



# Προγραμματισμός Υπολογιστών & Υπολογιστική Φυσική

Ενότητα 1: Εισαγωγή Στον Προγραμματισμό

Νικόλαος Στεργιούλας  
Τμήμα Φυσικής



Ευρωπαϊκή Ένωση  
Ευρωπαϊκό Κοινωνικό Ταμείο



ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ  
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ  
*επένδυση στην κοινωνία της γνώσης*  
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ  
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

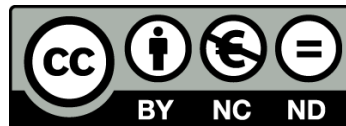
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης



ΕΣΠΑ  
2007-2013  
πρόγραμμα για την ανάπτυξη  
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

# Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.



# Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «Ανοικτά Ακαδημαϊκά Μαθήματα στο Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.



# ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΥΠΟΛΟΓΙΣΤΩΝ & ΥΠΟΛΟΓΙΣΤΙΚΗ ΦΥΣΙΚΗ

Μέρος 1ο

ΝΙΚΟΛΑΟΣ ΣΤΕΡΓΙΟΥΛΑΣ



ΤΜΗΜΑ ΦΥΣΙΚΗΣ

ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ

# ΓΙΑΤΙ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ;

Στα μαθηματικά και στη φυσική συχνά έχουμε να κάνουμε **υπολογισμούς** που ακολουθούν μια συγκεκριμένη **διαδικασία**

**ΠΑΡΑΔΕΙΓΜΑ: Εύρεση ρίζας**

Έστω ότι θέλουμε να υπολογίσουμε τη  $\sqrt{2}$ . Ένας τρόπος είναι να σχηματίσουμε την εξίσωση

$$f(x) = x^2 - 2 = 0$$

Αν η ρίζα είναι η  $x = x_0$  ( $= 1.414213562\dots$ ) τότε το ανάπτυγμα Taylor γύρω από τη ρίζα είναι

$$f(x) = \cancel{f(x_0)} + f'(x_0)(x - x_0) + \dots$$

# ΓΙΑΤΙ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ;

---

Άρα

$$x_0 \simeq x - f(x)/f'(x_0)$$

και αν στην περιοχή της ρίζας  $f'(x) \simeq f'(x_0)$

τότε

$$x_0 \simeq x - f(x)/f'(x)$$

και μπορούμε να δημιουργήσουμε την αναδρομική σχέση (μέθοδος Newton-Raphson)

$$x^{(n+1)} = [x - (f/f')]^{(n)}$$

Ξεκινάμε με μια αρχική εκτίμηση, π.χ.

$$n = 1 : x^{(1)} = 1.0$$

# ΓΙΑΤΙ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ;

και υπολογίζουμε διαδοχικά:

n	$x^{(n)}$	$x^{(n+1)}$	f(x)
1	1.0	1.5	2.5e-01
2	1.5	1.41667	7e-03
3	1.41667	1.414216	6e-06

Η παραπάνω διαδικασία είναι ένας **ΑΛΓΟΡΙΘΜΟΣ**, που μας επέτρεψε να υπολογίσουμε  $\sqrt{2} \simeq 1.414216$

# ΑΝΑΛΥΣΗ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ


Ο αλγόριθμος Newton-Raphson για το συγκεκριμένο παράδειγμα έχει την εξής δομή:

$$n = 1$$

αρχικοποίηση δείκτη

$$x^{(n)} = 1.0$$

αρχικοποίηση μεταβλητής


$$x^{(n+1)} = [x - f/f']^{(n)}$$

υπολογισμός νέας τιμής

$$x^{(n)} = x^{(n+1)}$$

αντικατάσταση παλαιάς τιμής από νέας

$$n \leftarrow n + 1$$

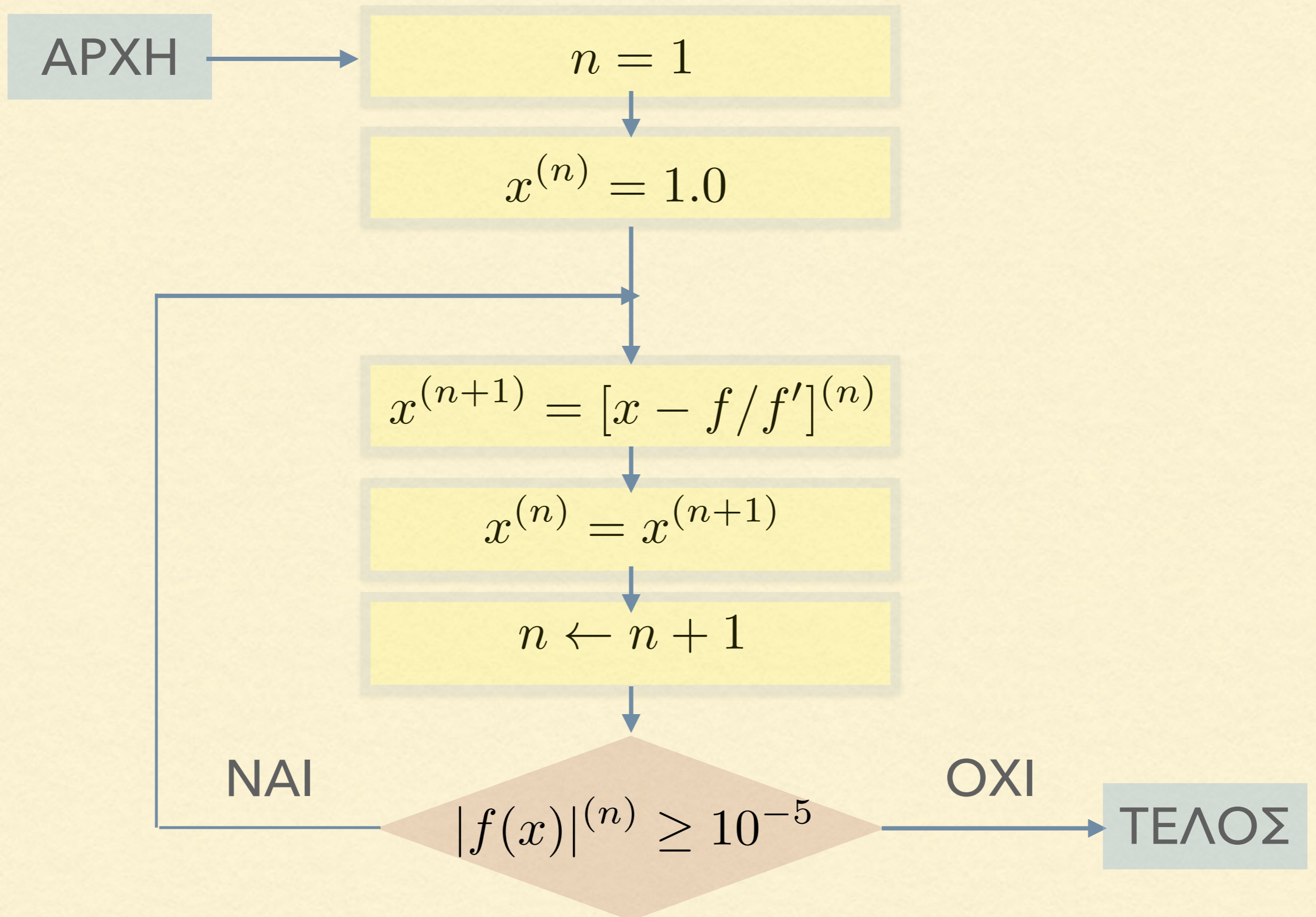
αύξηση δείκτη

$$|f(x)|^{(n)} < 1e - 5$$

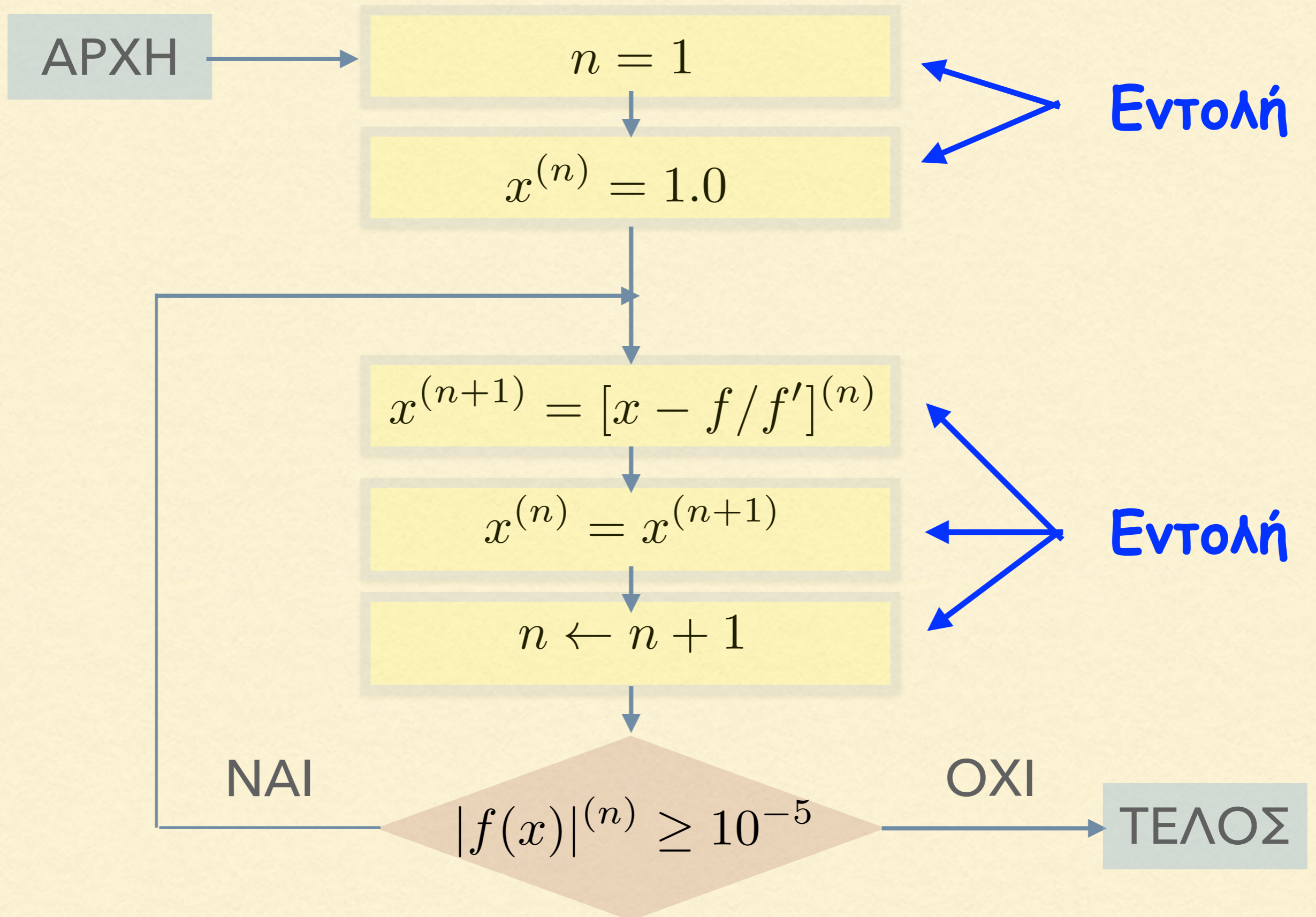
συνθήκη τερματισμού



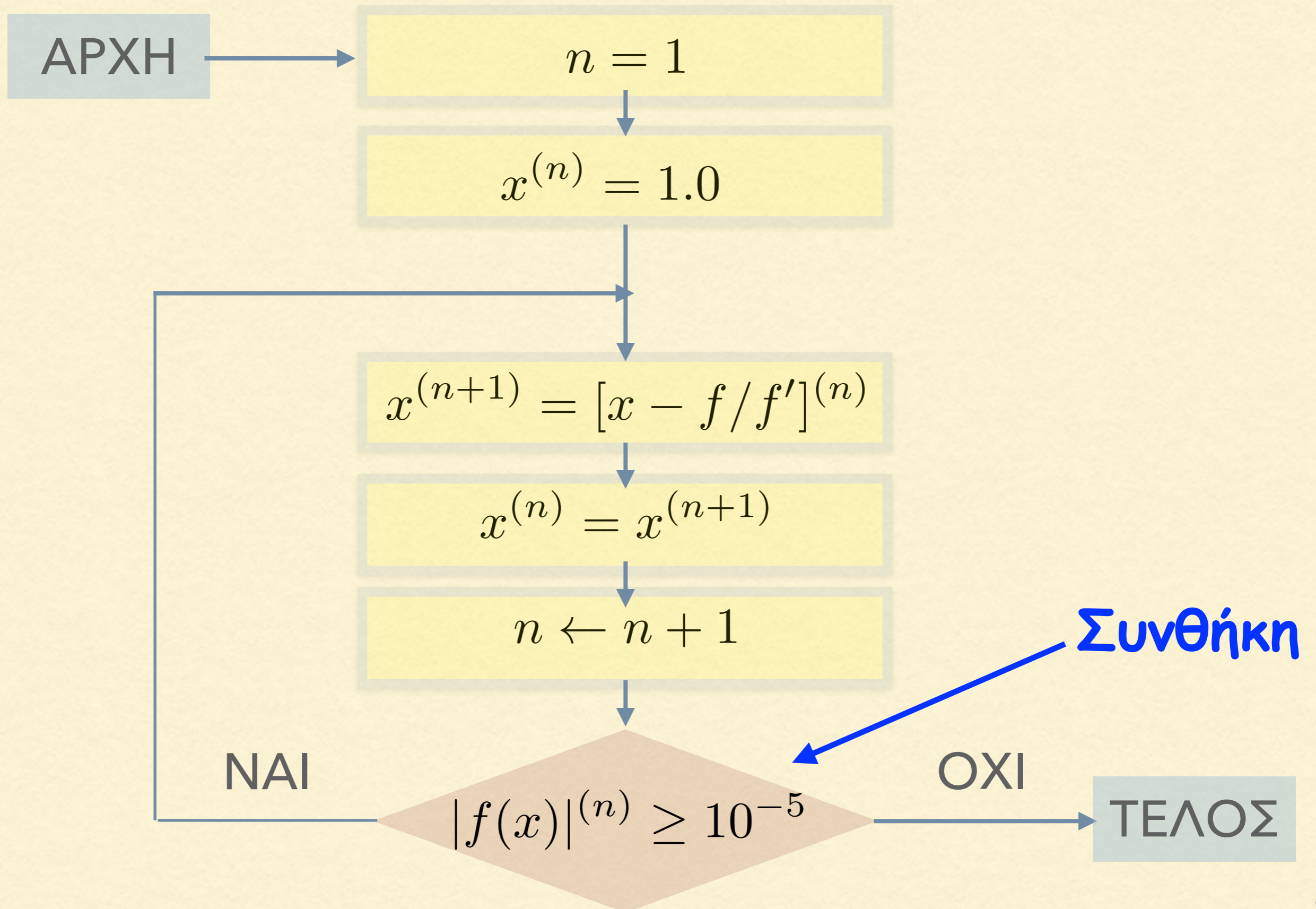
# ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ



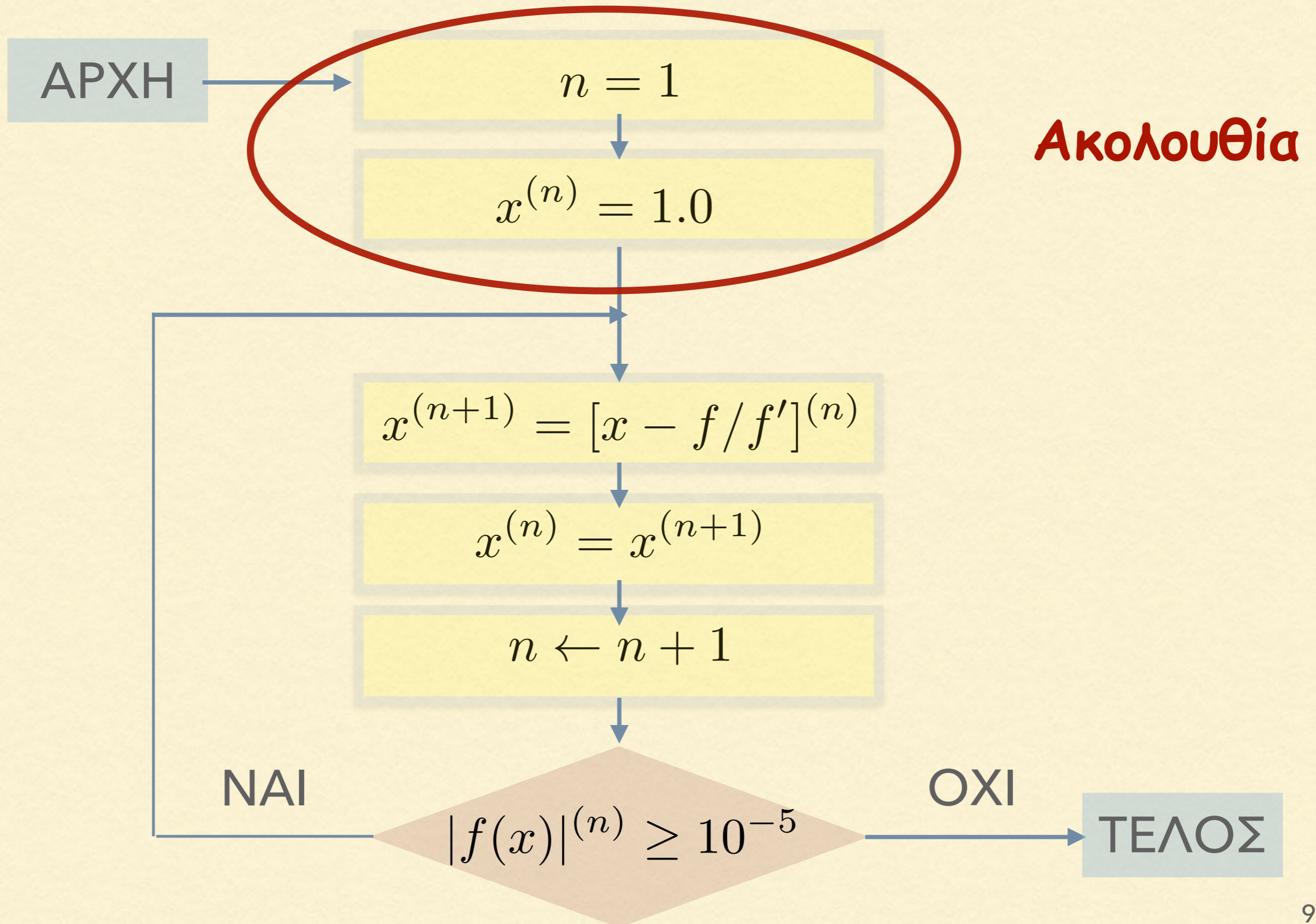
# ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ



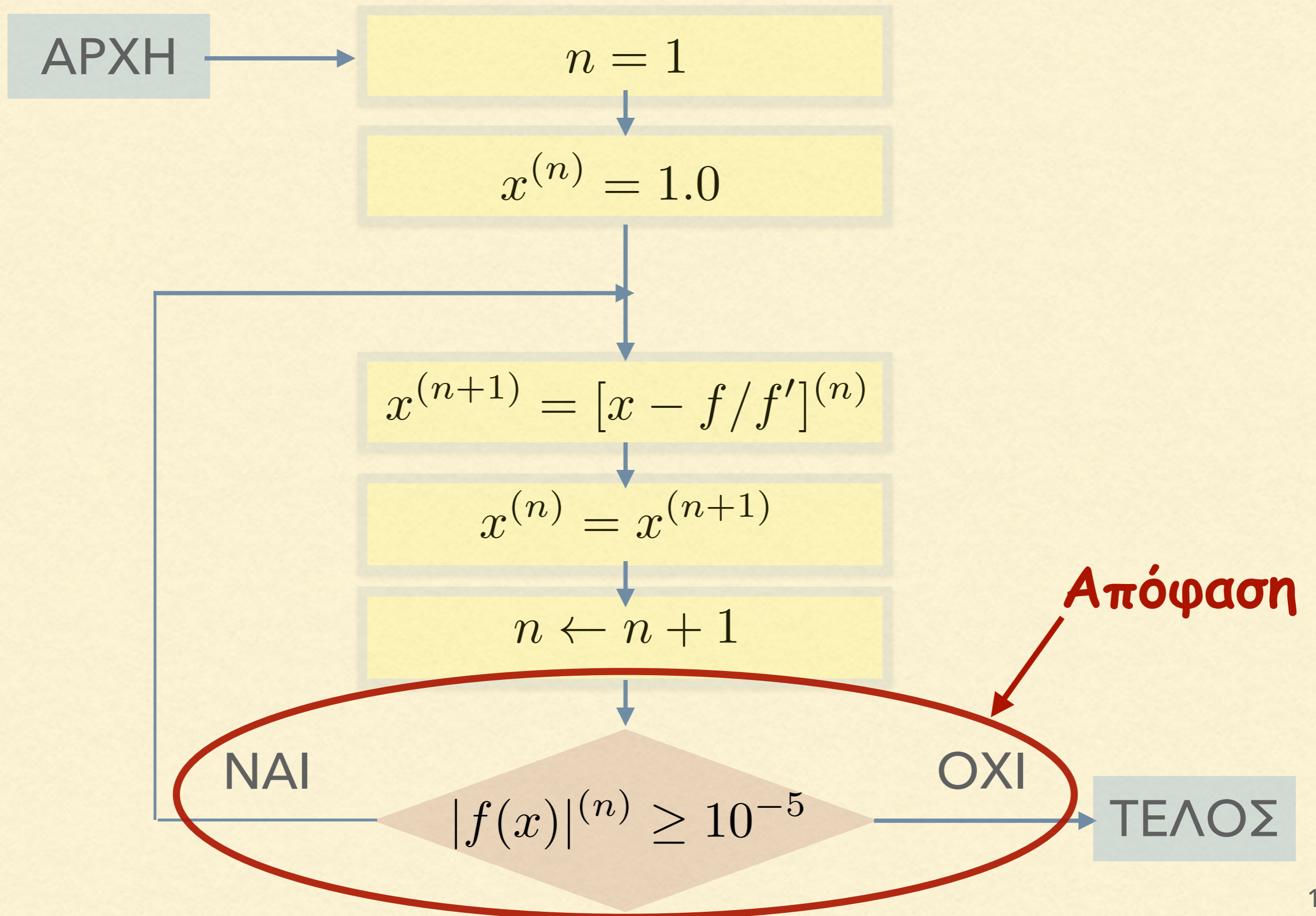
# ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ



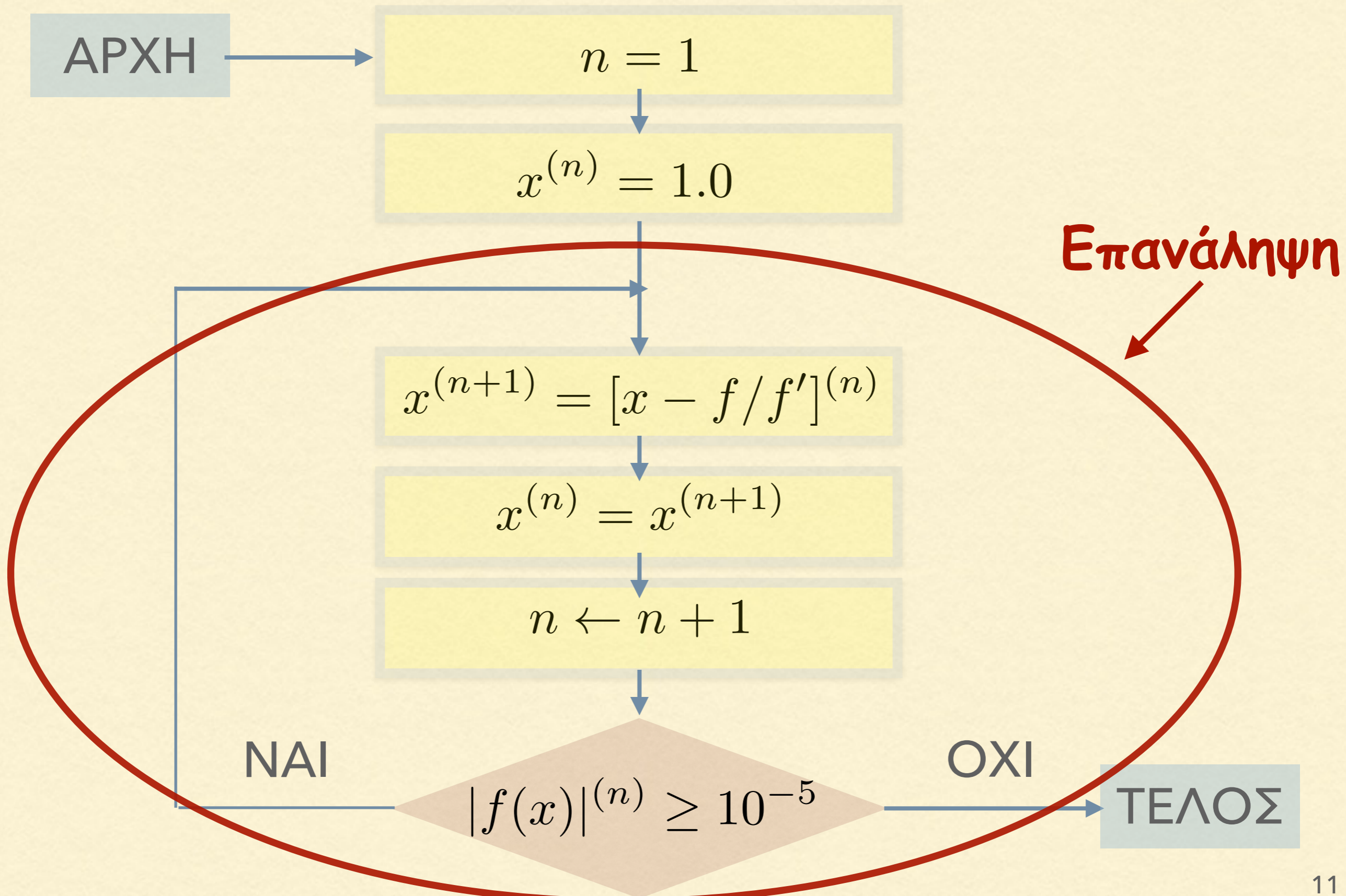
# ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ



# ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ



# ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ



# ΒΑΣΙΚΕΣ ΔΟΜΕΣ

---

Οι βασικές δομές του προγραμματισμού είναι:

- Η **ακολουθία** εντολών

είναι μια σειρά από εντολές που εκτελούνται διαδοχικά

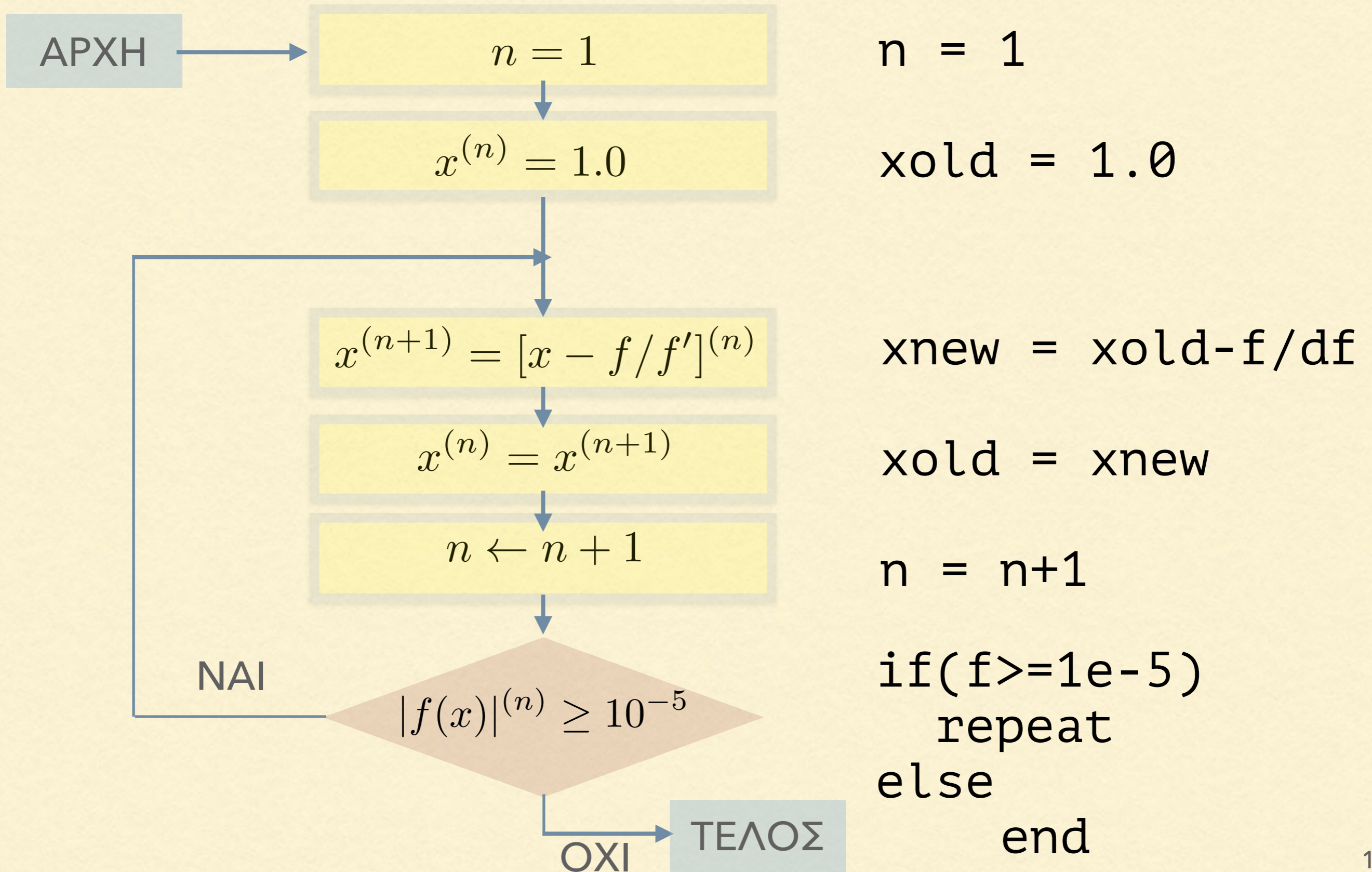
- Η **απόφαση**

ελέγχεται μια συνθήκη και αναλόγως εάν είναι αληθινή ή όχι εκτελείται μια εντολή ή κάποια άλλη

- Η **επανάληψη**

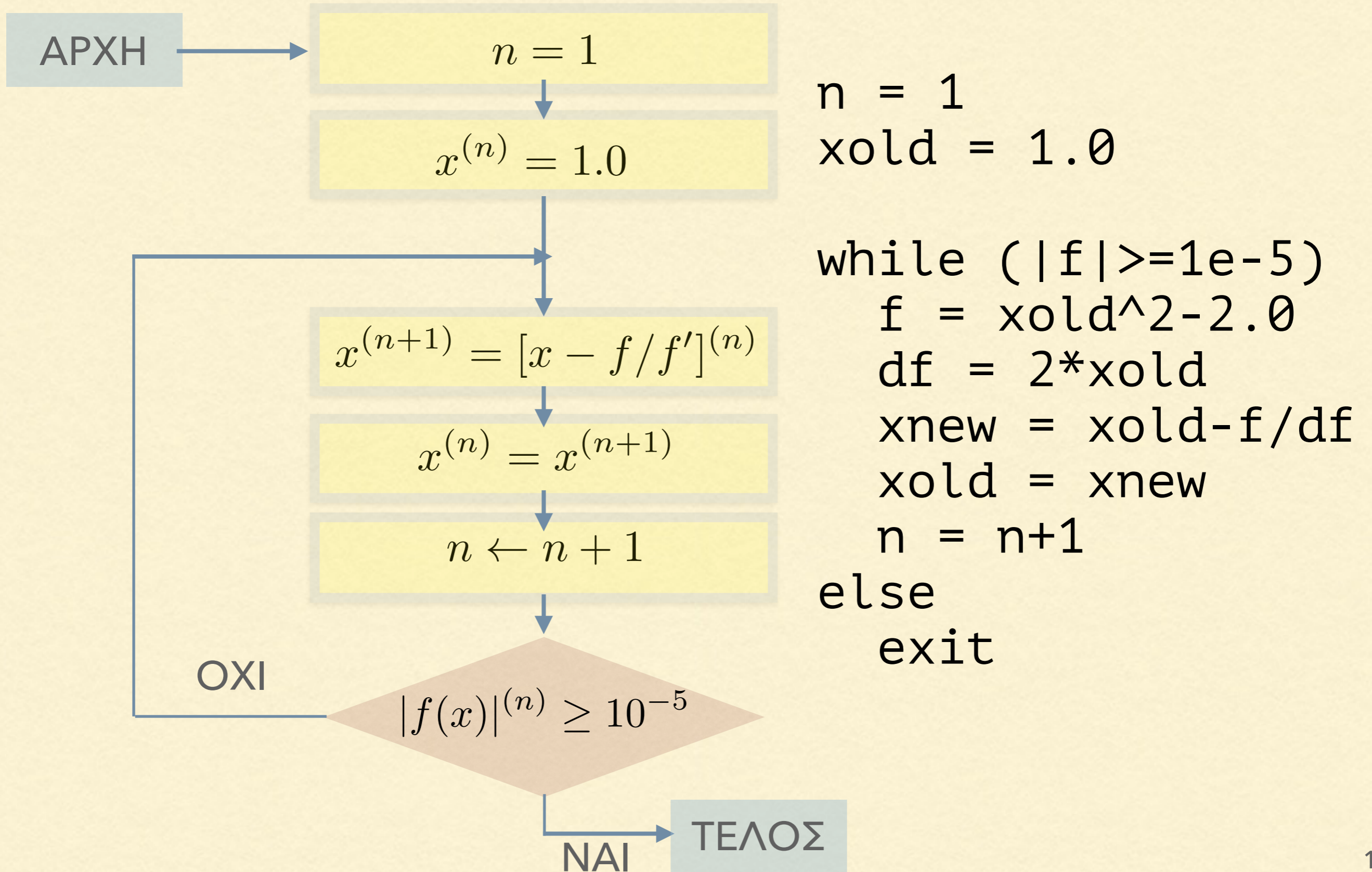
όσο ισχύει μια συνθήκη επαναλαμβάνεται ένα μέρος του κώδικα

# ΜΕΤΑΤΡΟΠΗ ΣΕ ΨΕΥΔΟΚΩΔΙΚΑ

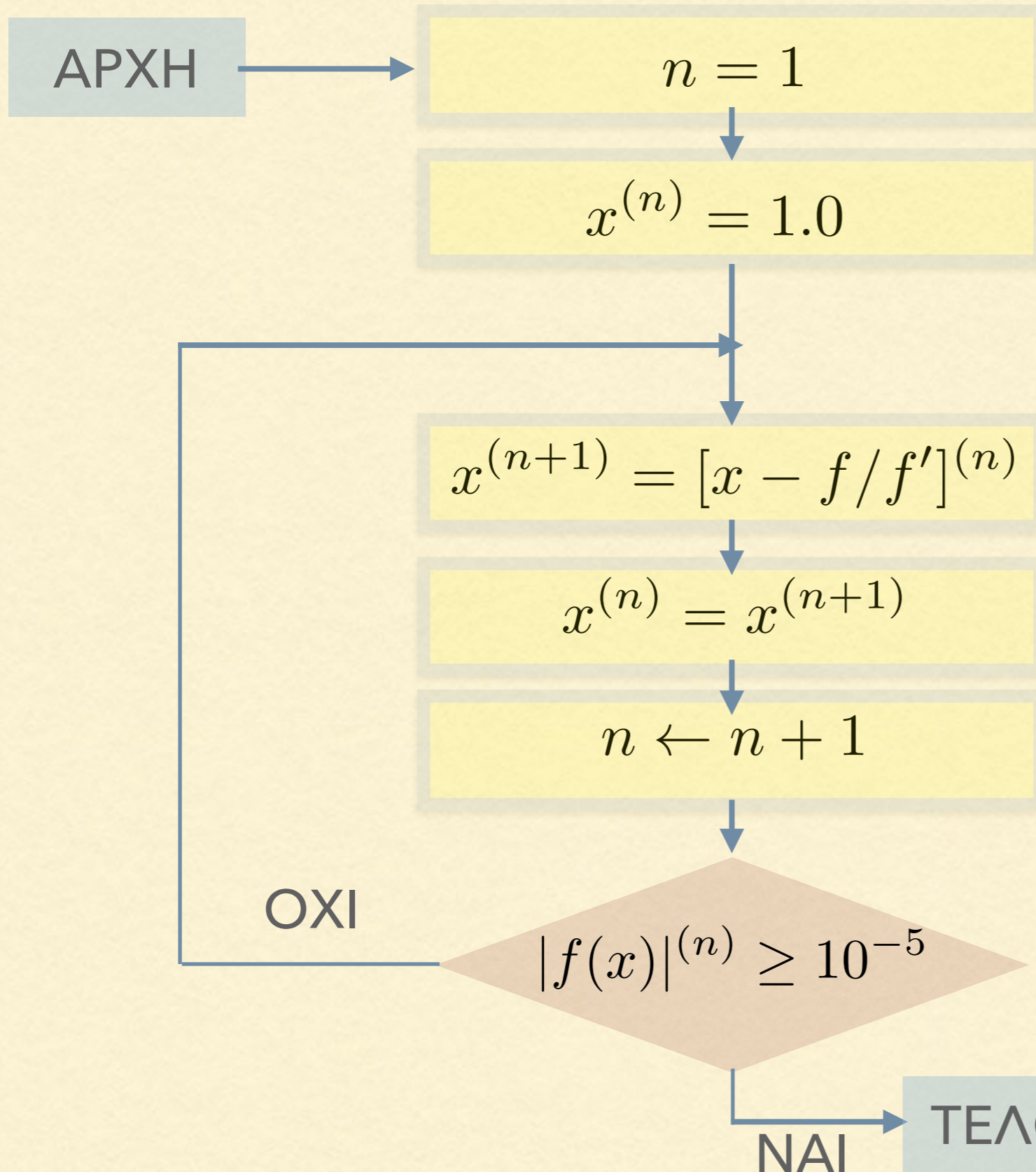




# ΜΕΤΑΤΡΟΠΗ ΣΕ ΨΕΥΔΟΚΩΔΙΚΑ



# ΜΕΤΑΤΡΟΠΗ ΣΕ ΨΕΥΔΟΚΩΔΙΚΑ



$n = 1$   
 $xold = 1.0$   
 $f = 1.0$

απαραίτητη  
αρχικοποίηση  
μεταβλητής

```
while (|f| >= 1e-5)  
    f = xold^2 - 2.0  
    df = 2 * xold  
    xnew = xold - f/df  
    xold = xnew  
    n = n + 1  
else  
    exit
```

# ΜΕΤΑΓΡΑΦΗ ΣΤΗ ΓΛΩΣΣΑ C

```
n = 1
xold = 1.0
f = 1.0

while (|f|>=1e-5)

    f = xold^2-2.0
    df = 2*xold
    xnew = xold-f/df
    xold = xnew
    n = n+1

else
    exit
```

```
n = 1;
xold = 1.0;
f = 1.0;

while( fabs(f) >= 1e-5)
{
    f = xold*xold-2.0;
    df = 2.0*xold;
    xnew = xold-f/df;
    xold = xnew;
    n = n+1;
}

return 0;
```

ἀγκίστρα

# ΜΕΤΑΓΡΑΦΗ ΣΤΗ ΓΛΩΣΣΑ C

```
n = 1
xold = 1.0
f = 1.0

while (|f|>=1e-5)

    f = xold^2-2.0
    df = 2*xold
    xnew = xold-f/df
    xold = xnew
    n = n+1

else
    exit
```

```
n = 1;
xold = 1.0;
f = 1.0;

while( fabs(f) >= 1e-5)
{
    f = xold*xold-2.0;
    df = 2.0*xold;
    xnew = xold-f/df;
    xold = xnew;
    n = n+1;
}

return 0;
```

συνάρτηση  
απόλυτης  
τιμής

# ΜΕΤΑΓΡΑΦΗ ΣΤΗ ΓΛΩΣΣΑ C

```
n = 1
xold = 1.0
f = 1.0

while (|f|>=1e-5)

    f = xold^2-2.0
    df = 2*xold
    xnew = xold-f/df
    xold = xnew
    n = n+1

else
    exit
```

```
n = 1;
xold = 1.0;
f = 1.0;

while( fabs(f) >= 1e-5)
{
    f = xold*xold-2.0;
    df = 2.0*xold;
    xnew = xold-f/df;
    xold = xnew;
    n = n+1;
}

return 0;
```

τερματισμός  
προγράμματος

# ΜΕΤΑΓΡΑΦΗ ΣΤΗ ΓΛΩΣΣΑ C

δήλωση  
τύπου  
μεταβλητών

```
n = 1
xold = 1.0
f = 1.0

while (|f|>=1e-5)

    f = xold^2-2.0
    df = 2*xold
    xnew = xold-f/df
    xold = xnew
    n = n+1

else
    exit
```

```
int n;
double xold, xnew, f, df;

n = 1;
xold = 1.0;
f = 1.0;

while( fabs(f) >= 1e-5)
{
    f = xold*xold-2.0;
    df = 2.0*xold;
    xnew = xold-f/df;
    xold = xnew;
    n = n+1;
}

return 0;
```

# ΠΡΟΣΘΗΚΗ ΕΝΤΟΛΗΣ ΕΚΤΥΠΩΣΗΣ

---

```
int n;  
double xold, xnew, f, df;  
  
n=1;  
xold=1.0;  
f=1.0e-5;  
  
while(fabs(f) >= 1.0e-5)  
{  
    f=xold*xold-2.0;  
    df=2.0*xold;  
    xnew=xold-f/df;  
    f=xnew*xnew-2.0;  
    xold=xnew;  
    n=n+1;  
}  
printf("%6.5e\n", xnew);  
return 0;
```

εντολή  
εκτύπωσης  
αποτελέσματος

# ΠΡΟΣΘΗΚΗ HEADERS

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

int main()
{ int n;
  double xold, xnew, f, df;

  n=1;
  xold=1.0;
  f=1.0e-5;

  while(fabs(f) >= 1.0e-5)
  {
    f=xold*xold-2.0;
    df=2.0*xold;
    xnew=xold-f/df;
    f=xnew*xnew-2.0;
    xold=xnew;
    n=n+1;
  }
  printf("%6.5e\n", xnew);
  return 0;
}
```

Απαραίτητη  
δήλωση  
συναρτήσεων  
της C



# ΠΡΟΣΘΗΚΗ ΚΥΡΙΑΣ ΣΥΝΑΡΤΗΣΗΣ

Απαραίτητη  
δήλωση  
κύριας  
συνάρτησης

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

int main()
{ int n;
  double xold, xnew, f, df;

  n=1;
  xold=1.0;
  f=1.0e-5;

  while(fabs(f) >= 1.0e-5)
  {
    f=xold*xold-2.0;
    df=2.0*xold;
    xnew=xold-f/df;
    f=xnew*xnew-2.0;
    xold=xnew;
    n=n+1;
  }
  printf("%6.5e\n", xnew);
  return 0;
}
```

# ΟΛΟΚΛΗΡΩΜΕΝΟ ΠΡΟΓΡΑΜΜΑ C

---

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

int main()
{ int n;
  double xold, xnew, f, df;

  n=1;
  xold=1.0;
  f=1.0e-5;

  while(fabs(f) >= 1.0e-5)
  {
    f=xold*xold-2.0;
    df=2.0*xold;
    xnew=xold-f/df;
    f=xnew*xnew-2.0;
    xold=xnew;
    n=n+1;
  }
  printf("%6.5e\n", xnew);
  return 0;
}
```

# ΜΕΤΑΓΛΩΤΤΙΣΗ ΠΡΟΓΡΑΜΜΑΤΟΣ

---

Ένας κώδικας για να “τρέξει” και να δώσει το αποτέλεσμα των πράξεων απαιτείται\* πρώτα να μεταγλωττιστεί σε **γλώσσα μηχανής** (machine language).

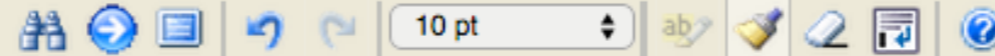
Η μεταγλώττιση γίνεται μέσω ειδικών προγραμμάτων μεταγλώττισης (**compilers**).

Για κάθε γλώσσα προγραμματισμού υπάρχουν διάφοροι compilers και για διάφορα λειτουργικά συστήματα.

\*υπάρχουν και γλώσσες προγραμματισμού που είναι **interpreted** (μεταγλώττιση/εκτέλεση μία εντολή τη φορά)

## run code

Language: C (gcc) Editor: EditArea



```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 { int n;
6   double xold, xnew, f, df;
7
8   n=1;
9   xold=1.0;
10  f=1.0e-5;
11
12  while(fabs(f) >= 1.0e-5)
13  {
14      f=xold*xold-2.0;
15      df=2.0*xold;
16      xnew=xold-f/df;
17      f=xnew*xnew-2.0;
18      xold=xnew;
19      n=n+1;
20  }
21  printf("%7.6e\n",xnew);
22  return 0;
23 }
```

Position: Ln 1, Ch 1 Total: Ln 23, Ch 300

Show compiler warnings [ + ] Compiler args [ + ] Show input

Compilation time: 0.12 sec, absolute running time: 0.14 sec, cpu time: 0 sec, memory peak: 3 Mb, absolute service time: 0.27 sec

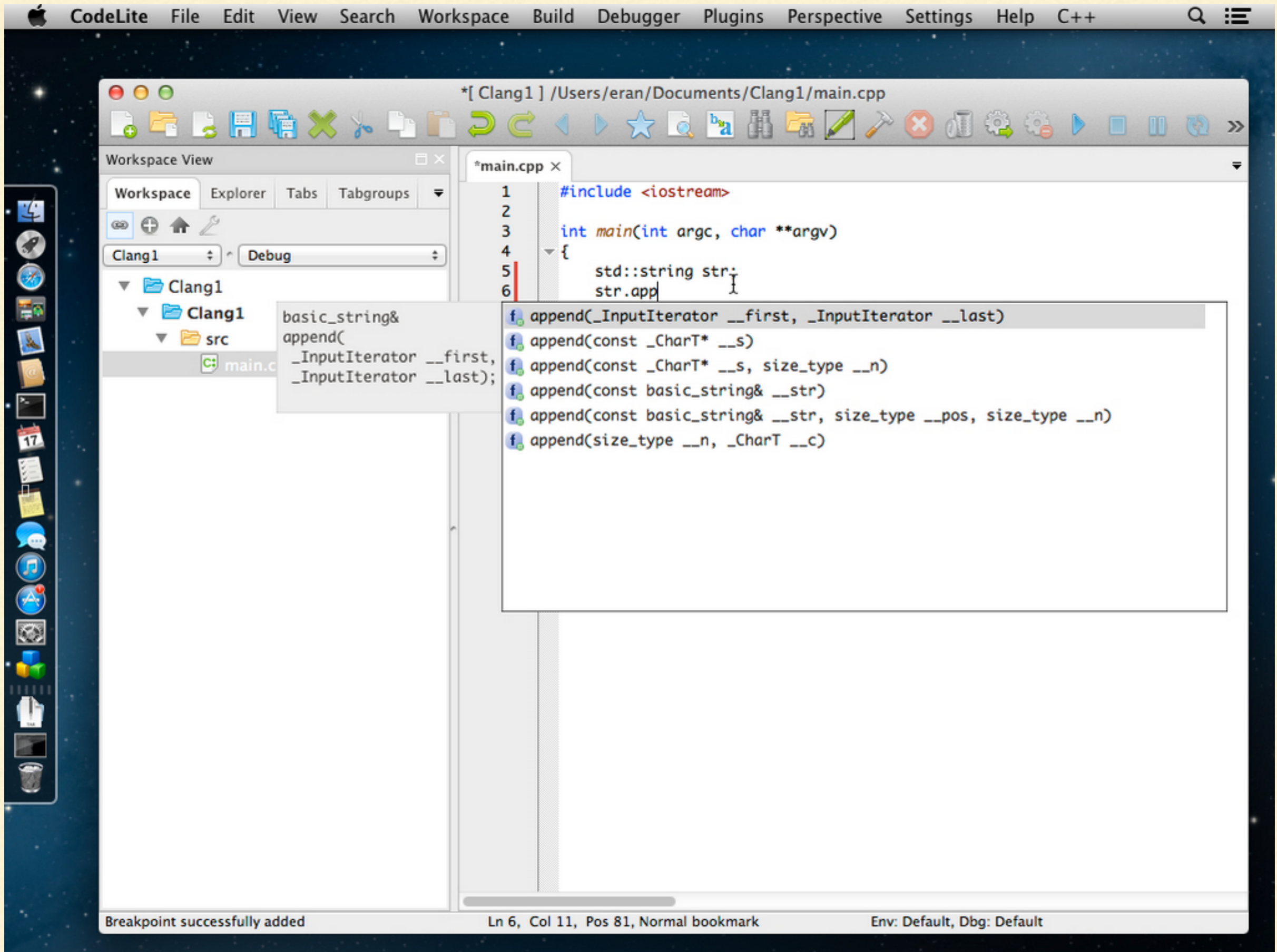
1.414216e+00

The image shows a web-based IDE for compiling C code. The interface includes a menu bar with options like 'System', 'File', 'Project', 'Edit', 'View', 'Tutorials', and 'Help'. Below the menu is a toolbar with icons for 'New Project', 'Compile', and 'Execute'. The main editor area displays the following C code:

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 { int n;
6   double xold, xnew, f, df;
7
8   n=1;
9   xold=1.0;
10  f=1.0e-5;
11
12  while(fabs(f) >= 1.0e-5)
13  {
14      f=xold*xold-2.0;
15      df=2.0*xold;
16      xnew=xold-f/df;
17      f=xnew*xnew-2.0;
18      xold=xnew;
19      n=n+1;
20  }
21  printf("%7.6e\n",xnew);
22  return 0;
23 }
```

At the bottom, a terminal window shows the execution results:

```
sh-4.3# gcc -o main *.c
sh-4.3# main
1.414216e+00
sh-4.3#
```



# [http://en.wikipedia.org/wiki/List\\_of\\_compilers#C\\_compilers](http://en.wikipedia.org/wiki/List_of_compilers#C_compilers)

Υπάρχουν πάνω από 50 compilers για τη γλώσσα C.

C compilers [\[edit\]](#)

*This list is incomplete; you can help by expanding it.*

Compiler	Author	Microsoft Windows	Unix-like	Other OSs	License type
AMPC	<a href="#">Axiomatic Solutions Sdn Bhd</a>	Yes	Yes	Yes	Proprietary
Aztec C	<a href="#">Manx Software Systems</a>	No	No	CP/M, CP/M-86, MS-DOS, Classic Mac OS	Proprietary
Amsterdam Compiler Kit	<a href="#">Andrew Tanenbaum</a> and <a href="#">Ceriël Jacobs</a>	No	Yes	Yes	BSD
CCS C Compiler	<a href="#">CCS, Inc.</a>	Yes	Yes	Yes	Proprietary
Ch	<a href="#">SoftIntegration, Inc</a>	Yes	Mac OS X, FreeBSD, Linux, Solaris, HP-UX, AIX, Qnx	Yes	Freeware
Clang	<a href="#">LLVM Project</a>	Yes	Yes	Yes	BSD
CodeWarrior	<a href="#">Metrowerks</a>	Yes	Yes	Classic Mac OS	Proprietary
CParser/libFirm	<a href="#">Matthias Braun</a> , <a href="#">Christoph Mallon</a> and <a href="#">Michael Beck</a>	Yes	Yes	Yes	GPL
DeSmet-C	<a href="#">C-Ware Corporation</a>	No	No	MS-DOS	GPL
Digital Mars	<a href="#">Digital Mars</a>	Yes	No	No	Proprietary
Dignus Systems/C	<a href="#">Dignus, L.L.C</a>	Yes (host)	Yes (host)	Z/Architecture	Proprietary
GCC C	<a href="#">GNU Project</a>	MinGW, Cygwin	Yes	IBM mainframe, AmigaOS, VAX/VMS, RTEMS	GPL
Hippo-C	<a href="#">Hippopotamus Software</a> , <a href="#">Haba Systems</a>	No	No	Classic Mac OS, Atari ST	Proprietary
IAR C/C++ Compilers	<a href="#">IAR Systems</a>	Yes	No	No	Proprietary
Interactive C	<a href="#">KISS Institute for Practical Robotics</a>	Yes	Unix, Mac OS X, Linux, IRIX, Solaris, SunOS	No	Freeware
LabWindows/CVI	<a href="#">National Instruments</a>	Yes	Yes	Yes	Proprietary
Lattice C	<a href="#">Lifeboat Associates</a>	No	Yes	MS-DOS, OS/2, Commodore, Amiga, Atari ST, Sinclair QL	Proprietary
lcc	<a href="#">Chris Fraser</a> and <a href="#">David Hanson</a>	Yes	Yes	Yes	Freeware (source code available for non-commercial use)
Mac C	<a href="#">Consular</a>	No	No	Classic Mac OS	Proprietary
Mark Williams C	<a href="#">Mark Williams Company</a>	Yes	Coherent	Yes	Proprietary
Micro-C Compiler (mcc)	<a href="#">Dunfield Development Services</a>	No	No	MS-DOS	Freeware (source code available)
Micro C Compiler (mcc)	<a href="#">Roshan Singh</a>	Yes	Yes	Yes	Freeware (source code available for non-commercial use)
MikroC Compiler	<a href="#">Mikroelektronika</a>	Yes	Yes	Yes	Proprietary
MPW C	<a href="#">Apple</a>	No	No	Classic Mac OS	Proprietary
Neatcc	<a href="#">Ali Gholami Rudi</a>	No	Yes	No	BSD
Nwcc	<a href="#">Nils Weller</a>	No	Yes	No	BSD
Open64	<a href="#">AMD SGI Google HP Intel Nvidia PathScale Tsinghua University and others</a>	No	Yes	Yes	GPL
Open Watcom	<a href="#">Sybase and SciTech Software</a>	Yes	Linux	OS/2, MS-DOS	Sybase Open Watcom Public License
Pelles C	<a href="#">Pelle Orinius</a>	Yes	No	No	Freeware
PGCC	<a href="#">The Portland Group</a>	Yes	Yes	Unknown	Proprietary
Portable C Compiler	<a href="#">Stephen C. Johnson</a> , <a href="#">Anders Magnusson</a> and others	Yes	Yes	Yes	BSD
Power C	<a href="#">Mix Software</a>	No	No	Yes	Proprietary
QuickC	<a href="#">Microsoft</a>	Yes	No	No	Proprietary
RCC (RCOR C Compiler)	<a href="#">Rodrigo Caetano Rocha</a> (rcoor)	Yes	Yes	No	GPL
Ritchie C Compiler (PDP-11)	<a href="#">Dennis Ritchie</a> and <a href="#">John Reiser</a> ; converted to cross-compiler by <a href="#">Doug Gwyn</a>	Yes	Yes	Yes	Freeware
SAS/C	<a href="#">SAS Institute</a>	Yes	Yes	Yes IBM mainframe, AmigaOS, 68K, 88K	Proprietary
SCORE C (tcc)	<a href="#">DDC-I</a>	Yes	Yes	Yes	Proprietary
Small-C	<a href="#">Ron Caine</a> , <a href="#">James E. Hendrix</a> , <a href="#">Byte magazine</a>	Yes	Yes	CP/M, MS-DOS	Public Domain
Smaller C	<a href="#">Alexey Frunze</a>	Yes	Linux, RetroBSD	MS-DOS	BSD
Small Device C Compiler	<a href="#">Sandeep Dutta</a> and others	Yes	Yes	Unknown	GPL
SubC	<a href="#">Nils M Holm</a>	MinGW	FreeBSD, NetBSD, Linux	MS-DOS	Public Domain
THINK C, Lightspeed C	<a href="#">THINK Technologies</a>	No	No	Classic Mac OS	Proprietary
Tiny C Compiler	<a href="#">Fabrice Bellard</a>	Yes	Yes	No	LGPL
(Borland) Turbo C	<a href="#">Embarcadero</a>	Yes	No	Yes	Proprietary - V 2.01 freely available
ups debugger (includes C interpreter)	<a href="#">Tom Hughes</a> , <a href="#">Ian Edwards</a> , and others	No	Yes	Solaris, SunOS	GPL
VBCC	<a href="#">Volker Barthelmann</a>	Yes	Yes	Yes	Freeware (source code available, modification not allowed)
Virtual-C IDE	<a href="#">Dieter Pawelczak</a>	Yes	Mac OS X	No	Freeware (for non-commercial use)
Visual C++ Express	<a href="#">Microsoft</a>	Yes	No	No	Freeware
Wind River (Diab) Compiler	<a href="#">Wind River Systems</a>	Yes	Yes	Yes	Proprietary
XL C	<a href="#">IBM</a>	No	AIX, Linux	No	Proprietary
MCP Compiler	<a href="#">Unisys</a>	No	No	MCP	Proprietary

# ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Οι 100 δημοφιλέστερες γλώσσες προγραμματισμού:

  
Ada (Gnat)

  
Algol-68

  
Angular JS

  
Assembly


  
AsciiDoc

  
Awk

  
Bash Shell

  
Befunge

  
Basic

  
Bootstrap

  
Brainf\*\*k

  
C

  
CSS-3

  
Clojure

  
Cobol

  
CoffeeScript


  
C99 Strict

  
C++

  
C++ 0x

  
C++ 11


  
C#

  
Dart


  
D Language

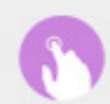
  
Embedded C

  
Erlang

  
Factor

  
Fantom

  
Falcon


  
Fortran-95

  
Forth

  
F#

  
Free Basic

  
Groovy

  
GO

  
Haxe

  
Haskell

  
HTML

  
ilasm

  
Intercal

  
ICON

  
Java

  
Java 8



# ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Java  
MySQL  
Java MySQL

JS

Jquery

julia  
julia

Ksh Shell

Lisp

LOLCODE

Lua

Matlab/Octave

Malgolbe

Markdown

MathML

Mozart-OZ

Nimrod

node JS  
Node.JS

Objective-C

OCaml

Pascal

PARI/GP

Pawn

Perl

Perl  
MySQL  
Perl MySQL

PHP

PHP  
MySQL  
PHP MySQL

Web View

Prolog

Python

Python-3

Python  
MySQL  
Python MySQL

Rexx

reStructure

Ruby

Rust

R Programming

Scala

Scheme

Smalltalk

Simula

smlnj  
SML/NJ

Script Basic

SQLite  
SQLite SQL

Tcl

Unlambda

VB.net  
VB.NET

Verilog

Whitespace

# ΕΦΑΡΜΟΓΕΣ ΤΩΝ ΓΛΩΣΣΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

http://lifehacker.com/learn-which-programming-language-to-choose-with-this-in-1669612111

## OBJECTIVE C

Objective C is an object oriented programming language, based on C, used by Apple developers.



Objective C was first developed in the 1990s.

Objective C files have an .m extension.

Objective C can:

Develop mobile apps for iOS

iOS



Develop applications for OS X



### Objective C Jobs



Tidbits of Wisdom

Though considered the correct language for iOS development, it cannot be used for other platforms such as Android.



## C++

C++ is an object-oriented programming language used to develop software, video games, and more.



C++ was first developed in 1983, as an enhancement to the C programming language

C++ files have a .c++ extension.

C++ can develop:

Apps for Windows and Linux



Video games



Mobile apps



### C++ Jobs



## JAVA®

This is a server-side interpreted compiled language, using a virtual machine. It is not JavaScript, and is not related to it.



Java was developed in 1995, and is one of the oldest programming languages on the web.

Java lets you:



Play online games



Upload photos



Take virtual tours



Use interactive maps

### Java Jobs



Tidbits of Wisdom



Users can disable Java on their machines.

Java is the basis of Android

Slow to change, so it's easier to keep up with

## PYTHON™

This is a server-side interpreted, open-source, non-compiled, scripting language. It can be used on its own, or as part of another framework, like django.

Python can:

Build websites



python™

Build desktop graphic user interfaces (GUIs)

Provide database access

Build software and games

### Python Jobs



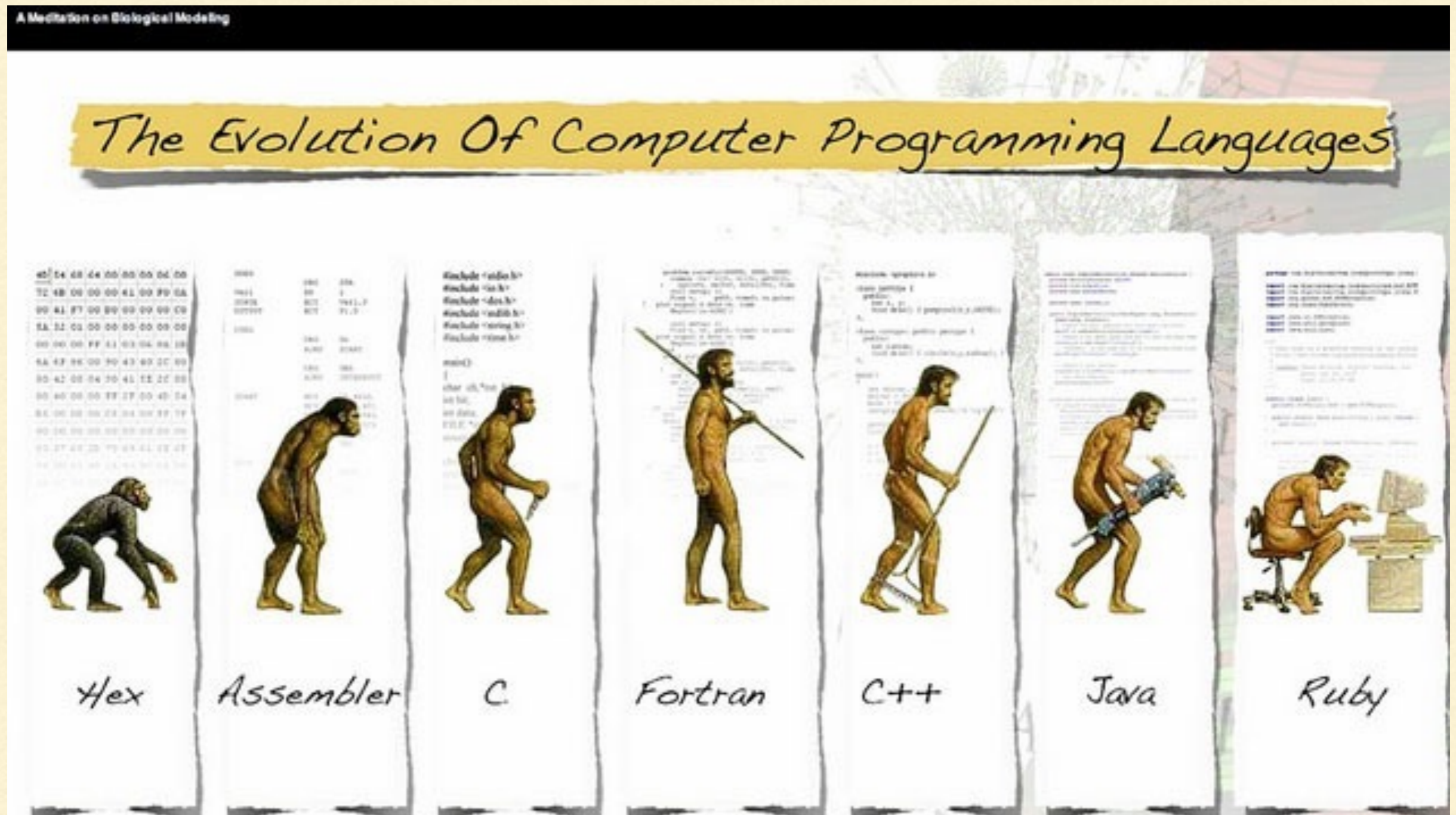
Tidbits of Wisdom

NASA's shuttle support contractor, United Space Alliance (USA) uses Python.

# ΕΞΕΛΙΞΗ ΤΩΝ ΓΛΩΣΣΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

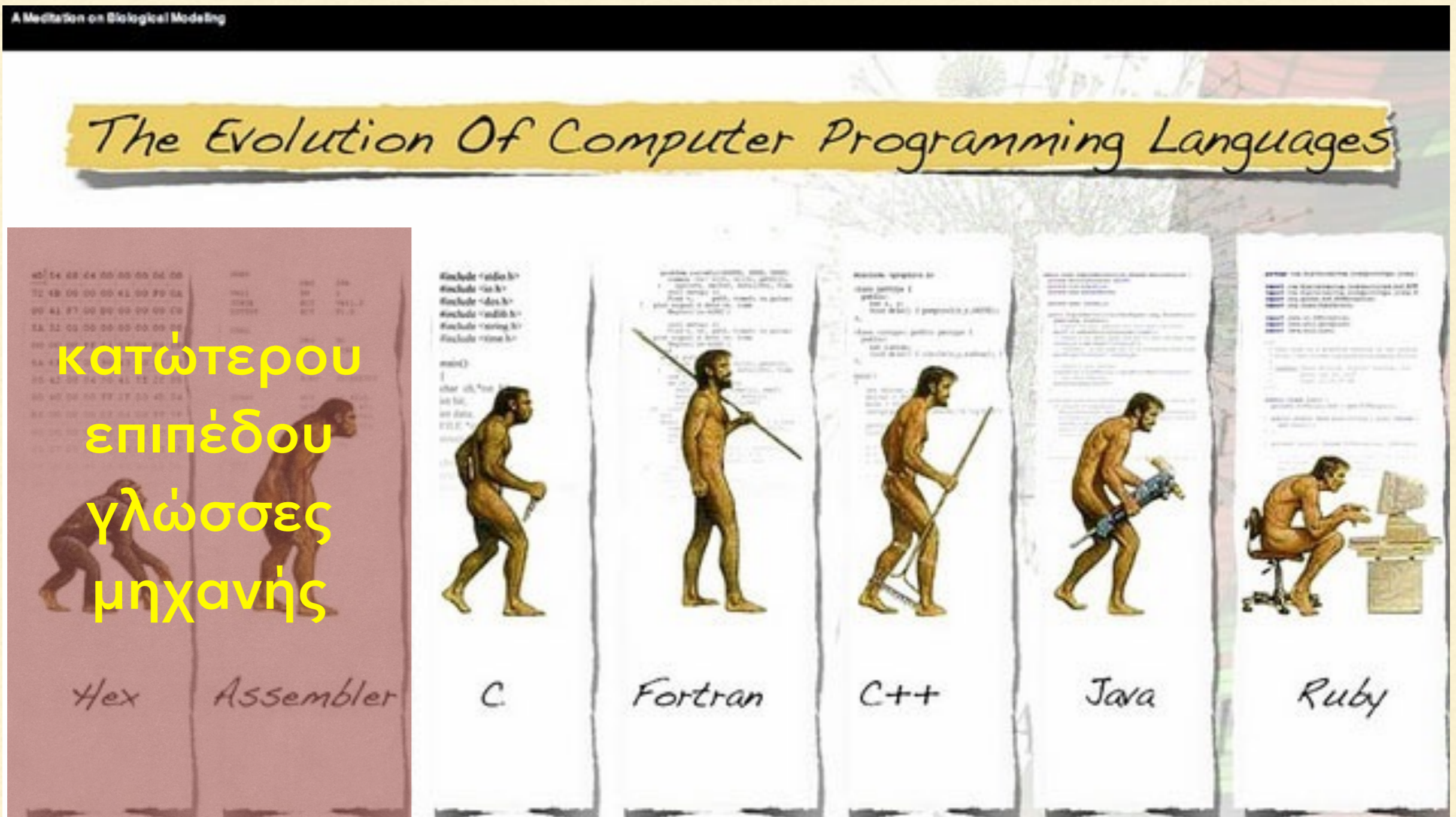
Η γλώσσα C είναι μια γλώσσα ανώτερου επιπέδου (**high-level language**) χωρίς την πολυπλοκότητα ή εξειδίκευση μεταγενέστερων γλωσσών.

<https://www.flickr.com/photos/dullhunk/4833512699/>



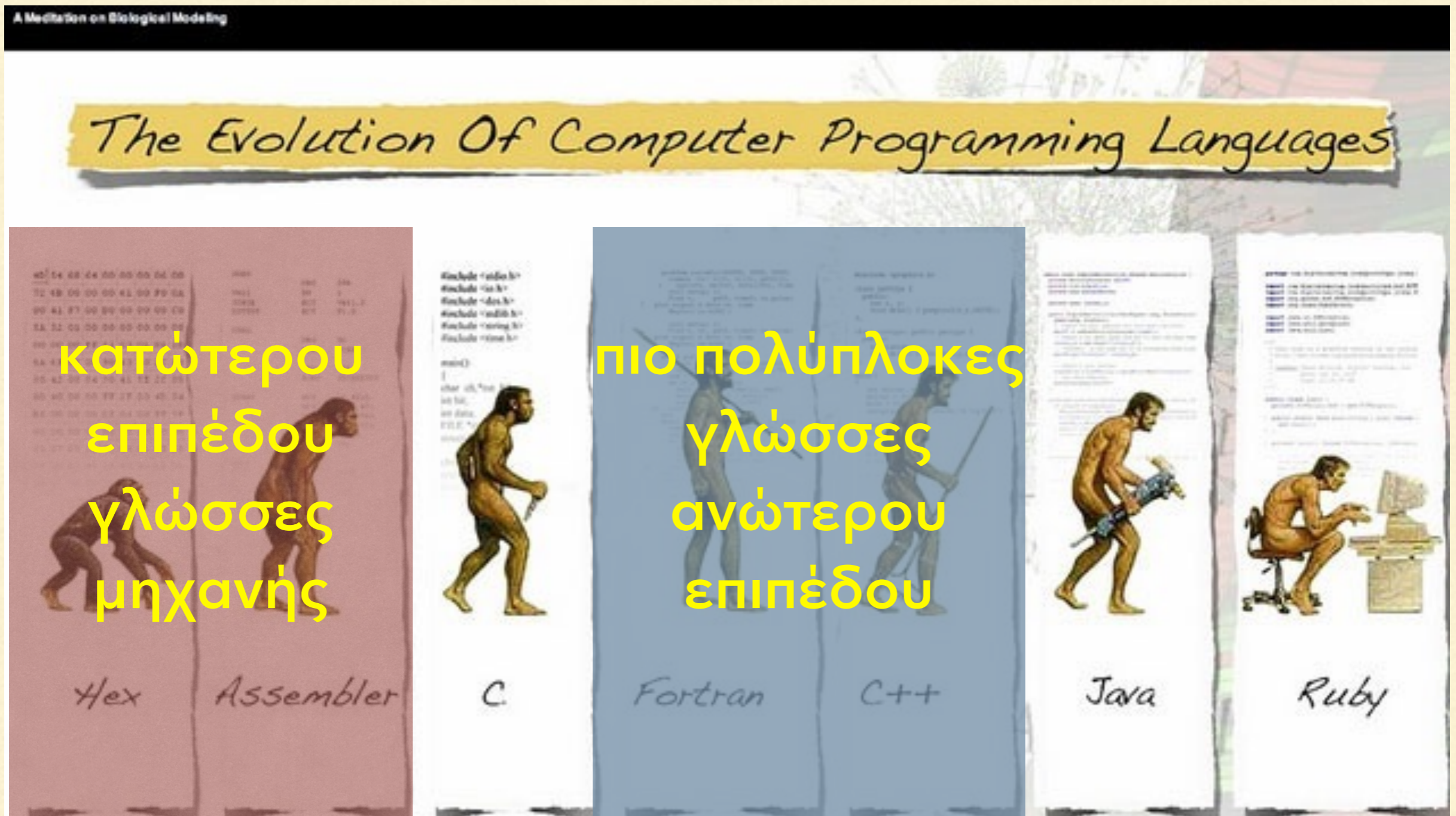
# ΕΞΕΛΙΞΗ ΤΩΝ ΓΛΩΣΣΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Η γλώσσα C είναι μια γλώσσα ανωτέρου επιπέδου (**high-level language**) χωρίς την πολυπλοκότητα ή εξειδίκευση μεταγενέστερων γλωσσών.



# ΕΞΕΛΙΞΗ ΤΩΝ ΓΛΩΣΣΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Η γλώσσα C είναι μια γλώσσα ανώτερου επιπέδου (**high-level language**) χωρίς την πολυπλοκότητα ή εξειδίκευση μεταγενέστερων γλωσσών.



# ΕΞΕΛΙΞΗ ΤΩΝ ΓΛΩΣΣΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Η γλώσσα C είναι μια γλώσσα ανώτερου επιπέδου (**high-level language**) χωρίς την πολυπλοκότητα ή εξειδίκευση μεταγενέστερων γλωσσών.

A Meditation on Biological Modeling

## The Evolution Of Computer Programming Languages

κατώτερου επιπέδου γλώσσες μηχανής

Hex Assembler

C

πιο πολύπλοκες εξειδικευμένες γλώσσες ανώτερου επιπέδου

Fortran C++

Java Ruby

# ΕΞΕΛΙΞΗ ΤΩΝ ΓΛΩΣΣΩΝ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Η γλώσσα C είναι μια γλώσσα ανώτερου επιπέδου (**high-level language**) χωρίς την πολυπλοκότητα ή εξειδίκευση μεταγενέστερων γλωσσών.

A Meditation on Biological Modeling

## The Evolution Of Computer Programming Languages

The diagram is divided into five vertical panels, each representing a different programming language and its corresponding evolutionary stage:

- Hex**: Represented by a monkey-like figure, labeled as the lowest level of programming language.
- Assembler**: Represented by a more upright monkey-like figure, labeled as the next level up.
- C**: Represented by a human-like figure, circled in blue, labeled as a high-level language without the complexity or specialization of later languages.
- Fortran** and **C++**: Represented by human-like figures holding spears, labeled as more complex high-level languages.
- Java** and **Ruby**: Represented by human-like figures sitting at a computer, labeled as specialized high-level languages.

Κατώτερου επιπέδου γλώσσες μηχανής

Πιο πολύπλοκες εξειδικευμένες γλώσσες ανώτερου επιπέδου

# ΓΙΑΤΙ C ΩΣ ΠΡΩΤΗ ΓΛΩΣΣΑ;

---

Η γλώσσα C είναι για τις άλλες γλώσσες ανώτερου επιπέδου ό,τι και τα **λατινικά** για τις νεότερες λατινογενείς γλώσσες (Ιταλικά, Γαλλικά, Ισπανικά).

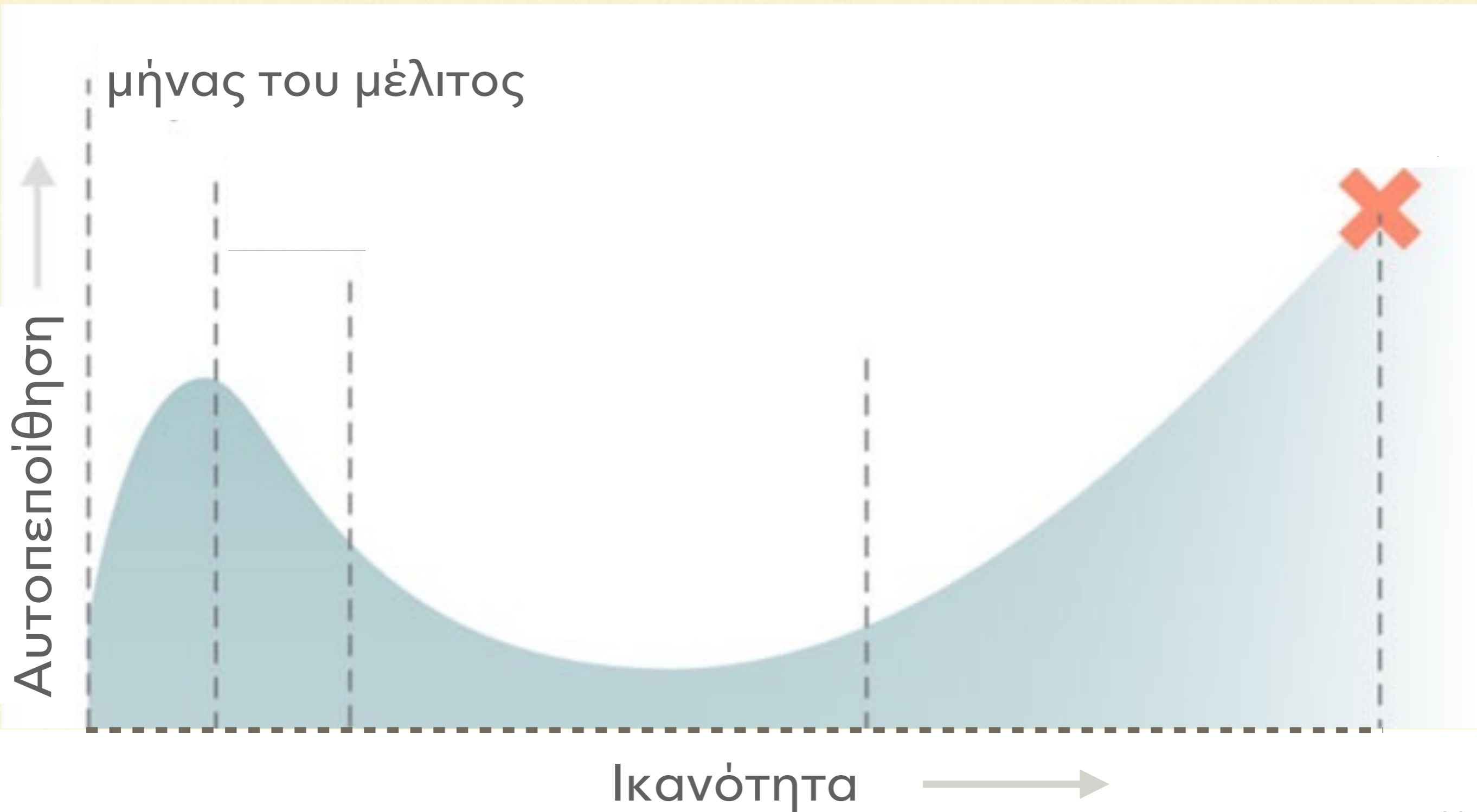
Το να ξεκινήσει κανείς να μαθαίνει προγραμματισμό με μια πιο εξειδικευμένη γλώσσα, όπως π.χ. με την Python, είναι σα να μαθαίνει να οδηγεί με αυτόματο αυτοκίνητο... το πιο πιθανό είναι να μη μάθει ποτέ αργότερα να οδηγεί με ταχύτητες...

Για τους παραπάνω λόγους, η γλώσσα C **ενδείκνυται ως μια πρώτη γλώσσα προγραμματισμού**, παρόλη την αυστηρότητά της.



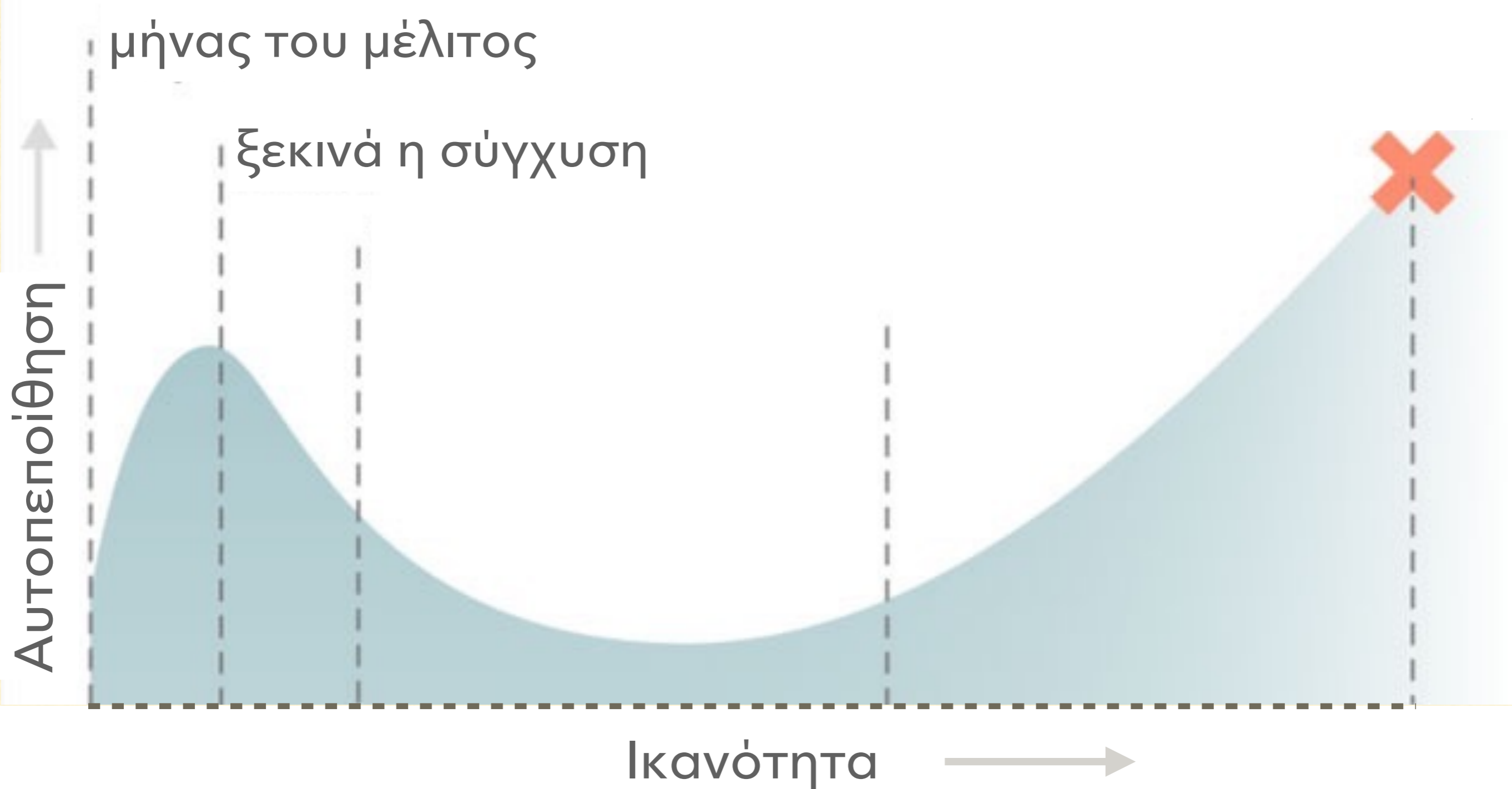
# Η ΕΞΕΛΙΞΗ ΕΝΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΤΗ

Η εξελικτική πορεία ενός προγραμματιστή είναι μη-γραμμική:



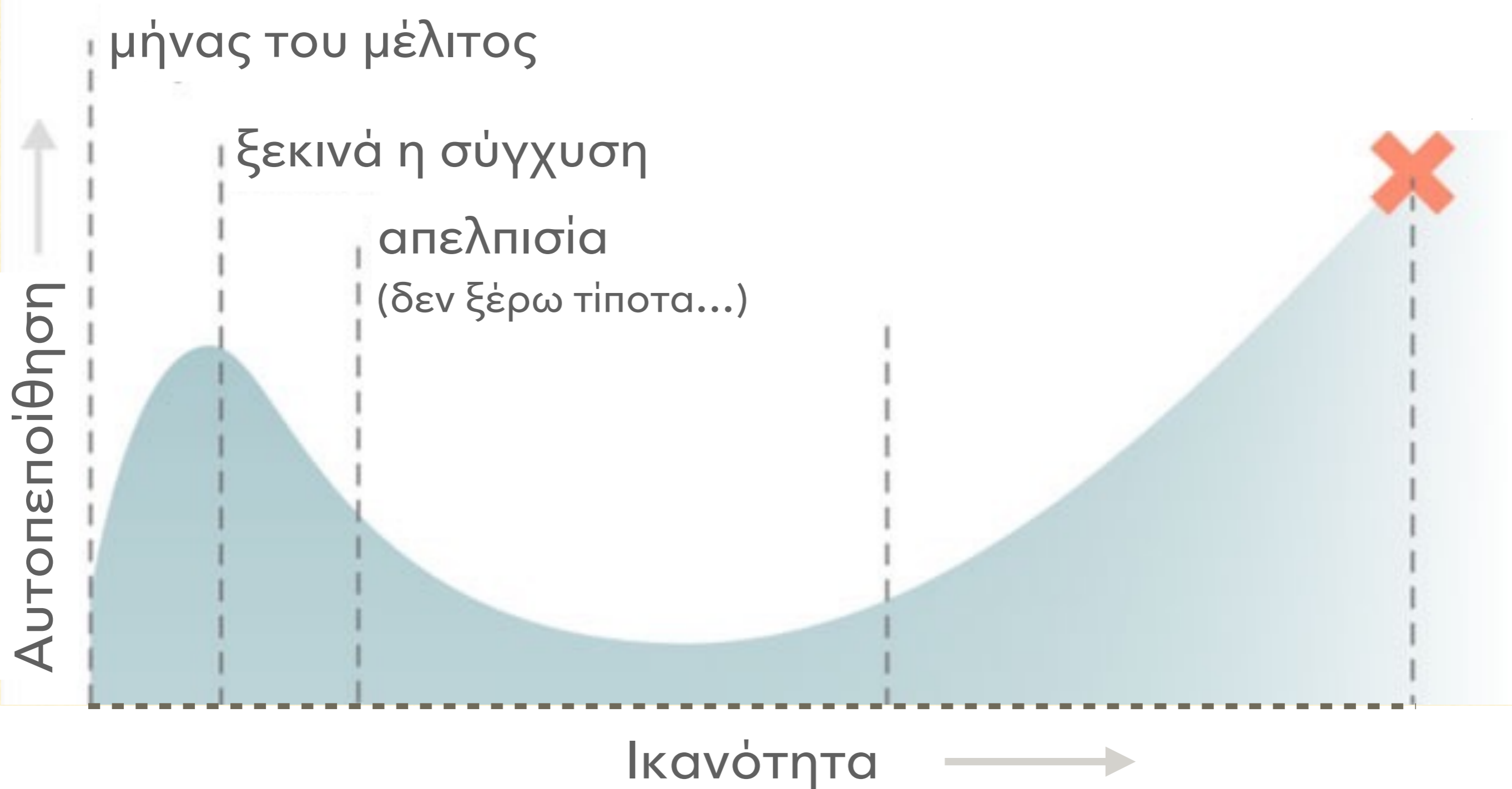
# Η ΕΞΕΛΙΞΗ ΕΝΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΤΗ

Η εξελικτική πορεία ενός προγραμματιστή είναι μη-γραμμική:



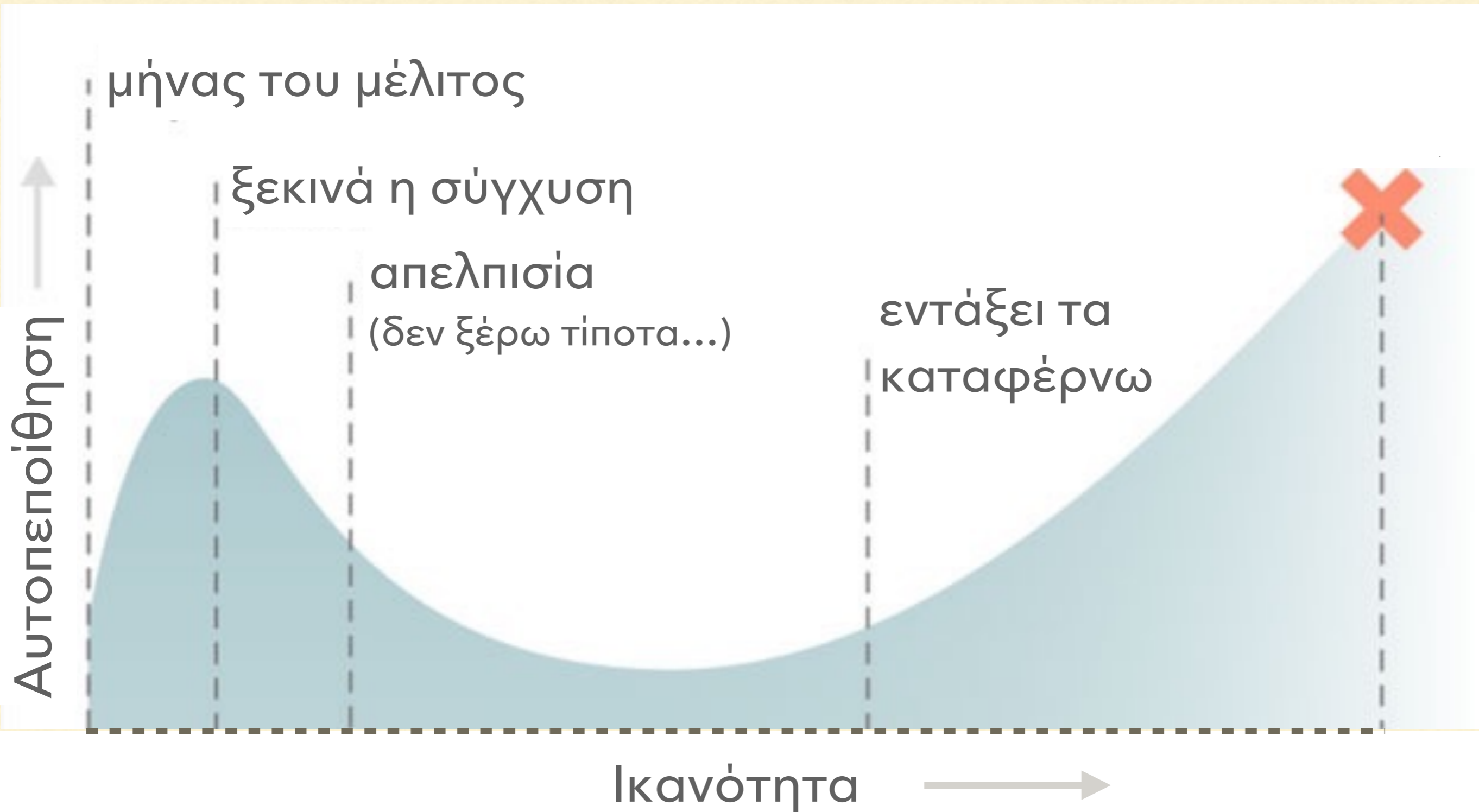
# Η ΕΞΕΛΙΞΗ ΕΝΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΤΗ

Η εξελικτική πορεία ενός προγραμματιστή είναι μη-γραμμική:



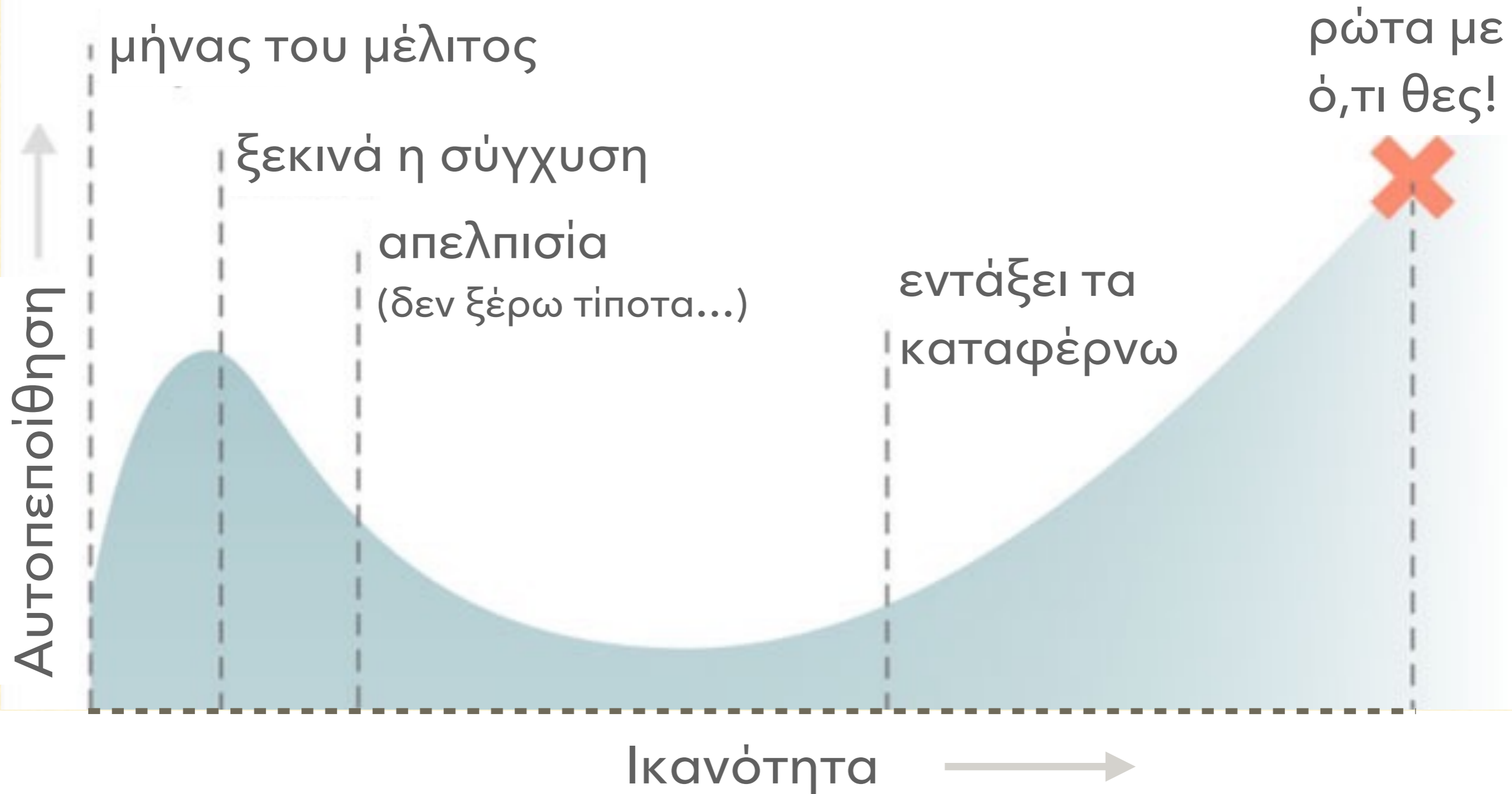
# Η ΕΞΕΛΙΞΗ ΕΝΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΤΗ

Η εξελικτική πορεία ενός προγραμματιστή είναι μη-γραμμική:



# Η ΕΞΕΛΙΞΗ ΕΝΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΤΗ

Η εξελικτική πορεία ενός προγραμματιστή είναι μη-γραμμική:



Δημιουργήθηκε από τον **Dennis Richie** στις αρχές της δεκαετίας του 1970 στα εργαστήρια Bell Labs της εταιρείας AT&T.

Ονομάστηκε C διότι κληρονόμησε πολλά χαρακτηριστικά από μια προηγούμενη γλώσσα που ονομαζόταν "B".

Ο κύριος σκοπός της δημιουργίας της γλώσσας C ήταν η χρήση της για την ανάπτυξη του λειτουργικού συστήματος **UNIX** (το οποίο μετεξελίχθηκε στο **Linux** για υπολογιστές και στο **Android** για κινητά/τάμπλετ).

# ANSI C

---

Το 1989 ο οργανισμός ANSI (American National Standard Institute) εισήγαγε το πρότυπο της **ANSI C**, το οποίο να **καθορίζει αυστηρά** μια σειρά από **κανόνες**, και **χαρακτηριστικά** της γλώσσας. Υποστηρίζεται από όλους τους μεταγλωττιστές C, άρα έχει **μεγάλη φορητότητα**.

Στο τέλος της δεκαετίας του '90 δημιουργήθηκε το νεότερο πρότυπο **C99**. Οι περισσότεροι μεταγλωττιστές το υποστηρίζουν εν μέρει, λίγοι όμως το υποστηρίζουν εξ' ολοκλήρου, γι' αυτό έχει **μικρή φορητότητα**.

- Το πρότυπο **ANSI C** παραμένει το επικρατέστερο.

# ΚΥΡΙΑ ΠΛΕΟΝΕΚΤΗΜΑΤΑ ΤΗΣ C

---

- ➔ Απλό συντακτικό και λίγες δεσμευμένες λέξεις
- ➔ Φορητότητα κώδικα (εκτελείται σε διαφορετικά λειτουργικά συστήματα)
- ➔ Υποστηρίζει υπορουτίνες και συναρτήσεις (δομημένος προγραμματισμός)
- ➔ Παρέχει έτοιμες συναρτήσεις



# ΚΥΡΙΑ ΜΕΙΟΝΕΚΤΗΜΑΤΑ ΤΗΣ C

---

- Έλλειψη αυστηρών περιορισμών (ο μεταγλωττιστής δεν θα αντιληφθεί όλα τα σφάλματα)
- Ένα πρόγραμμα C μπορεί να γραφεί με αρκετά πολύπλοκο τρόπο ακόμα και αν αποτελείται από λίγες γραμμές κώδικα
- Δεν είναι αντικειμενοστρεφής γλώσσα όπως η C++.

# ΚΥΚΛΟΣ ΔΗΜΙΟΥΡΓΙΑΣ ΕΝΟΣ ΚΩΔΙΚΑ C

---

- Συγγραφή κώδικα (**edit**)
- Μεταγλώττιση (**compile**)
- Σύνδεση με βιβλιοθήκες (Link)\*
- Εκτέλεση (**run/execute**)

\*Η σύνδεση με διάφορες βιβλιοθήκες γίνεται συνήθως αυτόματα, σε προχωρημένους κώδικες μπορεί να χρειασθεί να καθορίσουμε επιπλέον παραμέτρους

# ΕΝΑ ΠΡΩΤΟ ΑΠΛΟ ΠΡΟΓΡΑΜΜΑ

```
#include <stdio.h>

int main(void)
{
    printf("Hello, World!\n");

    return 0;
}
```

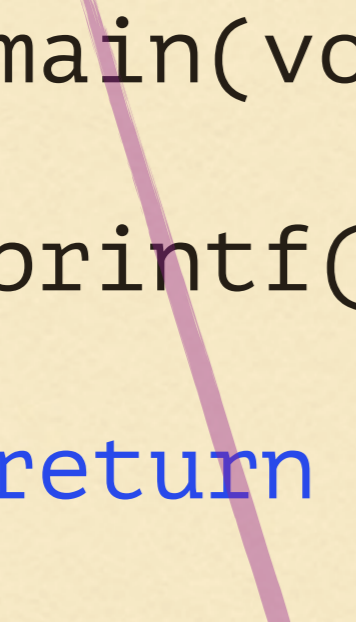
Αφού γράψουμε το πρόγραμμα, το σώζουμε ως ένα αρχείο με κατάληξη **.c** (π.χ. **hello.c**).

# Η ΟΔΗΓΙΑ #include

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

```
int main(void)
{
    printf("Hello, World!\n");

    return 0;
}
```



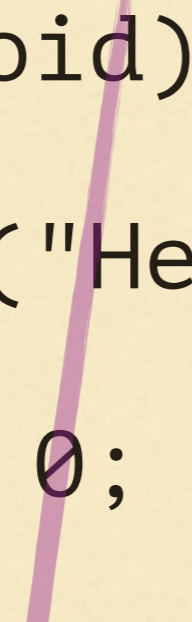
Η οδηγία **#include** ενσωματώνει στον κώδικα εντολές που βρίσκονται σε κάποιο άλλο αρχείο.

## ΤΟ ΑΡΧΕΙΟ #<stdio.h>

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

```
int main(void)
{
    printf("Hello, World!\n");

    return 0;
}
```



Το αρχείο **stdio.h** (standard input output) περιέχει τις δηλώσεις των συναρτήσεων για εμφάνιση δεδομένων στην οθόνη (**output**) και εισαγωγή δεδομένων από το πληκτρολόγιο (**input**).

# Η ΟΔΗΓΙΑ #include ΚΑΙ ΤΑ HEADER FILES

---

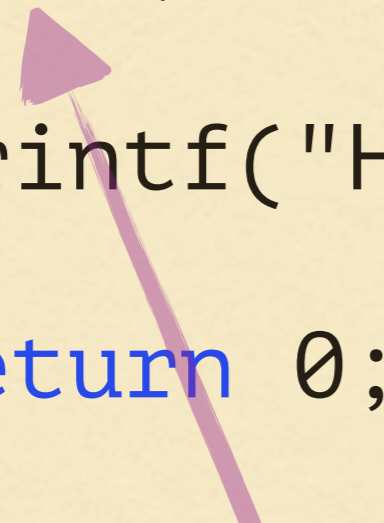
- ➔ Η οδηγία **#include** ξεκινάει πάντα με το σύμβολο # και δεν τελειώνει με ;
- ➔ Τα αρχεία με κατάληξη **.h** περιέχουν δηλώσεις συναρτήσεων και ονομάζονται **header files**.
- ➔ Όταν το όνομα του αρχείου είναι γραμμένο μέσα σε **< >**, τότε ο μεταγλωττιστής ψάχνει σε προεπιλεγμένους φακέλους του συστήματος.
- ➔ Μπορούμε να ενσωματώσουμε και **δικά μας αρχεία** έχοντας το όνομα γραμμένο μέσα σε **" "**. Ο μεταγλωττιστής ψάχνει στον ίδιο φάκελο που βρίσκεται το κυρίως πρόγραμμα.

# Η ΣΥΝΑΡΤΗΣΗ main()

```
#include <stdio.h>

int main(void)
{
    printf("Hello, World!\n");

    return 0;
}
```



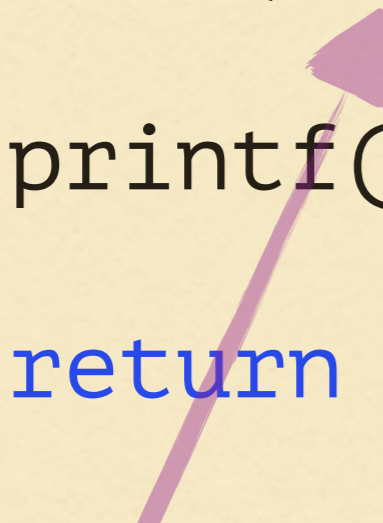
Η συνάρτηση **main(void)** υπάρχει υποχρεωτικά σε κάθε πρόγραμμα C και είναι η **κύρια συνάρτηση** του προγράμματος.

# Η ΣΥΝΑΡΤΗΣΗ main()

```
#include <stdio.h>

int main(void)
{
    printf("Hello, World!\n");

    return 0;
}
```



Το **void** (κενό) δηλώνει πως **δε χρησιμοποιούμε ορίσματα** στη συνάρτηση (μελλοντικά, θα χρησιμοποιήσουμε).



# Η ΣΥΝΑΡΤΗΣΗ `main()`


---

- ➔ Η εκτέλεση του προγράμματος **ξεκινάει από την πρώτη εντολή** που υπάρχει μέσα στη συνάρτηση `main()`.
- ➔ Η εκτέλεση του προγράμματος **τελειώνει με την τελευταία εντολή** που υπάρχει μέσα στη συνάρτηση `main()` ή και νωρίτερα με μία εντολή εξόδου.
- ➔ Όλες οι εντολές της συνάρτησης `main()` αλλά και κάθε άλλης συνάρτησης πρέπει να περιέχονται μέσα σε **άγκιστρα {...}**.
- ➔ Μπορούμε να γράψουμε **`main()`** αντί για **`main(void)`**, (δε συμφωνεί με το πρότυπο ANSI, αλλά είναι αποδεκτό).

# Η ΕΝΤΟΛΗ printf()

```
#include <stdio.h>

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



Η εντολή **printf()** είναι δηλωμένη στο `<stdio.h>` και **τυπώνει** ό,τι βρίσκεται μέσα σε `" "` στην οθόνη.

# Η ΕΝΤΟΛΗ printf()

```
#include <stdio.h>

int main(void)
{
    printf("Hello, World!\n");

    return 0;
}
```

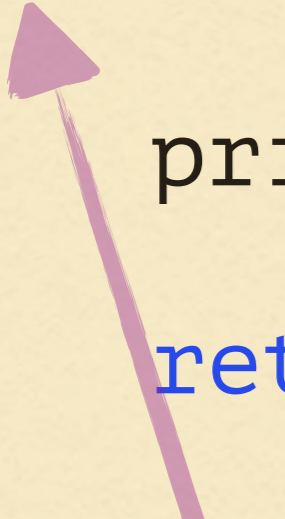
Ο χαρακτήρας `\n` δημιουργεί στο τέλος μια νέα γραμμή (**new line**), όπως το πλήκτρο Enter όταν πληκτρολογούμε.

# Ο ΤΥΠΟΣ ΤΗΣ MAIN

```
#include <stdio.h>

int main(void)
{
    printf("Hello, World!\n");

    return 0;
}
```



Στο συγκεκριμένο παράδειγμα, δηλώνουμε πως ο τύπος της `main()` είναι `int`, δηλαδή στο τέλος πρέπει να λάβει ως τιμή κάποιον ακέραιο αριθμό.

## Η ΕΝΤΟΛΗ return

```
#include <stdio.h>

int main(void)
{
    printf("Hello, World!\n");

    return 0;
}
```

Η εντολή **return** δίνει τιμή στον ακέραιο αριθμό της `main()` και τερματίζεται το πρόγραμμα. Συμβατικά, χρησιμοποιούμε το **0** για να δηλώσουμε τον **ομαλό τερματισμό** του προγράμματος.

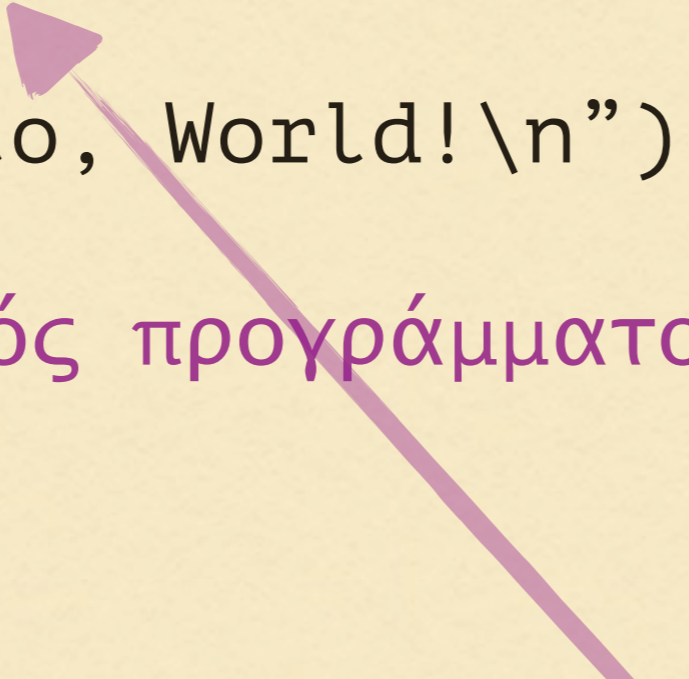
# ΣΧΟΛΙΑ

```
#include <stdio.h>

int main(void)
{
    /* Εκτύπωσε μια φράση */
    printf("Hello, World!\n");

    /* Τερματισμός προγράμματος */

    return 0;
}
```



Τα σχόλια περιλαμβάνονται μέσα σε `/* */`.

(κάποιοι compilers αναγνωρίζουν και το `//` στην αρχή μόνο κάθε γραμμής)

# Σημείωμα Αναφοράς

Copyright Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, **Νικόλαος Στεργιούλας**  
**«Προγραμματισμός Υπολογιστών & Υπολογιστική Φυσική»**. Έκδοση: 1.0.  
Θεσσαλονίκη 2014. Διαθέσιμο από τη δικτυακή διεύθυνση:  
[http://opencourses.auth.gr/eclass\\_courses](http://opencourses.auth.gr/eclass_courses).



# Σημείωμα Αδειοδότησης

Το παρόν υλικό διατίθεται με τους όρους της άδειας χρήσης Creative Commons Αναφορά - Παρόμοια Διανομή [1] ή μεταγενέστερη, Διεθνής Έκδοση. Εξαιρούνται τα αυτοτελή έργα τρίτων π.χ. φωτογραφίες, διαγράμματα κ.λ.π., τα οποία εμπεριέχονται σε αυτό και τα οποία αναφέρονται μαζί με τους όρους χρήσης τους στο «Σημείωμα Χρήσης Έργων Τρίτων».



Ο δικαιούχος μπορεί να παρέχει στον αδειοδόχο ξεχωριστή άδεια να χρησιμοποιεί το έργο για εμπορική χρήση, εφόσον αυτό του ζητηθεί.

[1] <http://creativecommons.org/licenses/by-sa/4.0/>

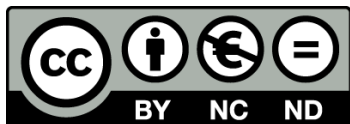






# Τέλος ενότητας

Επεξεργασία: Νικόλαος Τρυφωνίδης  
Θεσσαλονίκη, 20/09/2015



Ευρωπαϊκή Ένωση  
Ευρωπαϊκό Κοινωνικό Ταμείο



ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ  
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης



ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

**ΣΗΜΕΙΩΜΑΤΑ**

# Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.

