# Κεφάλαιο 7

## Προγραμματισμός υπολογιστή

- Έχετε χρησιμοποιήσει προγράμματα;

- Έχετε φτιάξει προγράμματα;

- Γνωρίζετε προγραμματιστές που έγιναν διάσημοι μέσω της δουλειάς τους;


# Κεφ. 7 - Προγραμματισμός υπολογιστή

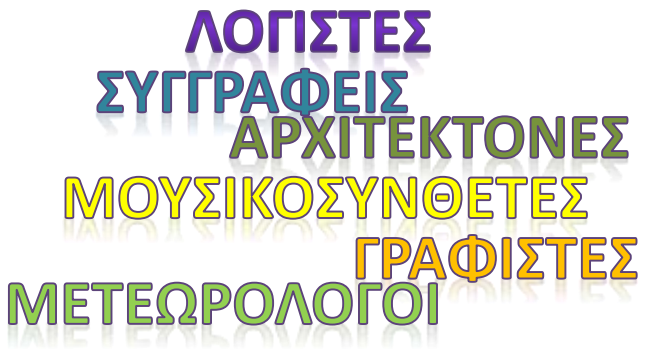
## 7.1 Η προγραμματιζόμενη μηχανή

- **Σε τι διαφέρει ο υπολογιστής από τις άλλες ηλεκτρονικές συσκευές;**

*Ο Η/Υ μπορεί να χρησιμοποιηθεί σε πάρα πολλές εργασίες, σε αντίθεση με τις υπόλοιπες συσκευές, οι οποίες είναι σχεδιασμένες να επιτελούν μία συγκεκριμένη λειτουργία.*

- **Πού οφείλεται η δυνατότητα του Η/Υ, να μπορεί να επιτελεί τόσες διαφορετικές εργασίες;**

*Αυτό που επιτρέπει στο υλικό να προσαρμόζεται σε τόσες διαφορετικές απαιτήσεις είναι το λογισμικό.*



# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.2 Οι χρήστες


***Εκείνοι που χρησιμοποιούν τον ΗΥ στη δουλειά τους.***

- **Πολιτικοί μηχανικοί**
- **Δημοσιογράφοι**
- **Ιατροί**
- **Τουριστικοί πράκτορες**
- **Εκπαιδευτικοί**
- **…**

***Εκείνοι που η δουλειά τους είναι ο ΗΥ.***

- **Προγραμματιστές**
- **Σχεδιαστές υλικού**
- **Τεχνικοί Η/Υ**
- **…**

# Κεφ. 7 - Προγραμματισμός υπολογιστή

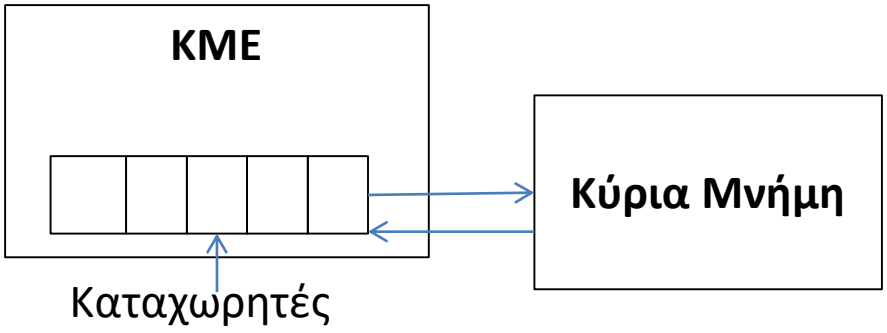
## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.1 Το πρόγραμμα


**Πρόγραμμα** είναι μια συγκεκριμένη ακολουθία εντολών τις οποίες πρέπει να εκτελέσει ένας υπολογιστής για να παράγει το επιθυμητό για τον χρήστη αποτέλεσμα.

*Οι σύγχρονοι ΗΥ είναι βασισμένοι στις αρχές που εισήγαγε ο **Von Neumann** τη δεκαετία του **1940**.*

*Μαθηματικός και πολυμαθής επιστήμονας, Αμερικανοουγγαρικής καταγωγής.*




*Η **ΚΜΕ** μπορεί να φορτώσει στοιχεία από την **Κύρια μνήμη** στους **καταχωρητές**, να εκτελέσει πράξεις πάνω τους και να επιστρέψει το αποτέλεσμα πίσω στη μνήμη.*

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

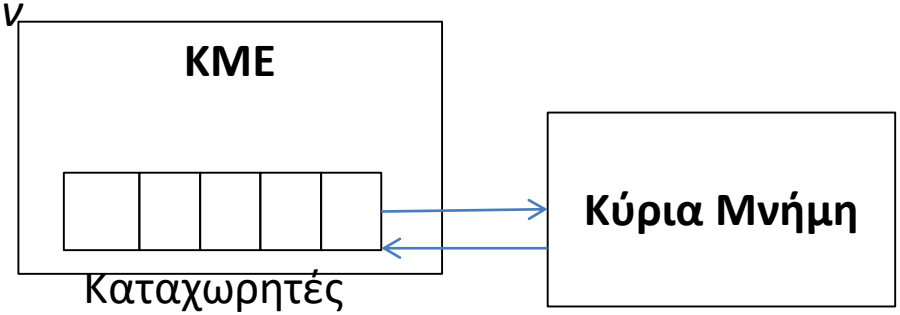
### 7.3.2 Γλώσσα μηχανής (1η γενιά)

Ένα πρόγραμμα σε **γλώσσα μηχανής** διατυπώνεται ως ακολουθία 0 και 1.

- **Γιατί επιλέχθηκε το δυαδικό σύστημα;**
- **Ποια άλλα αριθμητικά συστήματα γνωρίζεις;**

**Κατηγορίες εντολών** μιας ΚΜΕ:

- *μεταφορά δεδομένων μεταξύ μνήμης και καταχωρητών*
- *μεταφορά δεδομένων μεταξύ καταχωρητών*
- *αριθμητικές και λογικές πράξεις*
- *έλεγχος της ροής εκτέλεσης των εντολών*
- *διάφορες βοηθητικές εντολές*



# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.2 Γλώσσα μηχανής (1η γενιά)

| Κωδικός λειτουργίας | Τελεστέος |
|---|---|
| 11100110 | 10110011 |
| *Πρόσθεσε στον καταχωρητή accumulator…* | …τον αριθμό 179 |

**Το να προγραμματίζεις σε γλώσσα μηχανής αποτελούσε μια διαδικασία:**

- **δύσκολη**
- **χρονοβόρα**
- **αντιπαραγωγική**

**Διότι:**

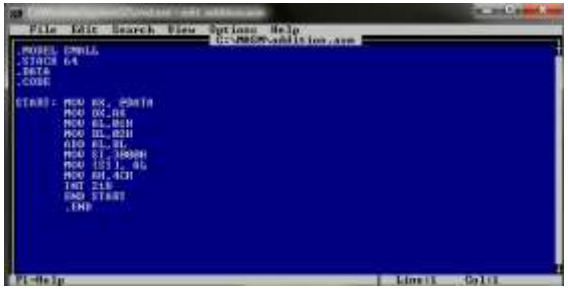
- ⚠ **απαιτούσε απομνημόνευση εντολών σε 0 & 1**
- ⚠ **ήταν δύσκολο να εντοπιστούν λάθη**

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.3 Συμβολική γλώσσα (2η γενιά)

Στη **συμβολική γλώσσα (assembly)** σε κάθε εντολή αντιστοιχίζεται μια ευκολομνημόνευτη λέξη. Επιπλέον, δίνεται η δυνατότητα να χρησιμοποιούνται ονόματα που αντιπροσωπεύουν αριθμούς ή θέσεις μνήμης.



```
...
ADD gradeA
ADD gradeB
...
```

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.4 Υψηλού επιπέδου (3η γενιά)

**Οι συμβολικές γλώσσες διευκόλυναν τη γραφή προγραμμάτων, αλλά προέκυπταν προγράμματα που ήταν:**

- μακροσκελή
- εξαρτημένα από την αρχιτεκτονική του Η/Υ *(αδυναμία μεταφερσιμότητας)*

**Μεταφερσιμότητα** ονομάζεται το χαρακτηριστικό ενός προγράμματος να μπορεί να εκτελεστεί σε άλλον τύπο ΗΥ με ελάχιστες αλλαγές.

Οι **γλώσσες υψηλού επιπέδου (high level languages)** έλυναν τα παραπάνω προβλήματα, προσφέροντας παράλληλα ένα σύνολο εντολών φιλικό προς το χρήστη.


```
…
aver = ((grA + grB)/2 + grWr)/2
printf  (“Ο τελικός ΜΟ στο μάθημα είναι:”, aver)
…
```

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.5 Ένα συγκριτικό παράδειγμα

| | |
|---|---|
| **Γλώσσα μηχανής** | 0000001001011010<br>0000101001011110<br>0000011011011110 |
| **Συμβολική γλώσσα** | LDA B<br>ADD C<br>STA A |
| **Γλώσσα υψηλού επιπέδου** | A = B + C |

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.6 Ιστορία γλωσσών υψηλού επιπέδου

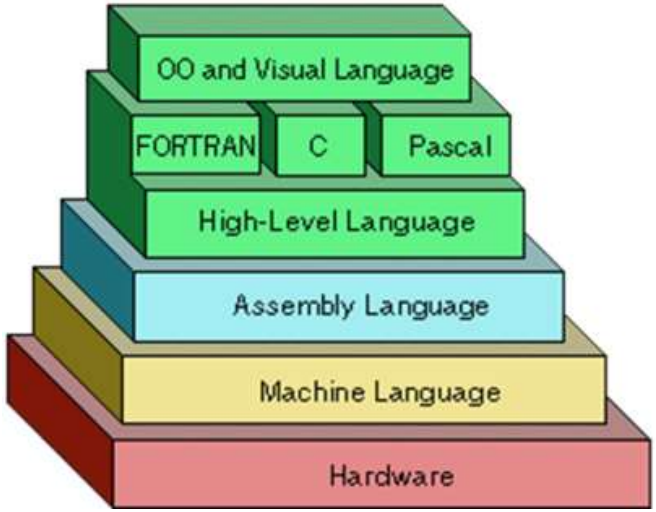
| | | | |
|---|---|---|---|
| **FORTRAN** | **FOR**mula **TRAN**slation | ‘54 | Κατάλληλη για υλοποίηση μαθηματικών μοντέλων |
| **ALGOL** | **ALGO**rithmic **L**anguage | ’63 | Εισήγαγε αξιόλογες ιδέες στον προγραμματισμό, όπως ο *δομημένος* προγραμματισμός |
| **COBOL** | **CO**mmon **B**usiness **O**riented **L**anguage | ’59 | Χρησιμοποιήθηκε ευρέως για εμπορικές εφαρμογές |
| **BASIC** | **B**eginner’s **A**ll-purpose **S**ymbolic **I**nstruction **S**et | ’60 | Σχεδιάστηκε για τη διδασκαλία προγραμματισμού σε φοιτητές. Αναπτύχθηκαν αργότερα πολλές επεκτάσεις |
| **SIMULA** | **SIMULA**tion | ’60 | Βασική εφαρμογή στις προσομοιώσεις |
| **LISP** | **LIS**t **P**rocessing | ‘50 | Βασική εφαρμογή στην τεχνητή νοημοσύνη |
| **LOGO** | Λόγος | ‘67 | Σχεδιάστηκε για τη διερεύνηση προγραμματιστικών εννοιών από μαθητές μικρής ακόμα ηλικίας |
| **PASCAL** | Προς τιμή του Γάλλου μαθηματικού και φιλόσοφου Blaise Pascal | ’71 | Μικρή και αποδοτική γλώσσα που υιοθέτησε αξιόλογες πρακτικές προγραμματισμού. Με πολλές επεκτάσεις στην πορεία |
| **C** | Εξέλιξη της **B**, που ήταν εξέλιξη της **A** | ‘72 | Σχεδιάστηκε για τη συγγραφή λειτουργικών συστημάτων, αλλά χρησιμοποιήθηκε ευρύτατα για την κατασκευή λογισμικού είτε συστήματος είτε εφαρμογών |
| **PROLOG** | **PRO**gramming in **LOGic** | ‘73 | Βασική εφαρμογή στην τεχνητή νοημοσύνη |
| **C++** | Εξέλιξη της **C+** | ‘79 | Υιοθετεί τον αντικειμενοστραφή προγραμματισμό |
| **JAVA** | **Ιάβα** *(νησί στην Ινδονησία που παράγει τους κόκκους του καφέ που έπιναν οι developers της γλώσσας)* | ‘95 | Κυριαρχεί στις διαδικτυακές εφαρμογές |
| **SQL** | **S**tructured **Q**uery **L**anguage | ’79 | Καθιερώθηκε ως γλώσσα επικοινωνίας με τις σχεσιακές βάσεις δεδομένων |

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.6 Ιστορία γλωσσών υψηλού επιπέδου

**Γλώσσες 4ης γενιάς** ονομάζονται οι γλώσσες που είναι πιο κοντά στη φυσική γλώσσα απ' ότι οι συνήθεις γλώσσες υψηλού επιπέδου.




**Π.χ. SQL**

```
select  lastName, FirstName
from classC
where gender="Male"  and year=1996"
```

Οι γλώσσες υψηλού επιπέδου επέφεραν τη ραγδαία ανάπτυξη λογισμικού, διότι

- η εκμάθηση των γλωσσών πρ/σμού ήταν ευκολότερη
- η κατασκευή λογισμικού ήταν ευκολότερη

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

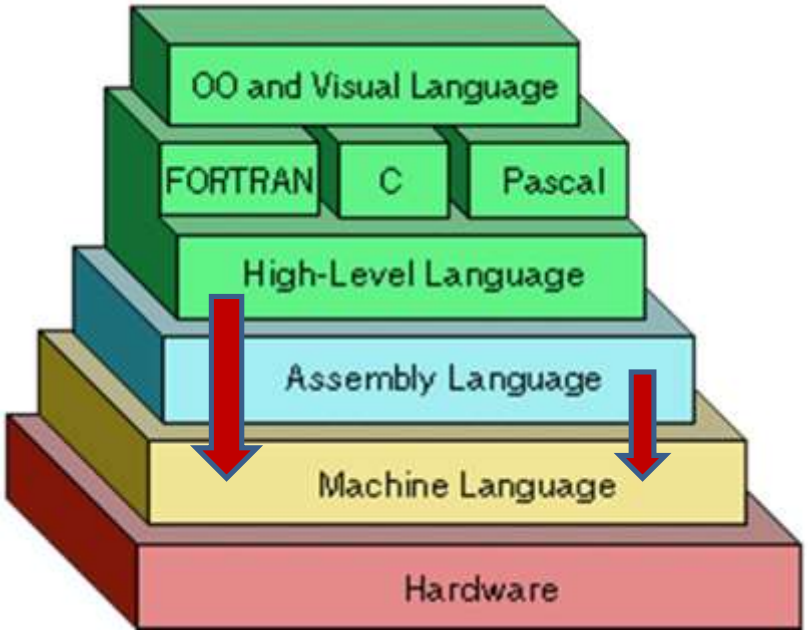
### 7.3.7 Εξάρτηση των γλωσσών από τον σκοπό

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.8 Μεταφραστές

Η ΚΜΕ εκτελεί εντολές γραμμένες σε γλώσσα μηχανής. Συνεπώς, αν το πρόγραμμα γραφεί σε συμβολική γλώσσα ή σε γλώσσα υψηλού επιπέδου, απαιτείται η μετατροπή του σε γλώσσα μηχανής.

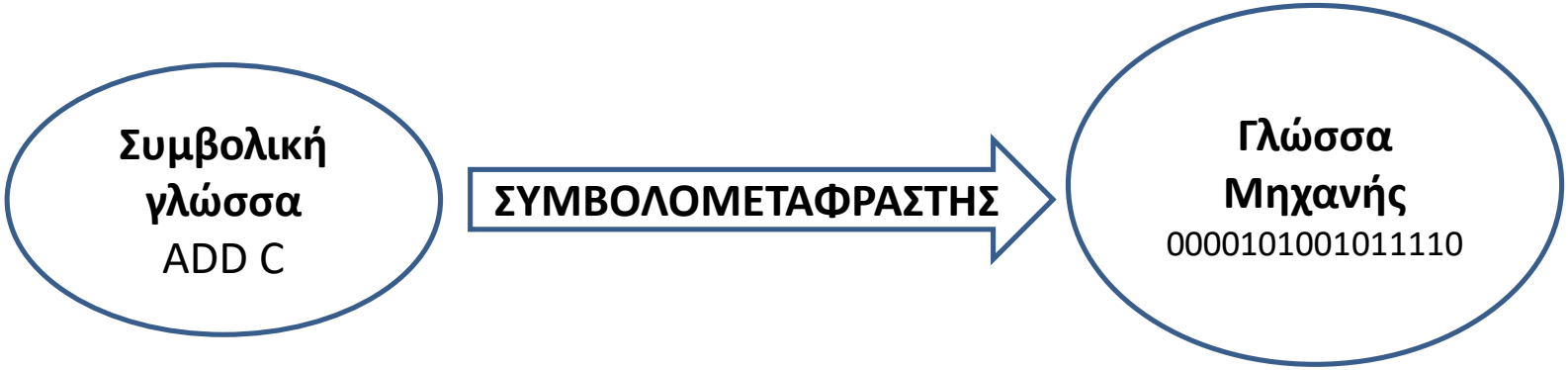



# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.8 Μεταφραστές

Ο **συμβολομεταφραστής (assembler)** είναι ένα πρόγραμμα που μετατρέπει τις εντολές ενός προγράμματος που είναι γραμμένο σε συμβολική γλώσσα σε γλώσσα μηχανής.

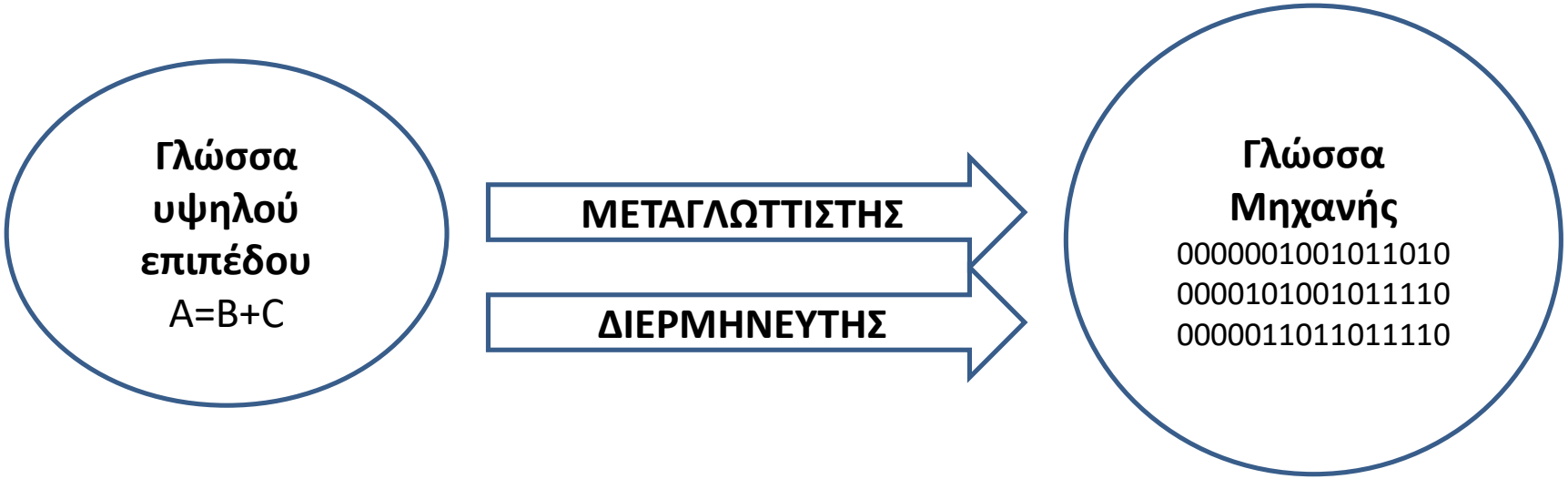


# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.8 Μεταφραστές

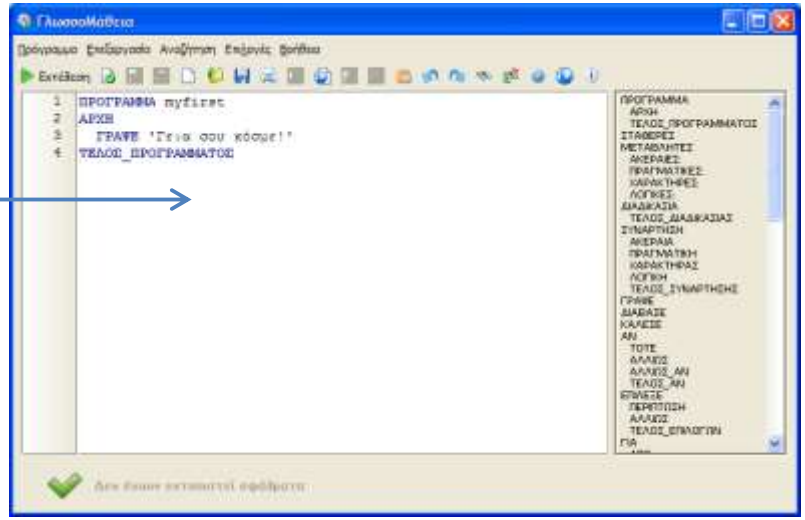

Για να εκτελεστεί ένα πρόγραμμα γραμμένο σε γλώσσα υψηλού επιπέδου, απαιτείται είτε ένας **μεταγλωττιστής** είτε ένας **διερμηνευτής**.



# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.8 Μεταφραστές - *Μεταγλωττιστής*




- **Τι χρειάζομαι για να γράψω ένα πρόγραμμα;**

  *Έναν* ***συντάκτη(editor)***.

- **Πώς ονομάζεται το πρόγραμμα που πληκτρολόγησα στον συντάκτη;**

  ***Πηγαίος κώδικας (source code)***.

- **Πώς θα το εκτελέσω («τρέξω»);**

  *Χρειάζεται ένας* ***μεταγλωττιστής (compiler)*** *ο οποίος αφού ελέγξει το κείμενο για απουσία συντακτικών λαθών, παράγει το* ***αντικείμενο (object) πρόγραμμα***.

- **Μπορώ να απολαύσω επιτέλους το αριστούργημά μου;**

  *Όχι ακόμα. Το αντικείμενο πρόγραμμα χρειάζεται κάποιες πληροφορίες που συνήθως βρίσκονται στις* ***βιβλιοθήκες*** *του λειτουργικού συστήματος. Τη σύνδεση του αντικειμένου με τις βιβλιοθήκες, τις αναλαμβάνει ο* ***συνδέτης (linker)*** *ο οποίος παράγει και το* ***εκτελέσιμο (executable)*** *πρόγραμμα.*

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.8 Μεταφραστές - *Μεταγλωττιστής*


**Τα βήματα είναι:**

1. **Συγγραφή προγράμματος** (→πηγαίο)
2. **Μεταγλώττιση** (→αντικείμενο)
3. **Σύνδεση** (→εκτελέσιμο)
4. **Εκτέλεση**

- **Τι επέκταση έχουν τα εκτελέσιμα προγράμματα στα MS-Windows;**
  - ***com** (COMmand)*
  - ***exe** (EXEcutable)*

- **Μπορώ να δώσω το καλλιτέχνημά μου στον φίλο μου που έχει Macintosh;**
  *Χλωμό! Αυτό είναι εφικτό μόνο σε cross platform περιβάλλοντα. Μπορείς όμως να το δώσεις στον άλλο φίλο σου που έχει κομματάκι καλύτερο επεξεργαστή!*

**Ας φτιάξουμε ένα προγραμματάκι!**

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.3 Το πρόγραμμα-Γλώσσες προγρ/σμού

### 7.3.8 Μεταφραστές - *Διερμηνευτής*

- **Λειτουργικά, σε τι διαφέρει ο διερμηνευτής (interpreter) από τον μεταγλωττιστή;**

*Ο μεταγλωττιστής μετατρέπει όλο το πηγαίο πρόγραμμα σε γλώσσα μηχανής, παράγοντας έτσι το εκτελέσιμο, το οποίο μπορούμε να το εκτελέσουμε όποια στιγμή το επιθυμούμε. Αντίθετα ο διερμηνευτής, μετατρέπει μία μία τις εντολές σε γλώσσα μηχανής και τις εκτελεί αμέσως.*

- **Πρακτικά, ποιες είναι οι διαφορές τους;**

*Ο διερμηνευτής*

- *εντοπίζει το συντακτικό λάθος σε μια εντολή, όταν αποπειραθεί να την μετατρέψει σε γλώσσα μηχανής και να την εκτελέσει (Δηλαδή, ξεκινάει την εκτέλεση του προγράμματος χωρίς να προαπαιτήσει την εκσφαλμάτωσή του)*

- *δεν παράγει εκτελέσιμο (Δηλαδή, για να εκτελεστεί σε έναν Η/Υ πρέπει σε αυτόν να είναι εγκατεστημένος ο συγκεκριμένος διερμηνευτής)*

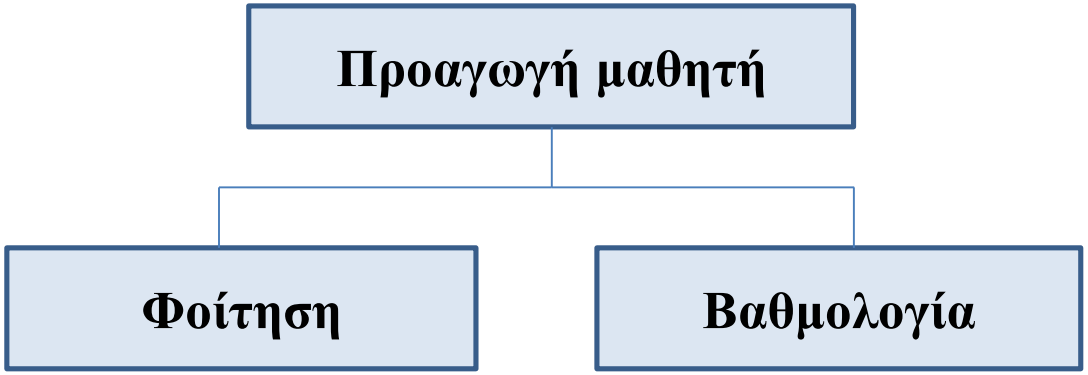
# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.4 Αρχές κατασκευής λογισμικού

**Ένα πρόγραμμα κατασκευάζεται από έναν προγραμματιστή ή από ομάδα προγραμματιστών;**

**Τμηματικός προγραμματισμός**

*Το πρόγραμμα διασπάται σε μικρότερα τμήματα, κάθε ένα από τα οποία θα κατασκευαστεί ανεξάρτητα. Στη συνέχεια θα συνδυαστούν όλα αυτά για να προκύψει το τελικό σύστημα.*



# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.4 Αρχές κατασκευής λογισμικού

**Υπορουτίνα**

*Είναι ένα τμήμα κώδικα στο οποίο έχουμε δώσει ένα όνομα και μπορεί να εκτελεστεί σε οποιοδήποτε σημείο του προγράμματος, απλώς αναφερόμενοι σε αυτό.*

```
ΕμβΚύκλου(r)
    ΕμβΚύκλου ← 2 * 3,14 * r
Τέλος
```

*Η συγκεκριμένη υπορουτίνα υπολογίζει το εμβαδό ενός κύκλου με ό,τι ακτίνα της ζητήσουμε*

```
Πρόγραμμα Υπολογισμοί
    .....
    ΕμβΚύκλου(5)
    .....
    ΕμβΚύκλου(13)
    .....
Τέλος
```

*Στο πρόγραμμα μπορούμε να καλέσουμε την υπορουτίνα όσες φορές χρειαζόμαστε με ό,τι ορίσματα θέλουμε*

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.4 Αρχές κατασκευής λογισμικού

**Δομημένος προγραμματισμός**

*Περιλαμβάνονται μόνο οι παρακάτω δομές ελέγχου:*

- ***διαδοχής ή ακολουθίας***

  *Οι εντολές εκτελούνται η μία μετά την άλλη με την σειρά που είναι γραμμένες*

- ***επιλογής***

  *Οι εντολές εκτελούνται μόνο αν ικανοποιείται η απαιτούμενη συνθήκη*

- ***επανάληψης***

  *Οι εντολές επαναλαμβάνονται όσο ικανοποιείται η απαιτούμενη συνθήκη*

**Ακολουθία**

**Επιλογή**

**Επανάληψη**

```
Γράψε 'Δώσε βαθμολογία'
Διάβασε βαθμολογία

Αν απουσίες > 60 τότε
    Γράψε 'Ελλιπής φοίτηση'
Τέλος

Όσο  κόστος  <  2000 επανάλαβε
    Γράψε 'Μπορείς να αγοράσεις κι άλλα'
    Διάβασε τιμή
    κόστος  ← κόστος + τιμή
Τέλος
```

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.5 Πρότυπα προγραμματισμού

### 7.5.1 Διαδικαστικός προγραμματισμός *(procedural programming)*

*Το πρόγραμμα περιλαμβάνει δύο ξεχωριστά δομικά στοιχεία:*

- *τις **εντολές**, που περιγράφουν βήμα βήμα τη διαδικασία επίλυσης του προβλήματος*
- *τις **δομές δεδομένων** στις οποίες αποθηκεύονται τα δεδομένα του προγράμματος*

*Π.χ. C, Java, Pascal*

**Οι διαδικαστικές γλώσσες προγρ/σμού περιλαμβάνουν τουλάχιστον τις παρακάτω εντολές:**

| | |
|---|---|
| **Ανάθεσης** | A := B + C |
| **Συνθήκης** | If grade>=18.5 then print “Άριστα” |
| **Επανάληψης** | while a>0 do<br>begin<br>  a := a – 1<br>end |
| **Εισόδου & εξόδου στοιχείων** | readln(grade)<br>writeln(average) |

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.5 Πρότυπα προγραμματισμού

### 7.5.2 Αντικειμενοστραφής προγραμματισμός *(object oriented programming)*

*Ένα σύστημα μπορεί να περιγραφεί με τα **αντικείμενα** από τα οποία αποτελείται. Κάθε αντικείμενο περιλαμβάνει τις διαδικασίες **(μεθόδους)** που μπορεί να υποστεί και τα δεδομένα **(ιδιότητες)** που το χαρακτηρίζουν. Τα αντικείμενα μπορούν να αλληλεπιδράσουν μέσω **μηνυμάτων**.*

| **Αντικείμενο** | κύκλος | παιδί |
|---|---|---|
| **Ιδιότητες** | μήκος ακτίνας | φύλο, ύψος, έτος_γέννησης |
| **Μέθοδοι** | Υπολογισμός διαμέτρου<br>Υπολογισμός περιμέτρου<br>Υπολογισμός εμβαδού | Περπάτημα<br>Στρίψιμο κεφαλιού προς τα αριστερά<br>Ύψωση δεξιού χεριού |
| **Μηνύματα** | «Πες μου την περίμετρό σου» | «Περπάτησε 10 βήματα» |

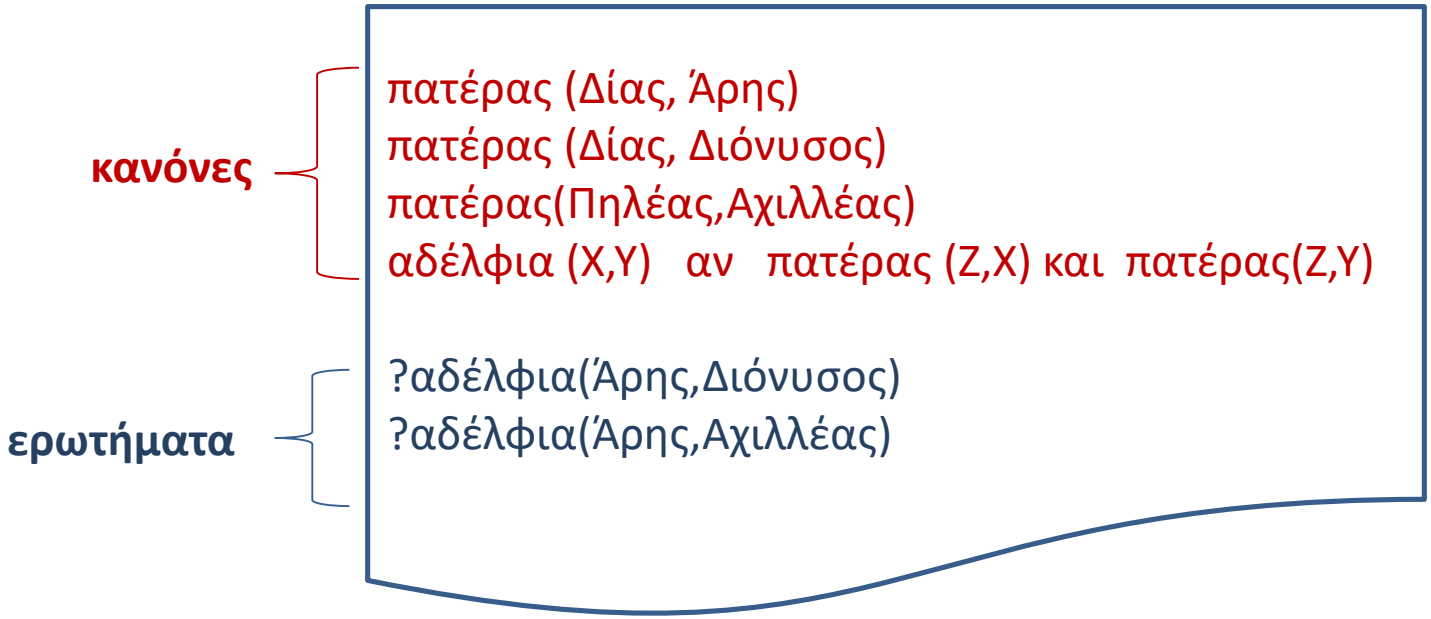
**Π.χ. C++, Java**

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.5 Πρότυπα προγραμματισμού

### 7.5.3 Λογικός προγραμματισμός

*Περιλαμβάνει ένα σύνολο λογικών προτάσεων και ένα μηχανισμό εξαγωγής συμπερασμάτων.*




**Π.χ. Prolog**

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.5 Πρότυπα προγραμματισμού

### 7.5.3 Συναρτησιακός προγραμματισμός

*Βασίζεται στις συναρτήσεις όπως αυτές εννοούνται στα μαθηματικά. Η μόνη δομή ελέγχου είναι η εφαρμογή συναρτήσεων πάνω σε δεδομένα.*

***Ας δούμε τον υπολογισμό του n! (παραγοντικό)*** *Αν n=0 τότε n!=0 αλλιώς n!=1*2*3*....*n*

**Διαδικαστικός προγρ/σμός**

```
factorial := 1
while (n > 0) do
begin
    factorial := factorial * n;
    n :=n-1
end
```

**Συναρτησιακός προγρ/σμός**

```
factorial n
    αν n=0  τότε 1
    αν n>0 τότε n*factorial(n-1)
```

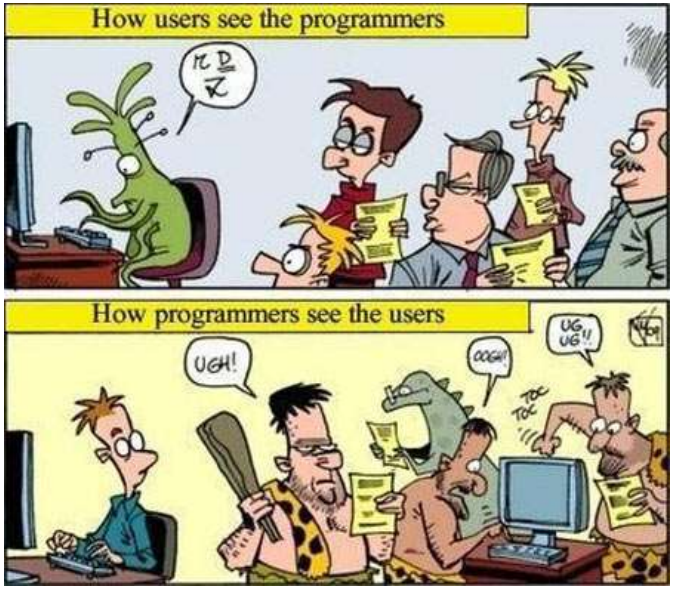
**Π.χ. Lisp**

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.6 Προγραμματίζοντας

Η **μέθοδος ανάπτυξης λογισμικού** περιλαμβάνει τα εξής βήματα:

1. ***Καθορισμός προβλήματος/απαιτήσεων***
2. ***Ανάλυση προβλήματος***
3. ***Σχεδιασμός αλγορίθμου για την επίλυση του προβλήματος***
4. ***Υλοποίηση του αλγορίθμου***
5. ***Έλεγχος του τελικού προγράμματος***
6. ***Συντήρηση του προγράμματος***
7. ***Τεκμηρίωση***



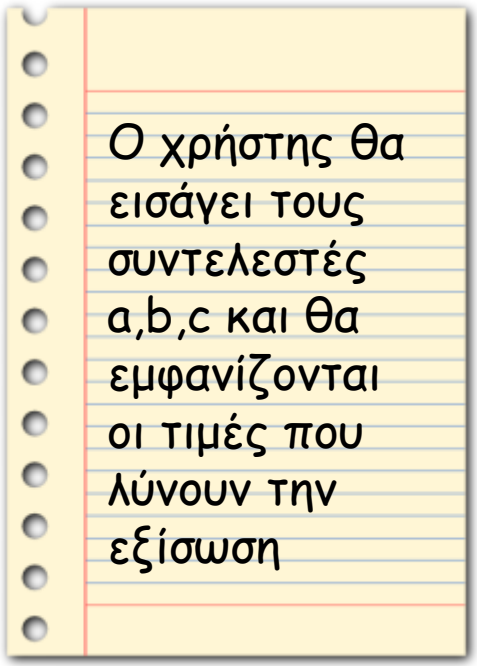

**Ας τα δούμε αναλυτικά μέσω ενός παραδείγματος:** να γράψουμε πρόγραμμα που να υπολογίζει τις ρίζες μιας δευτεροβάθμιας εξίσωσης. $ax^2+bx+c=0$

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.6 Προγραμματίζοντας

**Καθορισμός προβλήματος/απαιτήσεων**

*Μελετάμε το πρόβλημα εξαντλητικά, προσδιορίζουμε τα δεδομένα και καταγράφουμε τα ζητούμενα.*




Πόσο κρίσιμο είναι αυτό το στάδιο;

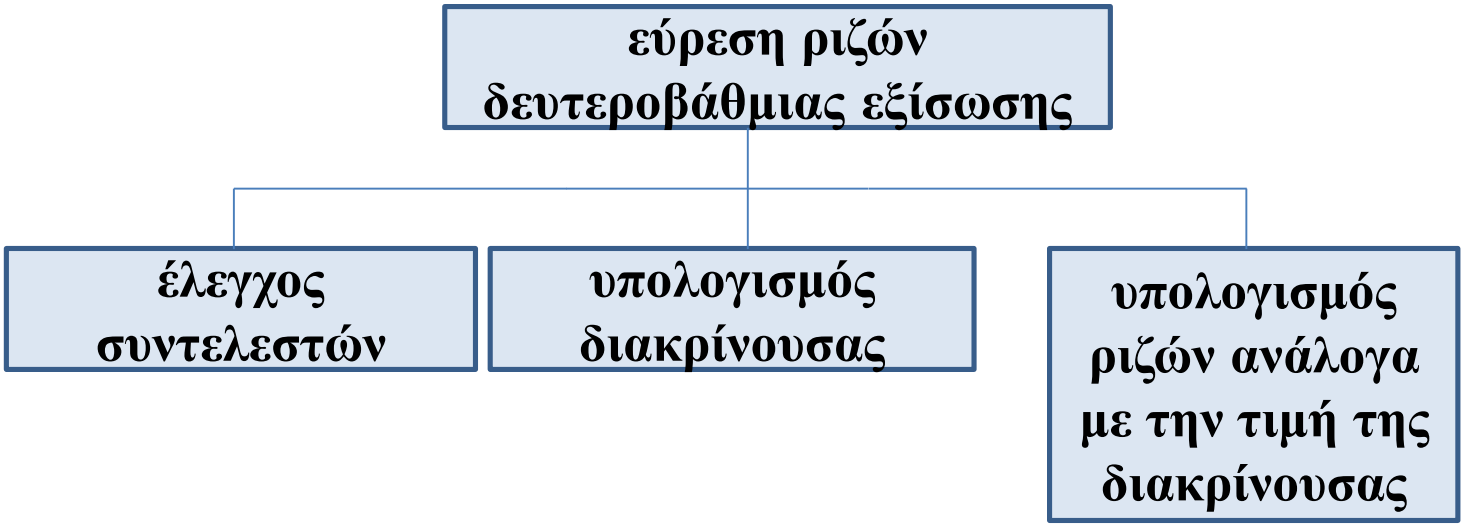
Ο προγραμματιστής πρέπει να γνωρίζει πώς λύνονται όλα τα μαθηματικά προβλήματα;

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.6 Προγραμματίζοντας

**Ανάλυση προβλήματος**

*Διασπάμε το πρόβλημα σε επιμέρους μικρότερα και αυτά σε ακόμα μικρότερα μέχρι να φτάσουμε στο σημείο όπου τα επιμέρους προβλήματα είναι τόσο απλά ώστε να μην χρήζουν περαιτέρω διάσπασης.*




Πόσο αντικειμενικό είναι αυτό το στάδιο;

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.6 Προγραμματίζοντας

**Αλγόριθμος** είναι μια σειρά από βήματα, τα οποία πρέπει να ακολουθήσουμε για να επιλύσουμε ένα συγκεκριμένο πρόβλημα.

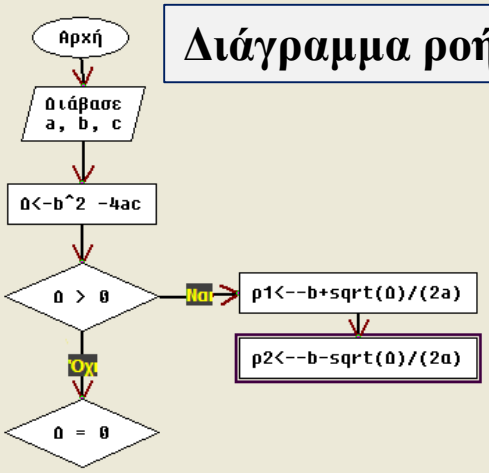
### Σχεδιασμός αλγορίθμου

*Καταγράφουμε τα βήματα επίλυσης του προβλήματος και τα διατυπώνουμε με έναν από τους παρακάτω τρόπους.*

**Φυσική γλώσσα**

1) *Να διαβάσω τους συντελεστές a,b,c*
2) *Να υπολογίσω τη διακρίνουσα με τον τύπο $b^2$-4ac*
3) *Αν Δ>0 τότε …..*

**Διάγραμμα ροής**




**Ψευδοκώδικας**

ΓλωσσοΜάθεια

Πρόγραμμα Επεξεργασία Αναζήτηση

Εκτέλεση

```
1  ΠΡΟΓΡΑΜΜΑ λολο
2  ΜΕΤΑΒΛΗΤΕΣ
3    ΠΡΑΓΜΑΤΙΚΕΣ: a,b,c,D, ρ1, ρ2
4  ΑΡΧΗ
5    ΔΙΑΒΑΣΕ a,b,c
6    D <-- b^2 + 4 * a * c
7    Αν D>0 τότε
8      ρ1 <-- (-b+sqrt(D))/(2*a)
9      ρ2 <-- (-b-sqrt(D))/(2*a)
10     Γράψε 'Έχει 2 πραγματικές ρίζες', ρ1, ρ2
11   αλλιώς_αν D=0 τότε
12     ρ1 <-- (-b)/(2*a)
13     Γράψε 'Έχει 1 πραγματική ρίζα', ρ1
14   αλλιώς
15     Γράψε 'Δεν έχει πραγματικές ρίζες'
16   τέλος_αν
17 ΤΕΛΟΣ_ΠΡΟΓΡΑΜΜΑΤΟΣ
```

**Ποια αναπαράσταση προτιμάτε;**

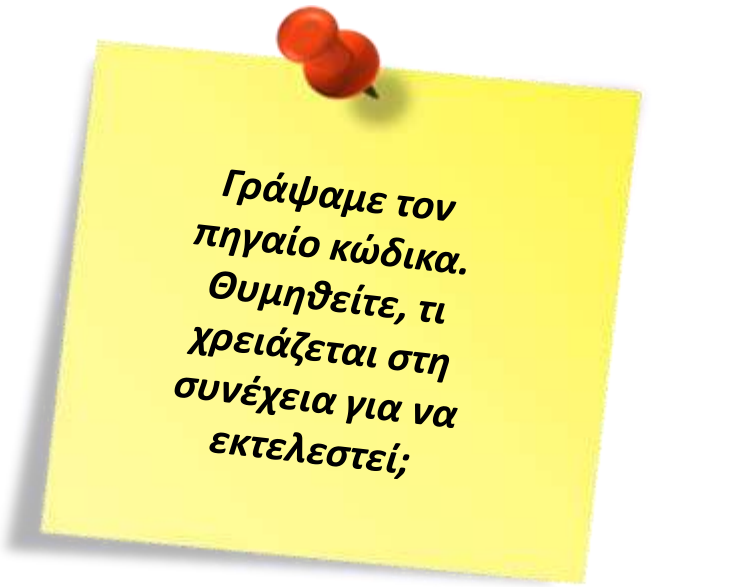
# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.6 Προγραμματίζοντας

### Υλοποίηση του αλγορίθμου

*Ο αλγόριθμος διατυπώνεται σε μια γλώσσα προγραμματισμού και πλέον αποτελεί ένα* ***πρόγραμμα*** *(εφαρμογή/application)!*

```
program week7_lab2_a1;
var a,b,c,i:integer;
x,x1,x2:real;

begin
  write('Enter the value of a :');
  readln(a);

  write('Enter the value of b :');
  readln(b);

  write('Enter the value of c :');
  readln(c);

  if (sqr(b)-4*a*c)>=0 then
    begin
      if ((a>0) and (b>0)) then
        begin
          x1:=(-1*b+sqrt(sqr(b)-4*a*c))/2*a;
          x2:=(-1*b-sqrt(sqr(b)-4*a*c))/2*a;

          writeln('x1=',x1:0:2);
          writeln('x2=',x2:0:2);
        end
      else
```



# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.6 Προγραμματίζοντας

### Έλεγχος του προγράμματος

*«Τρέχουμε» το πρόγραμμά μας με διαφορετικές εισόδους, φροντίζοντας να καλύψουμε όλες τις διαφορετικές περιπτώσεις. Εντοπίζουμε τυχόν λάθη και τα διορθώνουμε* ***(εκσφαλμάτωση/debugging)****.*

Στην περίπτωση πολύπλοκων προγραμμάτων, η διαδικασία του ελέγχου αποτελεί μια επίπονη και χρονοβόρα διαδικασία.

### Συντήρηση του προγράμματος

- *Διόρθωση σφαλμάτων που εντοπίστηκαν κατά τη χρήση του από τους πραγματικούς χρήστες του προγράμματος*
- *Τροποποιήσεις (μικρές ή μεγάλες) που προέκυψαν (π.χ. αλλαγή νομίσματος, ΦΠΑ)*

Τι δηλώνει η έκδοση (version) ενός προγράμματος;

### Τεκμηρίωση του προγράμματος (documentation)

*Έγγραφα* ***(εξωτερική τεκμηρίωση)*** *ή σχόλια* ***(εσωτερική τεκμηρίωση)*** *που σκοπό έχουν να καταγράψουν τη διαδικασία κατασκευής αλλά και λειτουργίας του εν λόγω προγράμματος.*

Μια βαρετή, αλλά αναγκαία διαδικασία!

# Κεφ. 7 - Προγραμματισμός υπολογιστή

## 7.7 Προγραμματιστικά περιβάλλοντα

*Τα εργαλεία λογισμικού που αξιοποιούνται στην ανάπτυξη άλλων προγραμμάτων ονομάζονται* ***προγραμματιστικά περιβάλλοντα.***

Μήπως τυχαίνει να γνωρίζετε κάποιο;

**Ένα προγραμματιστικό περιβάλλον συνήθως διαθέτει:**

- *συντάκτη κειμένων (editor)*
- *μεταφραστές (assembler, compiler, interpreter)*
- *εργαλεία εντοπισμού λαθών (debuggers)*

**Οπτικός προγραμματισμός (visual programming)**

*Στα περιβάλλοντα όπου υποστηρίζεται ο οπτικός προγραμματισμός, μεγάλο μέρος του προγράμματος κατασκευάζεται, αξιοποιώντας γραφικά εργαλεία (εικονίδια, μενού κ.τ.λ.).*

*Π.χ. Alice, GameMaker, Delphi, Scratch*