γλωσσών Προγραμματισμού
02/09/2019

The **TIOBE Programming Community index** is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the number of skilled engineers world-wide, courses and third party vendors. Popular search engines such as Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube and Baidu are used to calculate the ratings.
Note that the TIOBE index is not about the best programming language or the language in which most lines of code have been written.

Aug 2019	Aug 2018	Change	Programming Language	Ratings	Change
1	1		Java	16.028%	-0.85%
2	**2**		**C**	**15.154%**	**+0.19%**
3	4	↑	Python	10.020%	+3.03%
4	3	↓	C++	6.057%	-1.41%
5	6	↑	C#	3.842%	+0.30%
6	5	↓	Visual Basic .NET	3.695%	-1.07%
7	8	↑	JavaScript	2.258%	-0.15%
8	7	↓	PHP	2.075%	-0.85%
9	14	⇈	Objective-C	1.690%	+0.33%
10	9	↓	SQL	1.625%	-0.69%
11	15	⇈	Ruby	1.316%	+0.13%
12	13	↑	MATLAB	1.274%	-0.09%
13	44	⇈	Groovy	1.225%	+1.04%
14	12	↓	Delphi/Object Pascal	1.194%	-0.18%
15	10	⇊	Assembly language	1.114%	-0.30%
16	19	↑	Visual Basic	1.025%	+0.10%
17	17		Go	0.973%	-0.02%
18	11	⇊	Swift	0.890%	-0.49%
19	16	↓	Perl	0.860%	-0.31%
20	18	↓	R	0.822%	-0.14%

Αριθμός Αναζητήσεων (Turing-Complete) Γλωσσών στο WWW

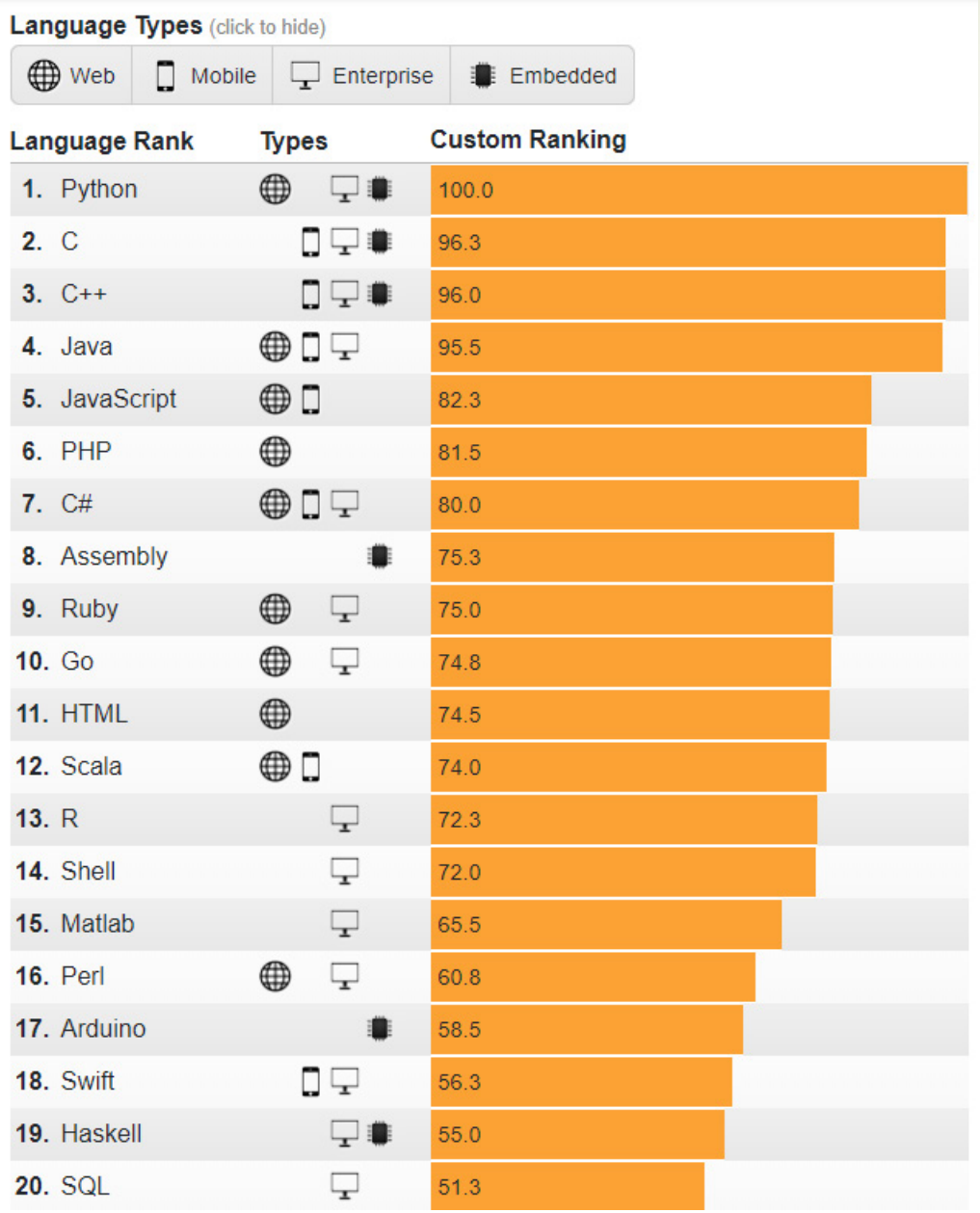
Dr. Andreas Aristidou a.aristidou@ieee.org

Γιατί μαθαίνουμε C;

Δημοτικότητα Γλωσσών Προγραμματισμού
02/09/2019

This app ranks the popularity of dozens of programming languages. This app was originally developed in collaboration with **IEEE Spectrum**.

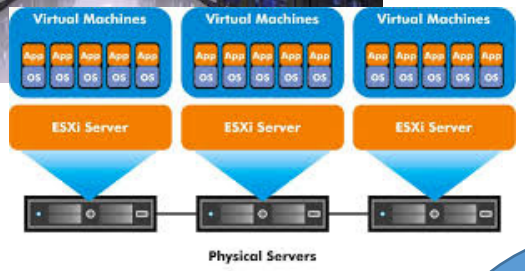
Δείτε τις Top programming Languages 2022



Dr. Andreas Aristidou a.aristidou@ieee.org

Η γλώσσα C και τα UNIX είναι παντού!

Vmware, Oracle, etc.



Cloud

IoT

Linux Kernel (OS)

Cars

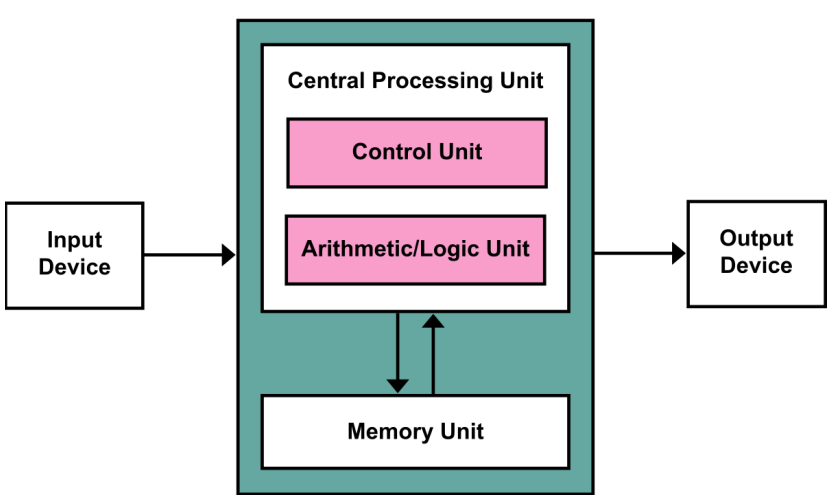
The Tesla GitHub repository contains the code for the Model S/X 2018.12 software release: **Tesla Autopilot platform**, the **kernel sources** for its underlying **hardware**, and the code for its **Nvidia Tegra-based infotainment system**.

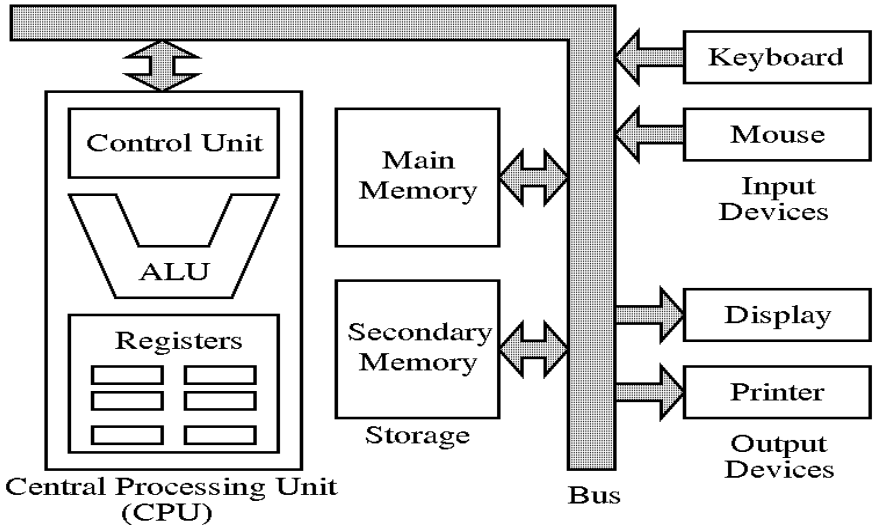
Mobile

Dr. Andreas Aristidou a.aristidou@ieee.org

Γιατί μαθαίνουμε C (μετά την Java);

- **Α) Για να κατανοήσουμε σε βάθος τον κύκλο εκτέλεσης των προγραμμάτων (Διαχείριση Μνήμης, Εισόδου/Εξόδου, Δεδομένα στη Δευτερεύουσα Μνήμη, κτλ).**
  - **Η αρχιτεκτονική Von Neumann** αποτελεί το υπόβαθρο ΟΛΩΝ των σύγχρονων υπολογιστών.





Περιγράψετε τι γίνεται όταν εκτελέσουμε ένα πρόγραμμα επεξεργασίας δεδομένων.

Γιατί μαθαίνουμε C (μετά την Java);

- Γνωρίζοντας το Von Neumann μηχάνημα, θα μπορούμε να εξηγήσουμε σε βάθος την **συμπεριφορά ενός προγράμματος και του συστήματος**
- Ιδιαίτερα, θα εκτιμήσουμε πως αλληλό-συμπληρώνονται τα μαθήματα του προγράμματος σπουδών
  - **Χαμηλού Επίπεδου / Υλικό**
    - ΕΠΛ121 Ψηφιακά Συστ. / ΕΠΛ370 Αρχιτεκ. / ΕΠΛ470 Ενσ. Συστ
    - ΕΠΛ221 Οργάνωση Υπολογιστών και Συμβολικός Προγραμματισμός
  - **Ενδιάμεσου Επιπέδου / Συστήματα**
    - Βάσεις Δεδομένων (ΕΠΛ342 και ΕΠΛ446)
    - Λειτουργικά Συστήματα (ΕΠΛ222),
    - Προγραμματισμός Συστημάτων (ΕΠΛ371)
    - Δίκτυα (ΕΠΛ324 & 375) και Ασφάλεια (ΕΠΛ475)

Γιατί μαθαίνουμε C (μετά την Java);

- **Β) Φιλοσοφικοί Λόγοι:** Η πολυγλωσσία είναι καλή στις μέρες μας.

- Δεν είναι όλες οι γλώσσες αντικειμενοστρεφείς….
  - **Συναρτησιακές Γλώσσες (αποκλειστική χρήση συναρτήσεων):** Haskell, Erlang, «SQL» κτλ.
  - **Λογικές Γλώσσες (αποκλειστική χρήση κανόνων και καταστάσεων):** π.χ., Prolog, Datalog Querying, R++
  - **Διαδικαστικές Γλώσσες (αποκλειστική χρήση διαδικασιών)**: C, Javascript, Fortran, Matlab, Python, Perl, Visual Basic, VB Scripting, Occam, Go, Eiffel, κτλ.
- Πολλές γλώσσες σήμερα παρέχουν διαχείριση αντικειμένων (object-based, π.χ., Obj.name), αλλά όχι αντικειμενοστρέφια.
  - Επομένως είναι καλό να εξασκηθούμε σε ένα διαφορετικό και διαδεδομένο μοντέλο προγραμματισμού.

Γιατί μαθαίνουμε C (μετά την Java);

- **Γ) Άλλοι Λόγοι:**
  - Η C είναι **αποδοτική (efficient)!**
  - Η C είναι η **γλώσσα** του **Unix/Linux,** πλατφόρμες που λειτουργούν το μεγαλύτερο ποσοστό των παγκόσμιων υποδομών & συσκευών στις μέρες μας.
  - Η C είναι **προτυποποιημένη** (standard libraries), **φορητή** (portable), **ευέλικτη** (flexible), **αρθρωτή** (modular), και **επιτρεπτική (permissive).**
    - Κατάλληλη για προχωρημένους προγραμματιστές…
  - Η C είναι η βάση της C++, της Java/C#, της Obj.-C
    - Δες επόμενες διαφάνειες
  - Η C είναι **Χαμηλού Επιπέδου** και **Ψηλού Επιπέδου**
    - από GUI μέχρι Συμβολικό Κώδικα (Assembly)!

Dr. Andreas Aristidou a.aristidou@ieee.org

Αδυναμίες της C

- Τα προγράμματα της μπορεί να είναι επιρρεπή σε λάθη.
- Τα προγράμματα της μπορεί να είναι δύσκολο να κατανοηθούν.
- Τα προγράμματα της μπορεί να είναι δύσκολο να τροποποιηθούν.

Mother Tongues

Tracing the roots of computer languages through the ages

Just like half of the world's spoken tongues, most of the 2,300-plus computer programming languages are either endangered or extinct. As powerhouses C/C++, Visual Basic, Cobol, Java and other modern source codes dominate our systems, hundreds of older languages are running out of life.

An ad hoc collection of engineers-electronic lexicographers, if you will-aim to save, or at least document the lingo of classic software. They're combing the globe's 9 million developers in search of coders still fluent in these nearly forgotten lingua frangas. Among the most endangered are Ada, APL, B (the predecessor of C), Lsp, Oberon, Smalltalk, and Simula.

Code-raker Grady Booch, Rational Software's chief scientist, is working with the Computer History Musuem in Silicon Valley to record and, in some cases, maintain languages by writing new compilers so our ever-changing hardware can grok the code. Why bother? "They tell us about the state of software practice, the minds of their inventors, and the technical, social, and economic forces that shaped history at the time," Booch explains. "They'll provide the raw material for software archaeologists, historians, and developers to learn what worked, what was brilliant, and what was an utter failure." Here's a peek at the strongest branches of programming's family tree. For a nearly exhaustive rundown, check out the Language List at *HTTP://www.informatik.uni-freiburg.de/Java/misc/lang_list.html*. - Michael Mendeno

Key

- 1954 Year Introduced
- **Active:** thousands of users
- **Protected:** taught at universities; compilers available
- **Endangered:** usage dropping off
- **Extinct:** no known active users or up-to-date compilers
- **Lineage continues**

1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001

Fortran — Once one of the most widely used languages in science and engineering.
Fortran IV — Created for the IBM 7090/94
Fortran 90 ISO/IEC
Flow-Matic — Originally B-O. One of the first English-like data-processing languages.
Cobol — The culprit behind many Y2K bugs. Also known as Common Business-Oriented Language.
Cobol 61
ANSI Cobol 74
ANSI Cobol 85
OO Cobol
IPL — Possibly the first list-processing language used in AI guru Ed Feigenbaum's PhD thesis.
Lisp — Invented by John McCarthy at MIT, Lisp has an unusual syntax made up of lots of nested parantheses. Still popular with AI researchers.
Scheme — A Lisp-like language designed by MIT researcher Seymour Papert to make programming accessible to children through graphics and simple syntax.
A Lisp offshoot that's still used in introdur-tory programming classes at universities.
Scheme 84
Scheme IEEE
Scheme R5RS
Haskell 1.0
Haskell 1.1
Logo
Object Logo
Clos — Common Lisp Object System.
Common Lisp
ANSI Common Lisp
George Radin's attempt to combine the best features of Fortran, Cobol, and Algol 60.
PL/1 — Algorithmic Language, designed as a portable language for scientific computations. Algol 68's complexity was either ahead of its time or rightfully doomed, depending on whom you ask.
ANSI PL/1
PL/M — Programming Language/Microcomputers
Self — A simple object-oriented language.
Self 4.0
NetRexx
Initially called Oak. Still one of the fastest-growing languages around, despite the standards fued between Sun and Microsoft. Somewhat like C++, Java allows for "write once, run anywhere" portability across the Net.
Smalltalk — The first language to show off the power of object-oriented code. Developed by Software Concepts Group, Xerox PARC, led by Alan Kay.
The version Xerox released to the public.
Smalltalk-80
Algol 58 — The hoped-for Esperanto of the computing world. Designed by an international Zurich-based committee as a universal language, it was one of the first attempts at making software more portable. Initially called International Algebraic Language.
Algol 68
Rex 1.00
Rex 2.00 — Restructured Extended Executor
Object Rexx
Simula — Popular in Europe during the '70s, Simula introduced the now-standard concept of object-oriented, rather than procedural, programming. Desgined by Kristen Nygaard and Ole-Johan Dahl.
Simula 67 — All-purpose version of Simula
An object-oriented language.
A simple language (A = argument, B = basic varue, C = ?).
Interactive object-oriented language.
Java
Java 2(v1.3)
Prolog — Programming en Logique for natural-language processing. Popular for AI programs.
Clu — Called Cluster, it's an object-oriented academic language created to teach rigid engineering skills.
Marco and text-processing language used by Unix admins to manipulate large config files programmatically, rather than editing them by hand.
Objective-C
Eiffel
Eiffel 3
Microsoft's answer to Java, it is a key component of the Microsoft.Net platform for Web services.
New AWK, a pattern scanning and pro-cessing language.
ABC
Sather 0.1
Ruby
An object oriented language.
AWK
NAWK
A popular language among Web site builders, it includes features missing from Perl.
Python
Python 1.6
ECMA Script
Possibly the most commen language today. Adds object-oriented features to C.
Tool Command Language, "tickle." The duct tape of programming (a scripting language for patching together different languages).
Cmm
C--, a scripting language.
JavaScript
JScript
Microsoft's version of JavaScript. Can't they share anything? :-)
C#
One of the most widely deployed languages today. Windows and Unix OSes are written mostly in C and its descendants.
C with Classes
C++
Internet C++
Concurrent C
Tcl
Tcl/Tk
Tcl/Tk 8.1
BCPL — Basic Combined Programming Language
B — Used primarily for non-numeric programming.
C
ANSI C (C89)
ISO C (C95)
Updates C++ for the Net with a Java-like virtual machine, so code can run on any kind of computer.
ADA — The US Department of Defense's effort to craft a standard object-oriented language for its work. Named after Ada Lovelace, arguably the world's first computer programmer, and created by Jean Ichbiah's team of Honeywell.
ADA 83
ADA 95
A teaching language named for French mathematician Blaise Pascal. Designed by legendary Swiss programmer Niklaus Wirth for simplicity, in reaction to the complexity of Algol 68.
Modula — Modular Language
Modula 3
Found in millions of Web pages. Originally dubbed LiveScript, it was renamed by Netscape marketers who licensed the name to ride Java's buzz. It has little in common with that language.
Modula 2 ISO
Object-oriented Pascal, designed for simplicity.
Object Pascal
Oberon
Another creation of Niklaus Wirth. Later modified by Robert Griesemer for numerical apps on supercomputers.
Delphi
Pascal
The swiss Army Knife of programming (a k a Practical Extraction and Report Language), used for patching together different languages. Spawned a quasi-literary culture that writes Perl haiku
Csh
Perl 1.000
Perl 4.000
String-processing language for text and formula manipulation, common in text processers. Invented by David Farber, Ralph Griswold, and Ivan Polonsky at Bell Labs.
Programmer's easy street - with this descendant of Snobol, there's no need to know the underlying OS.
sh
Ksh
Frame-based language.
King of the one-liners, also referred to as A Programming Language. An algebra-like language with a special set of characters for writing complex programs as short formulas. Designed by Kenneth Iverson at Harvard.
Snobol
Snobol4
A general-purpose language known for its beauty and grace. Designed by Ralph Griswold to be successor to Snobol4.
C-Shell, a scripting language and command-shell interpreter. Written by C programmers to make Unix command lines more like C's.
Standard ML, a general-purpose language.
The kitchen sink of command-line programming features for Unix, based on the original Bourne shell (a k a Bourne Again Shell).
Bash
ML — Meta Language
Icon
SML
SML 90
SML 97
Page-description language for printers and graphics sistems; 75 percent of all commercial documents are produced on PostScript printers.
Caml
Caml 2-6.1
Objective Caml
O Caml 2
O Caml 3.00
Word-based language first used to guide the National Radio Astronomy Observatory telescope at Kitt Peak in Arizone.
Forth
PostScript
PostScript Level 2
J
Although mocked by "real" programmers for its limited abilities, Basic has outlived many more advanced languages, as well as the RadioShack TRS-80 computers that made it a household word. Stands for Bigenner's All-Purpose Symbolic Instruction Code.
APL
Categorical Abstract Machine Language
APL 96
Basic
Microsoft Basic
MS Basic 2.0
Visual Basic
The Rodney Dangerfield of programming languages. Popular for building Web sites with Microsoft Visual Studio tools.

Survival of the Fittest

Reasons a language endures, with examples of some classic tongues

- **Appeals to a wide audience** C (bolstered by the popularity of Unix)
- **Gets a job done** Cobol (designed for business-report writing)
- **Delivers new functionality** Java (runs on any hardware platform)
- **Fills a niche** Mathematica (speeds up complex computations)
- **Offers a modicum of elegance** Icon (has friendly, line-oriented syntax)
- **Has a powerful user base or backer** C# (developed by Microsoft for .Net)
- **Has a charismatic leader** Perl (programmer-author Larry Wall)

Sources: Paul Boutin; Brent Hailpern, associate director of computer science at IBM Research; The Retrocomputing Museum; Todd Proebsting, senior researcher at Microsoft; Gio Wiederhold, computer scientist, Stanford University

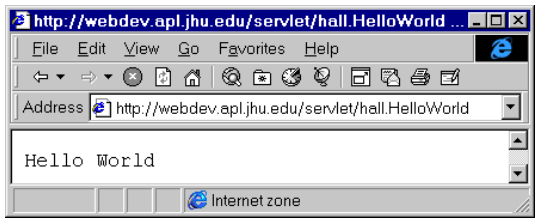
Γλώσσες βασισμένες στην C

«Όταν μάθετε C όλες οι άλλες γλώσσες θα είναι εύκολες»

- ***C++*** περιλαμβάνει όλα τα πλεονεκτήματα της C, αλλά περιλαμβάνει **κλάσεις (classes)** και άλλες έννοιες για **Αντικειμενοστρεφή προγραμματισμό.**
  - π.χ., Windows Systems & Applications, Libraries, κτλ.
- ***Java*** βασίζεται στην C++ αλλά την απλοποιεί εισάγοντας **απλουστευμένη σύνταξη**, διαχειριστή μνήμης (garbage collector), κ.α.
  - Web Applets, Enterprise Progr. (DBs), **Android**, etc.
- ***C#*** είναι πλέον η πιο διαδεδομένη γλώσσα της Microsoft βασισμένη στο μοντέλο της Java.
  - Microsoft .NET εφαρμογές. **Window Phone**, Cloud, κτλ.
- **Objective-C** είναι η γλώσσα για πλατφόρμες Apple
  - Ουσιαστικά πρόκειται για C με απλή αντικειμενοστρέφια
  - Π.χ., Εφαρμογές για Apple Mac, IPhone, IPad, IPod, κτλ.

Γλώσσες Bo

«Ομοιότητα σύνταξης
γλώσσες δίνουν περισσ



C (GNU, Cross-Platform) - Procedural

```
// hello.c
#include <stdio.h>
int main(int argc, const char *argv[] ) {
    printf( "hello world\n" );
    return 0;
}
```

C++ (GNU, Cross-Platform) – Obj. Orie.

```
// hello.cpp
#include <iostream> using namespace std;
int main () {
  cout << "Hello World!";
  return 0;
}
```

JAVA (Sun/Oracle, Cross-Platform) – OO

```
// hello.java
public class hello {
   public static void main (String args []) {
      System.out.println("Hello world");
   }
}
```

C# (Microsoft) - OO

```
// Hello1.cs
public class Hello1 {
   public static void Main() {
      System.Console.WriteLine("Hello, World!");
   }
}
```

Objective-C (Apple) – Proc. ή OO

```
// hello.m
#import <stdio.h>
int main( int argc, const char *argv[] ) {
    printf( "hello world\n" );
    return 0;
}
```

JAVA (Servlet)

```
package hall;
import java.io.*;
import javax.servlet.*;
public class HelloWorld extends HttpServlet {
  public void doGet(HttpServletRequest request,
     HttpServletResponse response)
      throws ServletException, IOException {
    PrintWriter out = response.getWriter();
    out.println("Hello World");  }
}
```


Dr. Andreas Aristidou a.aristidou@ieee.org

Το πρώτο πρόγραμμα C

```
#include <stdio.h>

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

- Το πιο πάνω πρόγραμμα αποθηκεύεται σε αρχείο με όνομα `hello.c`.
- Το όνομα του αρχείου μπορεί να είναι οτιδήποτε, αλλά η κατάληξη `.c` συχνά απαιτείται από τους μεταγλωττιστές.

Μεταγλώττιση & Σύνδεση

- Πριν εκτελεστεί ένα πρόγραμμα, τρία βήματα είναι συνήθως απαραίτητα:
  - ***Προεπεξεργασία (Preprocessing).*** Επεξεργασία εντολών αρχείου που ξεκινάνε με #, γνωστά ως οδηγίες προεπεξεργαστή **(*directives*)**
  - ***Μεταγλώττιση (Compiling).*** Μεταγλώττιση του αρχείου σε γλώσσα μηχανής (***object code***).
  - ***Σύνδεση (Linking).*** Ο ***linker*** συνθέτει το object code των επί μέρους αρχείων με οτιδήποτε επιπλέον κώδικα απαιτείται για να παραχθεί ένα εκτελέσιμο αρχείο.
- Ο preprocessor είναι συνήθως μέρος του compiler και όλα τα πιο πάνω εκτελούνται με τη μια όπως θα δούμε στην επόμενη διαφάνεια.

Μεταγλώττιση & Σύνδεση

- Ο μεταγλωττιστής C του UNIX είναι ο CC.

  `% cc hello.c` (όπου `%` η γραμμή εντολών UNIX)

- Το Linking γίνεται αυτόματα (εάν και μπορεί να γίνει επιλεκτικά με τη `ld` εντολή, θα το δούμε αργότερα)
- Το αποτέλεσμα είναι το `a.out` αρχείο (προεπιλογή) το οποίο είναι το **εκτελέσιμο πρόγραμμα** (executable)
  - Το όρισμα `-o` επιτρέπει τον προσδιορισμό του ονόματος του εκτελέσιμου

```
% cc -o hello hello.c  ; ./hello
```

Εάν το μονοπάτι μεταγλώττισης δεν είναι στην μεταβλητή περιβάλλοντος PATH, τότε απαιτείται το «./»

Dr. Andreas Aristidou a.aristidou@ieee.org

Μεταγλώττιση & Σύνδεση

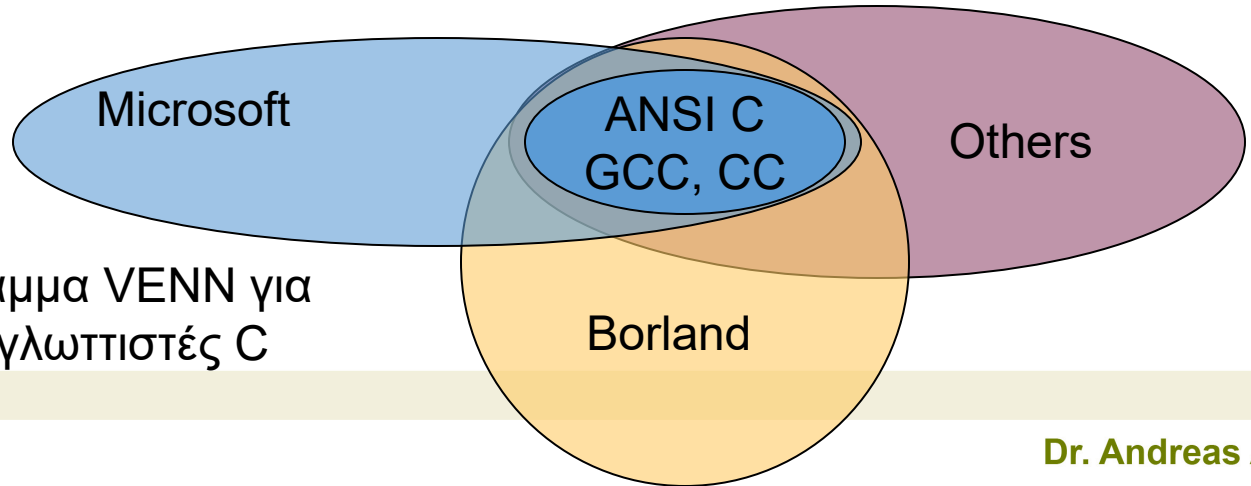
- Ο **GNU GCC** είναι ο πιο γνωστός μεταγλωττιστής σε C.
- Στο μάθημα θα χρησιμοποιήσουμε ΜΟΝΟ **τον GNU GCC μεταγλωττιστή,** η χρήση του οποίου είναι όμοια με το CC.

```
% gcc -o hello hello.c
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Μεταγλώττιση & Σύνδεση

- Η C αποτελείται από ένα σύνολο συντακτικών κανόνων.
- Για να εκτελεστεί ένα πρόγραμμα C πρέπει να χρησιμοποιηθεί ένας **μεταγλωττιστής (compiler),** ο οποίος κατασκευάζεται από διάφορες εταιρείες και οργανισμούς.
- Η **American National Standard Institutes (ANSI)** δημιούργησε το πρότυπο ANSI C για λόγους μεταφερσιμότητας (portability) του κώδικα το οποίο καλούνται οι διάφορες εταιρείες να ακολουθούν.
  - Εμείς θα χρησιμοποιήσουμε τον GNU GCC ο οποίος είναι συμβατός με την ANSI C18 (2018), C11 (2011) και C99 (1999) έκδοση, ενώ αρκετοί άλλοι υποστηρίζουν μόνο C89 και άλλες προεκτάσεις εκτός προτύπου.



Διάγραμμα VENN για μεταγλωττιστές C

Μεταγλώττιση & Σύνδεση

- Η γενική μορφή ενός προγράμματος έχει την ακόλουθη μορφή

```
directives

int main(void)
{
   statements
}
```

Η C χρησιμοποιεί τα `{` και `}` με παρόμοιο τρόπο που άλλες γλώσσες χρησιμοποιούν τις λέξεις `begin` και `end`.

Dr. Andreas Aristidou a.aristidou@ieee.org

Οδηγίες προεπεξεργαστή

- Προτού ένα πρόγραμμα C μεταγλωττιστεί, επεξεργάζεται πρώτα από από έναν προεπεξεργαστή.
- Οι εντολές που προορίζονται για τον προεπεξεργαστή ονομάζονται οδηγίες (directives).
- Παράδειγμα:

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

- `<stdio.h>` είναι ένα ***header*** που περιέχει πληροφορίες σχετικά με τα πρότυπα της C's σε βιβλιοθήκη I/O.

Οδηγίες προεπεξεργαστή

- Οι οδηγίες ξεκινούν πάντα με τον χαρακτήρα #.
- Από προεπιλογή, οι οδηγίες γίνονται μόνο σε μία γραμμή. Δεν υπάρχει ερωτηματικό ή άλλος ειδικός δείκτης στο τέλος.

Dr. Andreas Aristidou a.aristidou@ieee.org

Συναρτήσεις

- Η ***συνάρτηση*** (function) είναι μια σειρά δηλώσεων που έχουν ομαδοποιηθεί και τους έχει δοθεί ένα όνομα.
- Οι ***συναρτήσεις βιβλιοθήκης*** (Library functions) παρέχονται ως μέρος της υλοποίησης της C.
- Μια συνάρτηση που υπολογίζει μια τιμή χρησιμοποιεί την δήλωση `return` για να ορίσει την τιμή που επιστρέφει:

```
return x + 1;
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Η συνάρτηση **`main`**

- Η συνάρτηση `main` είναι υποχρεωτική.
- Η `main` είναι ειδική συνάρτηση: καλείται αυτόματα όταν το πρόγραμμα εκτελείται.
- Η `main` επιστρέφει έναν κωδικό κατάστασης. Η τιμή 0 δηλώνει κανονικό τερματισμό του προγράμματος.
- Αν δεν υπάρχει η δήλωση `return` στο τέλος της `main` συνάρτησης, πολλοί μεταγλωττιστές θα παράγουν ένα προειδοποιητικό μήνυμα.

Dr. Andreas Aristidou a.aristidou@ieee.org

Εντολές

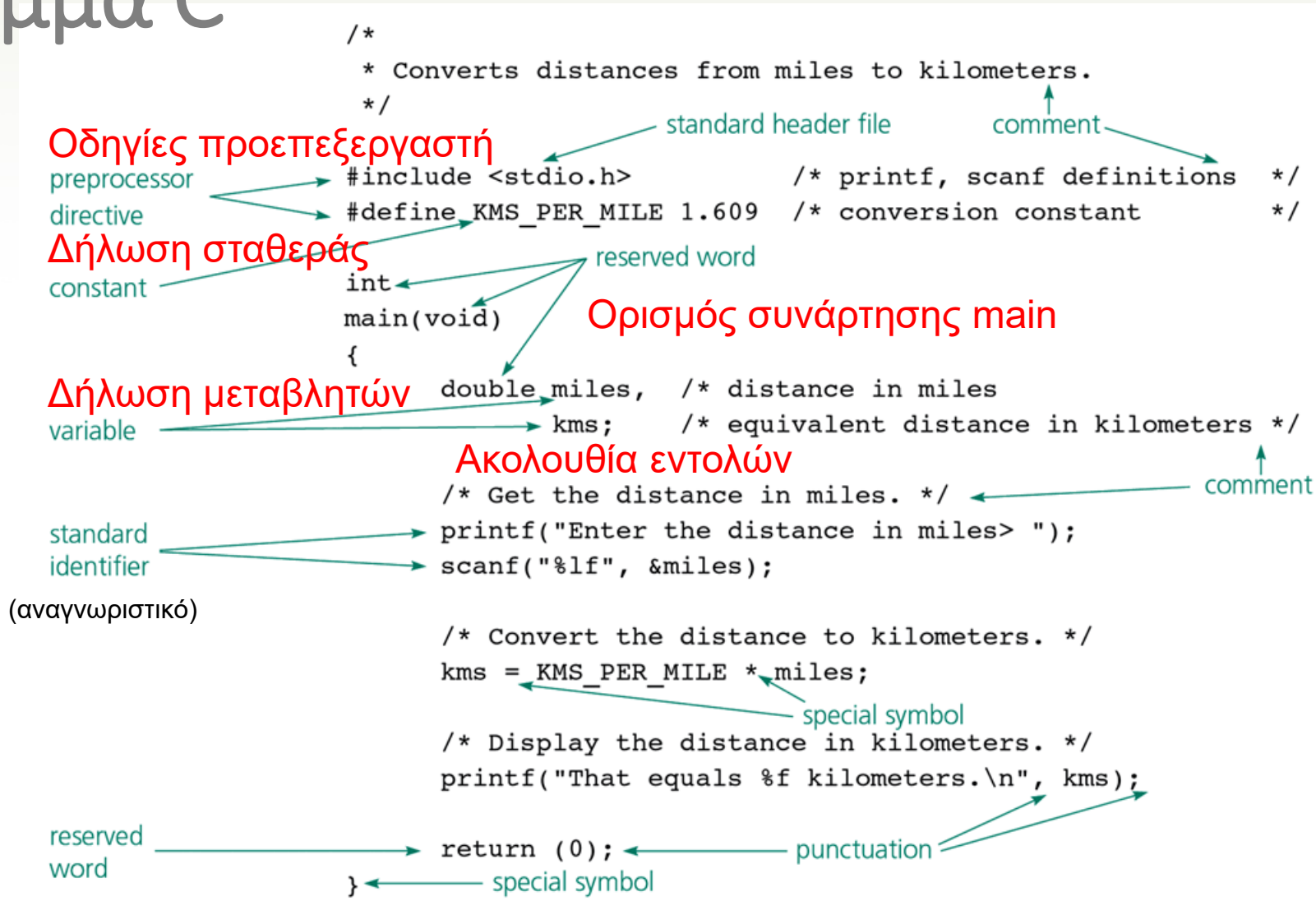
- Η ***εντολή*** (statement) είναι μια εντολή που θα εκτελεστεί όταν το πρόγραμμα εκτελείται.
- Για παράδειγμα, η `hello.c` χρησιμοποιεί μόνο δύο είδη εντολών. Η μία είναι το `return`; και η άλλη είναι το ***function call.***
- Η `hello.c` καλέι την `printf` να τυπώσει το ακόλουθο:

```
printf("To C, or not to C: that is the question.\n");
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Μετατροπή σε χιλιόμετρα από μίλια

Το δεύτερο πρόγραμμα C



Σφάλματα Μεταγλώττισης

- Υπάρχουν πολλαπλοί Λόγοι
  - **Pre-Processor (**Προεπεξεργαστή, gcc -E)
    - Π.χ., #include <non-existent-library.h>
  - **Parser** (Συντακτικός Αναλυτής):
    - Π.χ., ξεχνάμε να κλείσουμε μια παρένθεση.
  - **Assembler** (Συμβολομεταφραστή, as):
    - Μετατρέπει την **συμβολική γλώσσα (assembly)** σε **αντικειμενικό κώδικα (object code)**
    - Σπάνια λάθη που σχετίζονται με τον As
  - **Linker (ld):** Σύνδεση με μη-υπαρκτή συνάρτηση

Dr. Andreas Aristidou a.aristidou@ieee.org

Σφάλματα Μεταγλώττισης

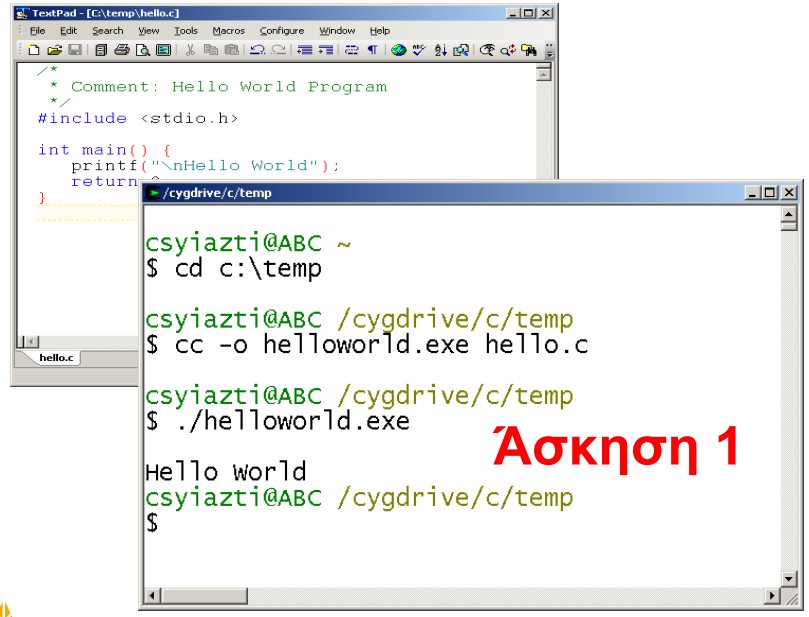
- Εάν ο `gcc` μπερδευτεί, τότε παρουσιάζονται εκατοντάδες μηνύματα:
  - Διορθώστε το πρώτο, μετά δοκιμάστε ξανά – αγνοώντας τα υπόλοιπα.
- Ο `gcc` θα δημιουργήσει ένα εκτελέσιμο με **προειδοποιήσεις (warnings)**, αρκεί να μην υπάρχει **λάθος (error)**
  - Μην αγνοείτε τις προειδοποιήσεις!
  - Κάνετε χρήση του `gcc -Wall` για να παρουσιάσετε ΌΛΕΣ τις προειδοποιήσεις, π.χ., :
    - `if (x = 0)` **VS.** `if (x == 0)`
    - `example.c:3: warning: suggest parentheses around assignment used as truth value`

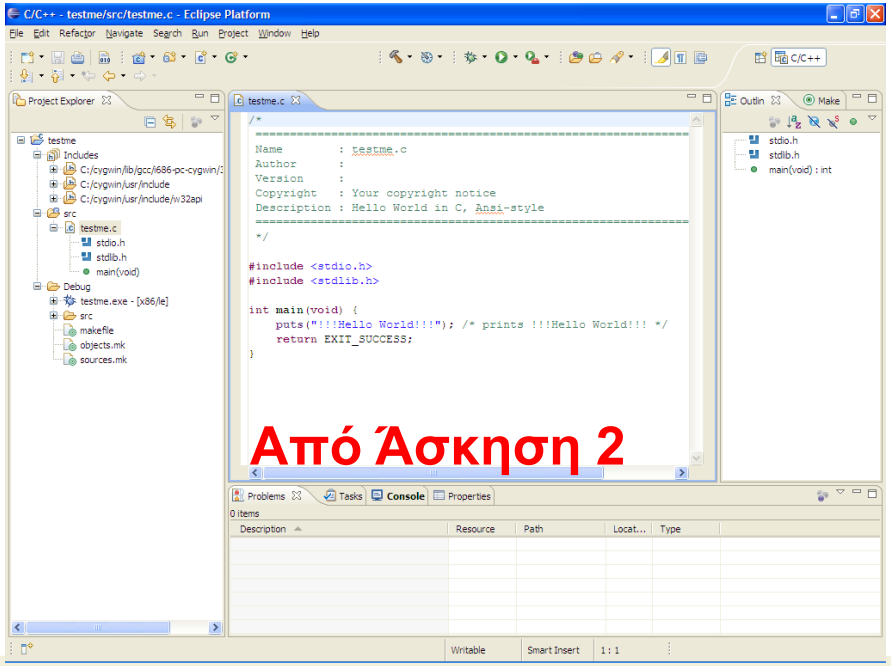
Dr. Andreas Aristidou a.aristidou@ieee.org

Integrated Development Environments

Ολοκληρωμένα Περιβάλλοντα Ανάπτυξης

- Ένα ***Integrated Development Environment (IDE)*** είναι ένα λογισμικό το οποίο επιτρέπει τη συγγραφή, μεταγλώττιση, εκτέλεση και αποσφαλμάτωση ενός προγράμματος.
- Για αρχή θα χρησιμοποιήσουμε τον συνδυασμό **κελύφους με κάποιο κειμενογράφο** για προγραμματιστές και στη συνέχεια του μαθήματος θα **λειτουργούμε μόνο με το IDE**.





Το πρόγραμμα σας ΠΡΕΠΕΙ να μεταγλωττίζεται στις μηχανές του εργαστηρίου εναλλακτικά θα μηδενίζεται η άσκηση σας!!!

Native vs. Intermed. Compile [C/C++/Objective-C vs. JAVA/Android/C#]

- **C/C++/Objective-C(iOS) :** Ο πηγαίος κώδικας μετατρέπεται πλήρως σε γλώσσα μηχανής κατά την μεταγλώττιση.
- **JAVA, Android, C#:** Ο πηγαίος κώδικας μετατρέπεται σε **ενδιάμεσο κώδικα (bytecode)**, ο οποίος μπορεί να εκτελεστεί σε οπουδήποτε σύστημα έχει το λεγόμενο **Java Virtual Machine (JVM).**
- Αυτό που γίνεται πρακτικά στο **JAVA, Android, C#** είναι ότι το bytecode μετατρέπεται κατά την εκτέλεση του προγράμματος σε κώδικα μηχανής από το (**Just-in-time JIT Compilation)** υποσύστημα του JVM.
  - Στη **Java** JIT γίνεται από το JVM (**Java Virtual Machine**)
  - Στη **C#** JIT γίνεται από το CLR (**Common Language Runtime**)
  - Στο **Android** JIT γίνεται από το DVM (**Dalvik Virtual Machine**), ή το **ART (Android RunTime)** σε εκδόσεις του Android μετά το 4.4 (Έκδοση 8 το 2017).
- Στη JAVA 9 (έκδοση 2017) υπάρχει το **ahead-of-time (AoT)** compilation, όπου η μεταγλώττιση γίνεται at runttime αλλά πριν τη χρήση για λόγους απόδοσης.

Dr. Andreas Aristidou a.aristidou@ieee.org

Σύγκριση C έναντι Java (Σύνοψη)

- Πρόγραμμα Java
  - Συλλογή από κλάσεις
  - Κάθε κλάση περιέχει μια main μέθοδο η οποία είναι μέθοδος εκκίνησης
    - Εκτελώντας `java` *`StartClass`* ενεργοποιεί την μέθοδο *`StartClass`*`.main`
    - Το Java Virtual Machine (JVM), που περιλαμβάνεται ως μέρος του JRE (Java Runtime Environment) ή JDK (Java Development Kit) φορτώνει τις υπόλοιπες κλάσεις οπότε χρειάζεται

Dr. Andreas Aristidou a.aristidou@ieee.org

Σύγκριση C έναντι Java (Σύνοψη)

- Πρόγραμμα C
  - Συλλογή από συναρτήσεις (functions)
  - Μια συνάρτηση με όνομα – `main()` – είναι η συνάρτηση εκκίνησης
  - Εκτελώντας το πρόγραμμα (εξ' ορισμού όνομα a.out) εκκινεί την συνάρτηση main
  - Συνήθως, όλος ο κώδικας ενός προγράμματος περιλαμβάνεται στον εκτελέσιμο κώδικα (**στατική σύνδεση**)
    - Εναλλακτικά, μπορεί να περιλαμβάνεται ο κώδικας αυτός με **δυναμική σύνδεση** (π.χ., .dll, .so)
    - **Στατική σύνδεση:** library.a (κατά μεταγλώττιση)
    - **Δυναμική σύνδεση:** library.so (κατά εκτέλεση)

Σχόλια

- ***/*from to comment*/***
- Σύμβαση για μακρύτερα σχόλια:

```
/*
 * AverageGrade()
 * Given an array of grades, compute the average.
 */
```

Καλύτερη ευκρίνεια ότι υπάρχει σχόλιο σε αυτή τη γραμμή

Θα υπάρχει εξειδικευμένο εργαστήριο για συστάσεις πάνω σε στυλ γραφής σχολίων και κώδικα!

- Αποφεύγετε την χρήση περιέργων κουτιών ****
  - Το βιβλίο εισηγείται το ακόλουθο πλάτους 60 χαρακ.

```
/*********************************************************
 * print_result: Notifies the user of the result, using  *
 *               the external variables set by           *
 *               analyze_hand.                           *
 *********************************************************/
```

  - Επίσης, μη χρησιμοποιείτε ΠΟΤΕ TAB αλλά 3 SPACES στη θέση κάθε TAB εφόσον αυτά δεν αλλάζουν πλάτος μεταξύ κειμενογράφων

Σχόλια

- ***Προσοχή:*** Το να ξεχάσετε να τερματίσετε ένα σχόλιο μπορεί να προκαλέσει το πρόγραμμα μεταγλώττισης να αγνοήσει μέρος του προγράμματός σας:

```
printf("My ");     /* forgot to close this comment...
printf("cat ");
printf("has ");    /* so it ends here */
printf("fleas");
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Σχόλια στην C99

- Στην C99, σχόλια μπορούν επίσης να εγγραφούν με τον ακόλουθο τρόπο:

```
// This is a comment
```

- Αυτό το στυλ σχολίου τελειώνει αυτόματα στο τέλος μιας γραμμής.
- Πλεονεκτήματα της γραφής `//` :
  - Ασφαλέστερο: δεν υπάρχει πιθανότητα ένα σχόλιο χωρίς τερματισμό να πάρει κατά λάθος μέρος του προγράμματος.
  - Τα σχόλια πολλών γραμμών ξεχωρίζουν καλύτερα.

Τι γνωρίζαμε μέχρι σήμερα;

Αριθμητικοί Τύποι Δεδομένων

Σε Αρχιτεκτονική 32 bit (x86)

Τύπος	Bytes	Εύρος Τιμών
char (χαρακτήρας)	1	-128 … 127
unsigned char	1	0..255
short (ακέραιος)	2	-65536…65535
int, long [int] (ακέραιος)	4,4 (x86)	-2,147,483,648 to 2,147,483,647
long long [int] (ακέραιος)	8	2^{64}
float (πραγματικός)*	4	3.4E+/-38 (7 digits)
double (πραγματικός)	8	1.7E+/-308 (15 digits)

** Εάν αποθηκεύσω πραγματικό με τιμή 0.1 μπορεί αργότερα να βρώ ότι έχει τιμή 0.09999999999999987, λόγω λάθους στρογγυλοποίησης*

Dr. Andreas Aristidou a.aristidou@ieee.org

Αριθμητικοί Τύποι Δεδομένων (Επισημάνσεις)

- Επίσης, το **`unsigned`** υποδηλεί **μη-προσημασμένη** τιμή.
  - δηλ., παίρνει μη-αρνητικές τιμές
  - Συνεπώς, χρησιμοποιούνται τα ίδια bits, τα οποία ωστόσο έχουν διαφορετική σημασία στο πρόγραμμα
- `unsigned char` = 1 “character”, μπορεί να υποστηρίξει μόνο ASCII (8-bit), δηλαδή 256 χαρακτήρες (0..255)
  - Το Unicode (16-bit ή 32 bit) υποστηρίζεται στη C μέσω κάποιων επιπλέον βιβλιοθηκών (δες Κεφ. 25: <locale.h>, <wchar.h> και <wctype.h>)

Ο Πίνακας ASCII (πρώτοι 128 χαρακτήρες)

- American Standard Code for Information Interchange

ASCII Code Chart

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Οι 128 ASCII χαρακτήρες, συμπεριλαμβανομένου των μη εκτυπώσιμων χαρακτήρων (αναπαρίστανται με συντομογραφία).

Ο Πίνακας ASCII (πρώτοι 128 χαρακτήρες)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Spa	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DE

Αριθμητικοί Τύποι Δεδομένων (32/64-bit Προγραμ. Μοντέλα)

- Λίγα λόγια για τύπους δεδομένων με τους 2 πιο διαδεδομένα μοντέλα δεδομένων
  - **I**(ntegers) L(ong) P(ointer) 32 => x86 Model
  - L(ong) P(ointer) 64 => x64 Model

Datatype	**ILP32 Model**	**LP64 Model**
char	8	8
short	16	16
int	**32**	32
long	**32 (4 bytes)**	**64 (8 bytes)**
pointer	**32 (4 bytes)**	**64 (8 bytes)**

Η μνήμη μπορεί να έχει ΜΟΝΟ μέχρι $2^{32} = \sim 4 \times 10^9$ (δηλ., 4GB) διευθύνσεις ☹!

Η μνήμη μπορεί να έχει μέχρι 16 Exa ($\times 10^{18}$) διευθύνσεις☺!

Δοκιμάστε στο σπίτι το ακόλουθο:
printf("%d, %d, %d", sizeof(int), sizeof(long), sizeof(void *));

Δηλώσεις

- Οι μεταβλητές πρέπει να ***δηλώνονται*** (declared) προτού χρησιμοποιηθούν.
- Οι μεταβλητές μπορούν να δηλώνονται μία κάθε φορά:

```
int height;
float profit;
```

- Ή αρκετές ταυτόχρονα:

```
int height, length, width, volume;
float profit, loss;
```

Dr. Andreas Aristidou a.aristidou@ieee.org

Δηλώσεις

- Όταν η `main` περιέχει δηλώσεις (declarations), πρέπει να προηγούνται των εντολών (statements):

```
int main(void)
{
   declarations
   statements
}
```

- Αυτό δεν είναι απαραίτητο για την C99.

Dr. Andreas Aristidou a.aristidou@ieee.org

Εκτυπώνοντας μια Μεταβλητή

- **`printf("%d", variable)`**
  - `%d` integer, `%ld` (64 bit), `%x` hex., `%o` octal
  - `%f` float (6 δεκ. ψηφ.), `%.2f` (2 δεκ. ψηφ.)
  - `%c` char
  - `%s` string (πίνακας char με τελικό NUL (\0))
- Υπάρχουν εκατοντάδες ορίσματα κάποια εκ' των οποίων θα δούμε συνοπτικά στην ερχόμενη διάλεξη.
  - το Κεφ. 22 καλύπτει το θέμα σε περισσότερο βάθος το οποίο δε θα χρειαστεί για αυτό το μάθημα.

Διαβάζοντας μια Μεταβλητή

- `scanf("%f", &x);`
  - `%d` integer, `%ld` (64 bit), `%x` hex., `%o` octal
  - `%f` float
  - `%c` char
  - `%s` string (πίνακας char με τελικό NUL (\0))

- Και πάλι, υπάρχουν εκατοντάδες ορίσματα κάποια εκ' των οποίων θα δούμε συνοπτικά στην ερχόμενη διάλεξη.
  - το Κεφ. 22 καλύπτει το θέμα σε περισσότερο βάθος το οποίο δε θα χρειαστεί για αυτό το μάθημα.

Αναγνωριστικά της C (Identifiers)

- Τα ονόματα **μεταβλητών**, **συναρτήσεων**, **μακρο-εντολών** και **οντοτήτων** ονομάζονται **αναγνωριστικά (*identifiers*).**
  - Η C όπως και η JAVA και C# (αντίθετα με π.χ., VB) είναι **ευαίσθητη στο τύπο χαρακτήρα (case sensitive)**
  - Ένα αναγνωριστικό μπορεί να περιέχει **letters**, **digits**, και **underscores**, αλλά πρέπει πάντα να ξεκινά με **letter** ή **underscore**:

```
times10   get_next_char   _done    OK
10times   get-next-char             ERROR
```

- Επιβάλλεται ένα ομοιόμορφο στυλ, π.χ.,
  - `symbol_table` ή `symbolTable`
  - 3 κενά spaces ανά εμφωλευμένη έκφραση (όχι tabs)
  - 80 στήλες ΜΟΝΟ. (περισσότερα στο εργαστήριο 3)

A word on case

- ~~Sentence Case~~
- UPPERCASE
- lowercase
- camelCase
- ***PascalCase***
- hypen-case
- snake_case

Dr. Andreas Aristidou a.aristidou@ieee.org

Δεσμευμένα Αναγνωριστικά

- Οι ακόλουθες **λέξεις κλειδιά** (***keywords)*** ΔΕΝ μπορούν να χρησιμοποιηθούν για αναγνωριστικά στην C:

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

***C99 only**

Dr. Andreas Aristidou a.aristidou@ieee.org