

Πανεπιστήμιο Πάτρας

Τμήμα Διοίκησης Επιχειρήσεων

**ΕΙΣΑΓΩΓΗ ΣΤΟΥΣ Η/Υ: ΟΡΓΑΝΩΣΗ ΚΑΙ
ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ**

Ακαδημαϊκό έτος 2023 – 2024

Διδάσκων: Γιάννης Σταματίου

Email: stamatiu@upatras.gr

Προγραμματισμός Η/Υ: Αλγόριθμοι και γλώσσες προγραμματισμού

Η εξέλιξη των γλωσσών προγραμματισμού

Κάθε πρόγραμμα Η/Υ φτιάχνεται σε μια γλώσσα προγραμματισμού που μπορεί να είναι κατανοητή από τον άνθρωπο. Για να εκτελεστούν από ένα Η/Υ αυτά τα προγράμματα πρέπει να μεταφραστούν σε **κώδικα μηχανής**.

Οι γλώσσες προγραμματισμού διακρίνονται:

- Γλώσσες μηχανής (Machine languages)
- Σε συμβολικές γλώσσες (Assembly languages)
- Σε γλώσσες Υψηλού-Επιπέδου (High-level languages)

Mother Tongues

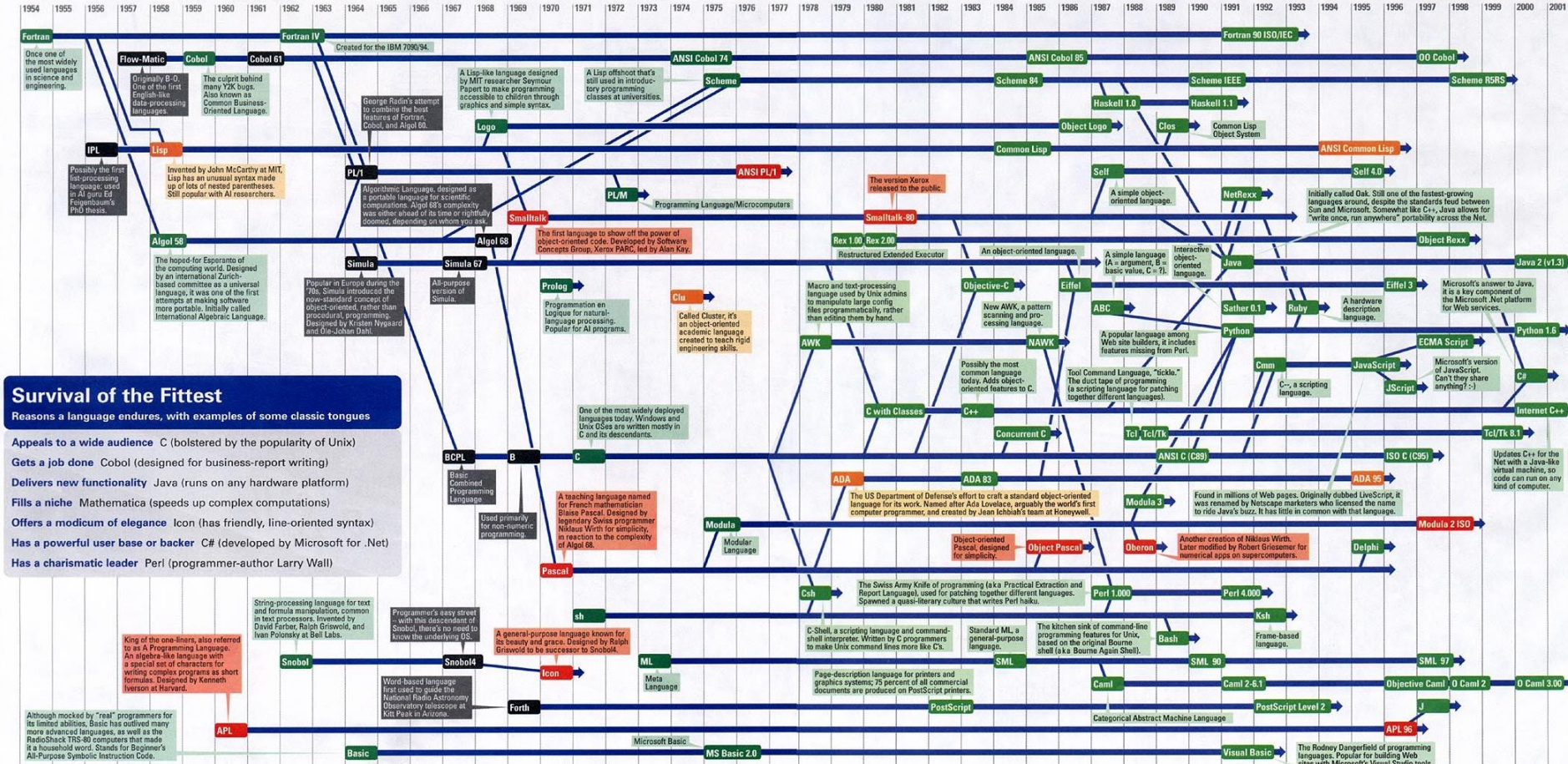
Tracing the roots of computer languages through the ages

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 francas. Among the most endangered are Ada, APL, B (the predecessor of C), Lisp, Oberon, Smalltalk, and Simula.

Code-raker Grady Booch, Rational Software's chief scientist, is working with the Computer History Museum 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 www.informatik.uni-freiburg.de/Java/misc/lang_list.html. — Michael Menduno

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	



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

Γλώσσα προγραμματισμού	Δυνατότητες	Τυπική χρήση
C/C++ και C#	<ul style="list-style-type: none"> Μπορεί να δημιουργήσει συμπαγή κώδικα που εκτελείται γρήγορα Παρέχει πρόσβαση υψηλού και χαμηλού επιπέδου 	<ul style="list-style-type: none"> Χρησιμοποιείται σε βιομηχανικές εφαρμογές, όπως τραπεζικά συστήματα και μηχανική
Java	<ul style="list-style-type: none"> Αρχιτεκτονικά ουδέτερη Αντικειμενοστραφής 	<ul style="list-style-type: none"> Χρησιμοποιείται για τη δημιουργία applet που παραδίδονται στο web
Objective C	<ul style="list-style-type: none"> Διαθέτει πλαίσιο εργασίας για δημιουργία εφαρμογών iOS 	<ul style="list-style-type: none"> Χρησιμοποιείται για τη δημιουργία εφαρμογών για το macOS και για φορητές συσκευές της Apple
Visual Basic	<ul style="list-style-type: none"> Εύκολη στην εκμάθηση και τη χρήση Αντικειμενοστραφής Διαθέτει διεπαφή μεταφοράς και απόθεσης 	<ul style="list-style-type: none"> Χρησιμοποιείται για την ανάπτυξη πρωτοτύπων Χρησιμοποιείται για τη σχεδίαση γραφικών διεπαφών χρήστη
Τεχνολογίες web	Δυνατότητες	Τυπική χρήση
AJAX	<ul style="list-style-type: none"> Χρησιμοποιεί έναν συνδυασμό υπαρχουσών τεχνολογιών, όπως JavaScript, CSS και XML 	<ul style="list-style-type: none"> Δημιουργεί διαδικτυακούς τόπους που μπορούν να ενημερώνονται χωρίς να γίνει ανανέωση σελίδας από τον χρήστη
HTML5	<ul style="list-style-type: none"> Τελευταία έκδοση της HTML 	<ul style="list-style-type: none"> Νέες ετικέτες όπως <video> και υποστηρίζει μεταφορά και απόθεση
VBScript	<ul style="list-style-type: none"> Παρόμοια στη σύνταξη με τη Visual Basic Έχει κλάσεις που αναπαριστούν κουμπιά, αναπτυσσόμενες λίστες και άλλα στοιχεία ιστοσελίδων 	<ul style="list-style-type: none"> Δημιουργεί κώδικα που βρίσκεται στον υπολογιστή πελάτη και προσθέτει αλληλεπίδραση με ιστοσελίδες
XML	<ul style="list-style-type: none"> Επιτρέπει στους χρήστες να ορίζουν τις δικές τους ετικέτες 	<ul style="list-style-type: none"> Διευκολύνει την ανταλλαγή πληροφοριών από υπηρεσίες web
JSON	<ul style="list-style-type: none"> Η μορφοποίηση κειμένου ορίζεται με ζεύγη ονόματος/τιμής 	<ul style="list-style-type: none"> Πολύ κοινή μορφή για ανταλλαγή πληροφοριών από υπηρεσίες web

Γλώσσες Μηχανής

- Οι γλώσσες μηχανής (γλώσσες 1ης γενιάς) είναι η πιο βασική μορφή γλώσσας προγραμματισμού και περιλαμβάνουν εντολές που η μορφή τους είναι σειρές από δυαδικούς αριθμούς.
- Η γλώσσα μηχανής είναι η μόνη γλώσσα που το hardware καταλαβαίνει άμεσα.
- **Διαφορετικές CPUs χρησιμοποιούν διαφορετικές γλώσσες μηχανής.** Π.χ. οι IBM H/Y διαθέτουν διαφορετική γλώσσα μηχανής από ότι οι Apple H/Y.
- **Intel vs. Motorola** (διαφορετικές CPUs)

Γλώσσα Μηχανής

1	00000000	00000100	000000000000000000
2	01011110	00001100110000100000000000000010	
3		11101111	00010110000000000000000101
4		11101111	10011110 000000000000001011
5	11111000	10101101	11011111 00000000000010010
6		0110001011011111	00000000000010101
7	11101111	00000010	11111011 00000000000010111
8	11110100	1010110111011111	00000000000011110
9	0000001110100010	11011111	00000000000100001
10	11101111	00000010	11111011 00000000000100100
11	01111110	11110100	10101101
12	11111000	10101110	110001010000000000101011
13	0000011010100010	11111011	00000000000110001
14	11101111	00000010	11111011 00000000000110100
15		00000100	00000000000111101
16		00000100	00000000000111101

Γλώσσα Assembly (συμβολική γλώσσα)

- Οι συμβολικές γλώσσες (γλώσσες 2ης γενιάς) είναι κάπως ευκολότερες στην χρήση για τον άνθρωπο από ότι οι γλώσσες μηχανής.
- Για να αναπτύξει κάποιος προγράμματα σε γλώσσα assembly, χρησιμοποιεί κρυπτογραφικές φράσεις της Αγγλικής. Οι φράσεις αυτές αντικαθιστούν τις σειρές από δυαδικούς αριθμούς της γλώσσας μηχανής.
- Ο κώδικας σε assembly κατόπιν μεταφράζεται σε αντικειμενικό κώδικα (object code) χρησιμοποιώντας ένα μεταφραστικό πρόγραμμα που καλείται **assembler**.

```

;CLEAR SCREEN USING BIOS
CLR: MOV AX,0600H      ;SCROLL SCREEN
    MOV BH,30          ;COLOUR
    MOV CX,0000        ;FROM
    MOV DX,184FH       ;TO 24,79
    INT 10H            ;CALL BIOS;
;INPUTTING OF A STRING
KEY: MOV AH,0AH        ;INPUT REQUEST
    LEA DX,BUFFER      ;POINT TO BUFFER WHERE STRING STORED
    INT 21H            ;CALL DOS
    RET                ;RETURN FROM SUBROUTINE TO MAIN PROGRAM;
; DISPLAY STRING TO SCREEN
SCR: MOV AH,09          ;DISPLAY REQUEST
    LEA DX,STRING      ;POINT TO STRING
    INT 21H            ;CALL DOS
    RET                ;RETURN FROM THIS SUBROUTINE;

```

Assembly code

Assembler

```

0001010010110101010101010101010100010
11101101010101010101010101110010100010110
0010100101010010111101011101011101010
100101001011010101010101010101010110110
0110100100110010111101011101010100010
0001000101011101010101000101010111010
1010100101010010101101011101011101011
0001010010110101010101010101010100010

```

Object code

```

int dotp( short a [ ], short b [ ] )
{
    int sum, i;
    int sum1 = 0 ;
    int sum2 = 0 ;

    for( i = 0; i < 100/2; i+2 )
    {
        sum1 += a[i] * b[i];
        sum2 += a[i+1] * b[i+1];
    }
    return sum1 + sum2;
}

```

(a) C Code for Dot Product.

```

_dotp .cproc a, b
    reg sum1, sum2, i
    reg val_1, val_2, prod_1, prod_2
    mvk    50, i          ; i = 100/2
    zero   sum1           ; Set sum1 = 0
    zero   sum2           ; Set sum2 = 0
loop:
    ldw    *a++, val_1    ; load a[0, 1] and add a by 1
    ldw    *b++, val_2    ; load b[0, 1] and add b by 1
    mpy    val_1, val_2, prod_1 ; a[0] * b[0]
    mpyh   val_1, val_2, prod_2 ; a[1] * b[1]
    add    prod_1, sum1, sum1 ; sum1 += a[0] * b[0]
    add    prod_2, sum2, sum2 ; sum2 += a[1] * b[1]
    add    -1, i, i        ; i--
    [i] b   loop           ; if i>0, goto loop
    add    sum1, sum2, A4   ; get final result
    .return A4
    .endproc

```

Loop Body

(b) Assembly Code for Dot Product.

High-Level Languages

Οι γλώσσες υψηλού επιπέδου είναι πιο δυναμικές και πιο κατανοητές από τον άνθρωπο αφού χρησιμοποιούν περισσότερες λέξεις και προτάσεις της Αγγλικής.

Οι γλώσσες υψηλού επιπέδου διακρίνονται σε 3 “γενιές”:

- 3ης γενιάς (3GL)
- 4ης γενιάς (4GL)
- 5ης γενιάς (5GL)

Γλώσσες Υψηλού Επιπέδου (3GL)

- Είναι αρχαιότερες γλώσσες υψηλού επιπέδου. Είναι μεταφέρσιμες. Δηλαδή, ο object code που δημιουργούν για ένα τύπο συστήματος μπορεί να «τρέξει» και σε συστήματα άλλων τύπων.
- Για κάθε γλώσσα υπάρχει χωριστό μεταφραστικό πρόγραμμα:
 - **Compiler** (μεταγλωττιστής)
 - **Interpreter** (Διερμηνευτής).
- Οι παρακάτω είναι γλώσσες 3GLs:

FORTAN

COBOL

BASIC

Pascal

C

C++

Java

ActiveX

Γλώσσες Υψηλού Επιπέδου– 4GLs

- Είναι ακόμη πιο εύκολες στην χρήση από ότι οι 3GLs.
- Επιτρέπουν την εργασία σε ένα **οπτικό περιβάλλον** (visual environment) με χρήση γραφικών εργαλείων όπως τα Windows.
- Οι παρακάτω είναι γλώσσες 4GLs:

Visual Basic (VB)

VisualAge

Visual C

Γλώσσες Υψηλού Επιπέδου

/* Αυτό το πρόγραμμα είναι γραμμένο σε γλώσσα C. Διαβάζει δύο ακεραίους αριθμούς από το πληκτρολόγιο και τυπώνει το γινόμενο τους.

***/**

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

```
int main (void)
```

```
{
```

```
    int number1, number2, result;
```

```
    printf("Δώσε δύο αριθμούς:");
```

```
    scanf("%d", &number1); scanf("%d", &number1);
```

```
    result = number1 * number2;
```

```
    printf("Το γινόμενο είναι %d", result);
```

```
}
```

Γλώσσες Υψηλού Επιπέδου

(* Αυτό το πρόγραμμα είναι γραμμένο σε γλώσσα **Pascal**. Διαβάζει δύο ακραίους αριθμούς από το πληκτρολόγιο και τυπώνει το γινόμενο τους.

*)

```
Program Product_of_two_numbers;
```

```
Var
```

```
    number1, number2, result : integer;
```

```
Begin
```

```
    write("Δώσε δύο αριθμούς:");
```

```
    readln(number1, number1);
```

```
    result = number1 * number2;
```

```
    writeln("Το γινόμενο είναι ", result);
```

```
End.
```

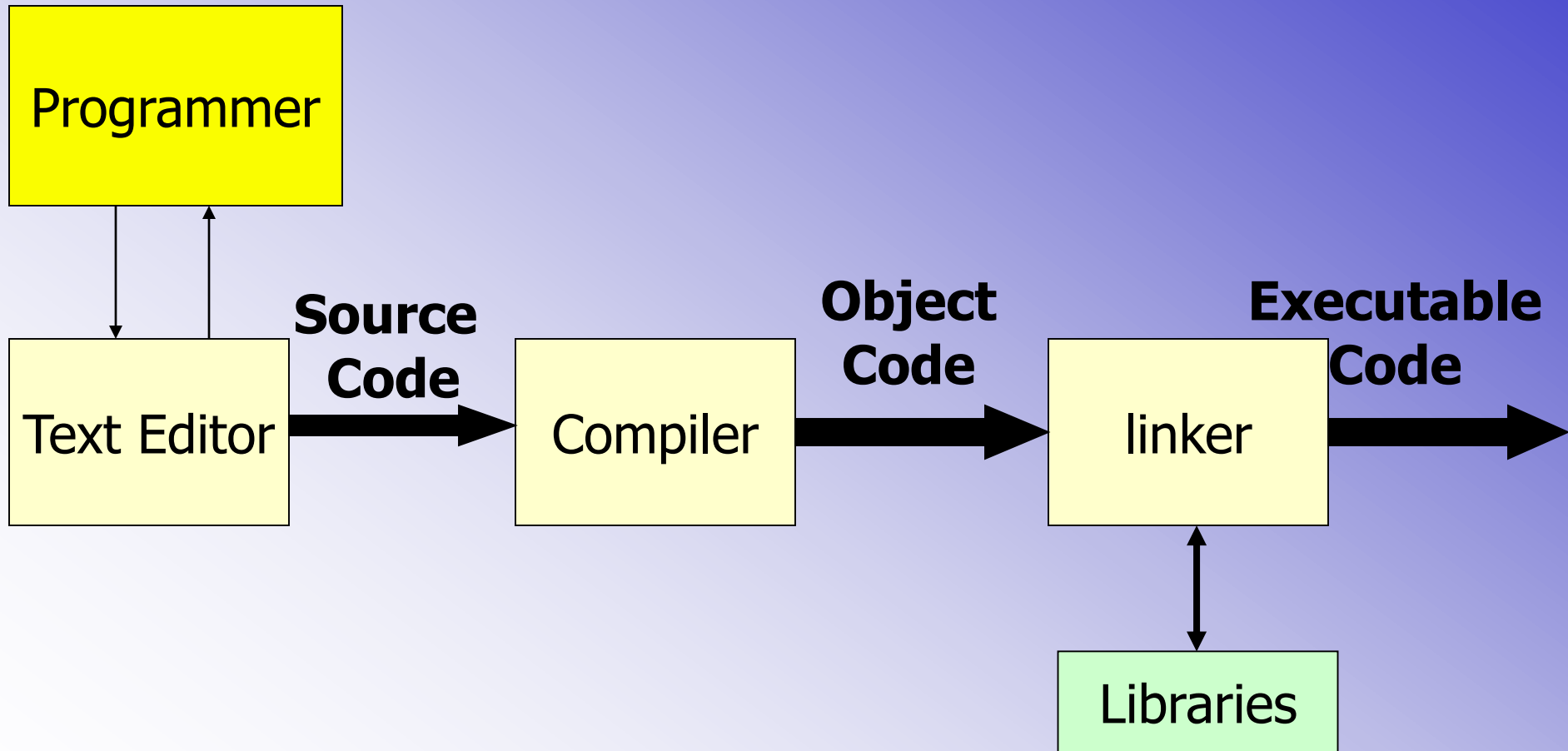
Γλώσσες Υψηλού Επιπέδου - 5GLs

- Είναι η τελευταία γενιά γλωσσών. Κάποιοι προγραμματιστές θεωρούν ότι δεν υφίσταται 5^η γενιά γλωσσών.
- Οι γλώσσες αυτές θα χρησιμοποιούν στοιχεία και μηχανισμούς της **Τεχνητής Νοημοσύνης** για τη δημιουργία προγραμμάτων.

Κατασκευή ενός Προγράμματος

- Δουλειά του προγραμματιστή είναι να γράψει ένα πρόγραμμα και μετά να το μετατρέψει σε **εκτελέσιμο αρχείο**. Αυτή η διαδικασία έχει τρία βήματα:
 1. Συγγραφή και διόρθωση του προγράμματος
 2. Μεταγλώττιση του προγράμματος
 3. Σύνδεση του προγράμματος με τις απαραίτητες υπομονάδες βιβλιοθηκών

Compilers (μεταγλωττιστές)



Αλγόριθμοι

Αλγόριθμοι (Algorithms)

- Ένας αλγόριθμος είναι μια σειρά από βήματα που πρέπει ακολουθήσουμε ώστε να φτάσουμε στην λύση ενός προβλήματος.
- Τα βήματα ενός αλγορίθμου είναι ίδια είτε επιλύουμε το πρόβλημα με το χέρι, είτε μέσω προγράμματος υπολογιστή.
- Η ανάπτυξη αλγορίθμων είναι το προστάδιο της υλοποίησης ενός προγράμματος Η/Υ.

Αλγόριθμοι - συνέχεια

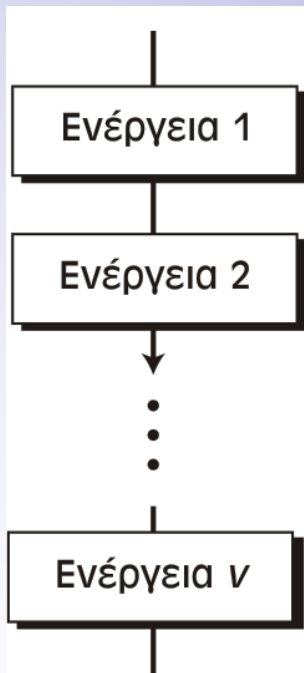
Σκοπός του αλγορίθμου είναι η επίλυση ενός προβλήματος μέσω ηλεκτρονικού υπολογιστή.

Ένας αλγόριθμος δέχεται κάποια δεδομένα (**είσοδος**) και παράγει κάποια αποτελέσματα (**έξοδος**).

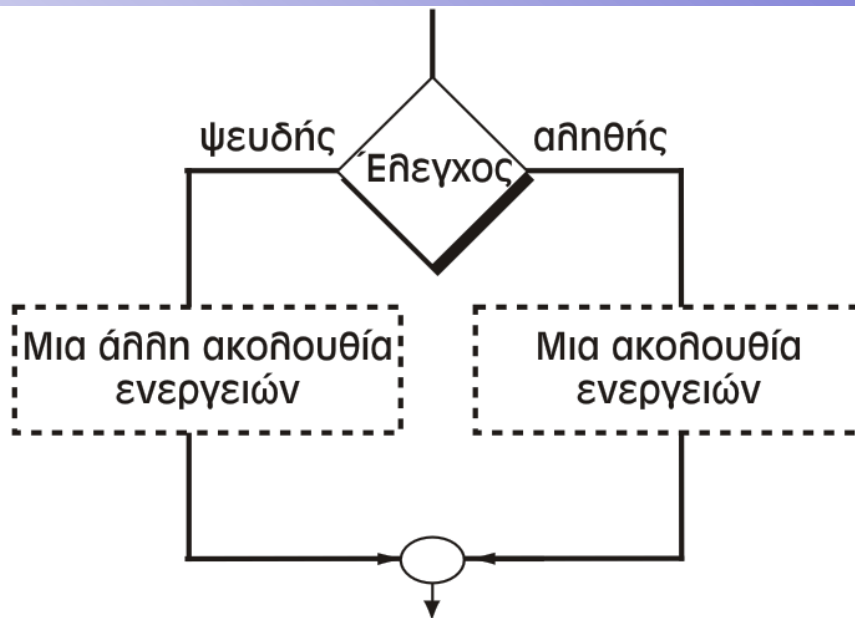
Ο αλγόριθμος είναι άμεσα κατανοητός από τον άνθρωπο (όχι από τους ηλεκτρονικούς υπολογιστές) και συντάσσεται:

- σε **φυσική γλώσσα**
- σε **ψευδοκώδικα**
- σε **διάγραμμα ροής**

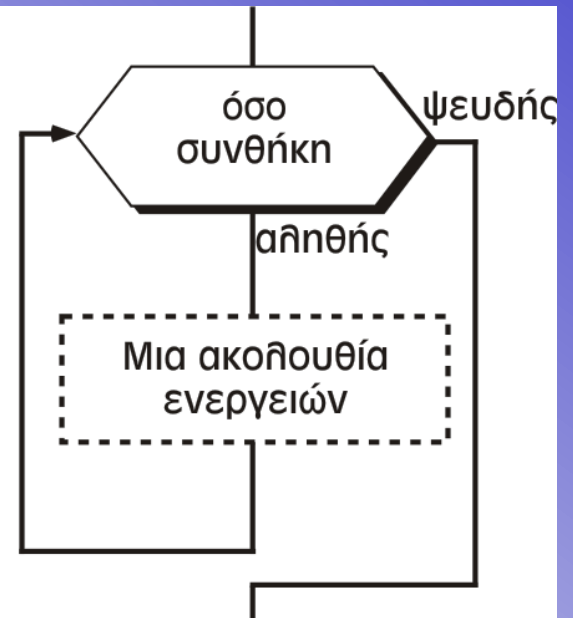
Στη συνέχεια μετατρέπεται σε πρόγραμμα Η/Υ με χρήση κάποιας γλώσσας προγραμματισμού, ώστε να μπορεί να εκτελεστεί



α. Ακολουθία



β. Απόφαση



γ. Επανάληψη

ενέργεια 1
ενέργεια 2

⋮

ενέργεια n

α. Ακολουθία

αν (συνθήκη)

τότε

ενέργεια

ενέργεια

...

αληθώς

ενέργεια

ενέργεια

...

Τέλος αν

β. Απόφαση

όσο (συνθήκη)

ενέργεια

ενέργεια

...

Τέλος όσο

γ. Επανάληψη

Αλγόριθμοι - συνέχεια

Κρίσιμη παρατήρηση:

- Ο αλγόριθμος πρέπει να μπορεί να επιλύει μια **ομάδα προβλημάτων** και όχι μία συγκεκριμένη περίπτωση του προβλήματος.
- Για παράδειγμα, δεν κατασκευάζουμε έναν αλγόριθμο που να λύνει την εξίσωση $3x + 2 = 5$, αλλά έναν αλγόριθμο που να λύνει την εξίσωση $ax + b = c$. Τότε ο αλγόριθμος θα είναι ικανός να λύνει το συγκεκριμένο πρόβλημα αλλά και κάθε πρωτοβάθμια εξίσωση, ανεξαρτήτως των τιμών των a , b και c .

Απλά προβλήματα αλγορίθμων

Πρόβλημα 1: Να υπολογιστεί ο μέσος όρος 3 αριθμών.

Είσοδος: Τρεις αριθμοί

Έξοδος: Ο μέσος όρος των 3 αριθμών

Αλγόριθμος σε φυσική γλώσσα:

Διάβασε 3 αριθμούς. Υπολόγισε το άθροισμα τους. Διαίρεσε το άθροισμα με το πλήθος των αριθμών και εκτύπωσε το αποτέλεσμα της διαίρεσης.

Πρόβλημα 1: Να υπολογιστεί ο μέσος όρος 3 αριθμών.

Αλγόριθμος σε μορφή ψευδοκώδικα

Βήμα 1. Διάβασε 3 αριθμούς στις μεταβλητές A, B, C

Βήμα 2. Υπολόγισε $SUM = A+B+C$

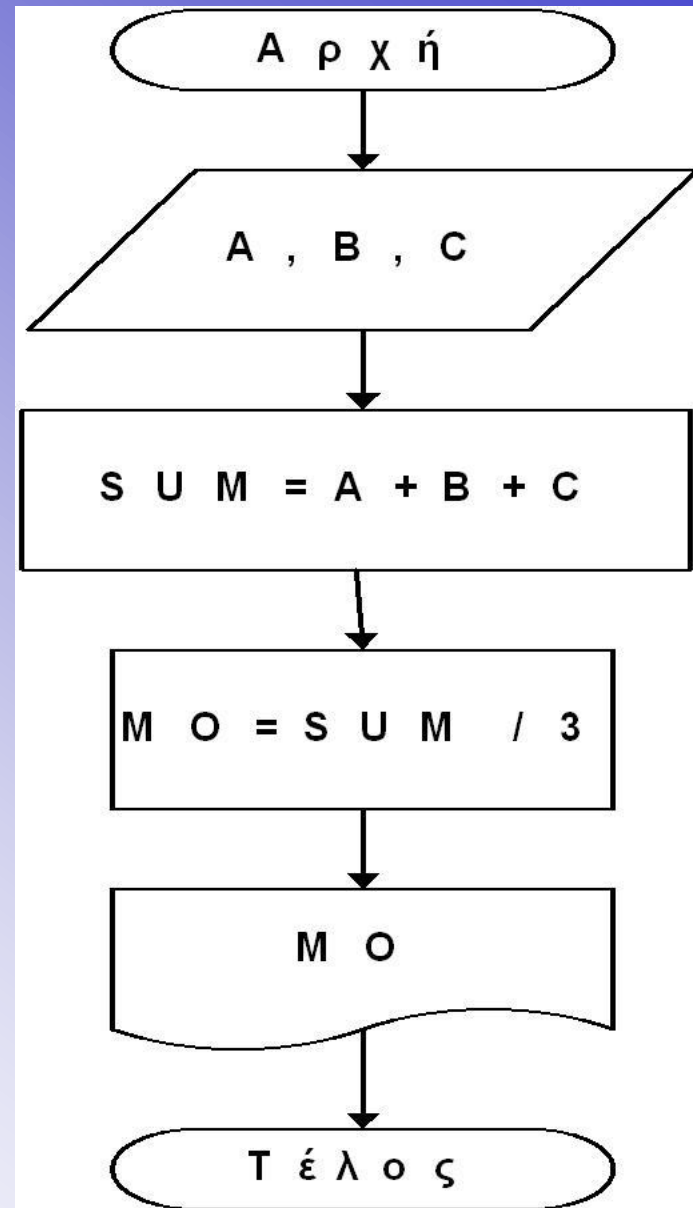
Βήμα 3. Υπολόγισε $MO = SUM / 3$

Βήμα 4. Τύπωσε “Μέσος όρος= “, MO

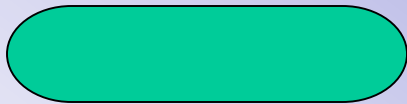
Βήμα 5. Τέλος

Πρόβλημα 1:
Να υπολογιστεί
ο μέσος όρος 3
αριθμών.

Αλγόριθμος
σε μορφή
διαγράμματος
ροής



Βασικοί συμβολισμοί στα διαγράμματα ροής



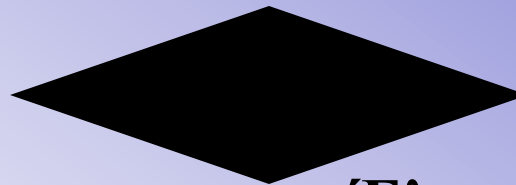
Αρχή ή τέλος



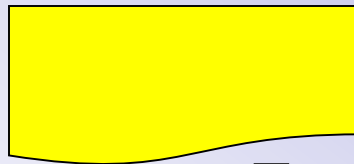
Επεξεργασία



Είσοδος δεδομένων
(ανάγνωση)



Έλεγχος συνθήκης



Εκτύπωση
αποτελεσμάτων

Πρόβλημα 2: Να βρεθεί ο μεγαλύτερος από 3 αριθμούς.

Είσοδος: Οι 3 αριθμοί.

Έξοδος: Ο μεγαλύτερος αριθμός.

Ο αλγόριθμος σε μορφή ψευδοκώδικα

1: Διάβασε 3 αριθμούς A, B, C

2: Θέσε ως μέγιστο τον πρώτο, **MAX = A**

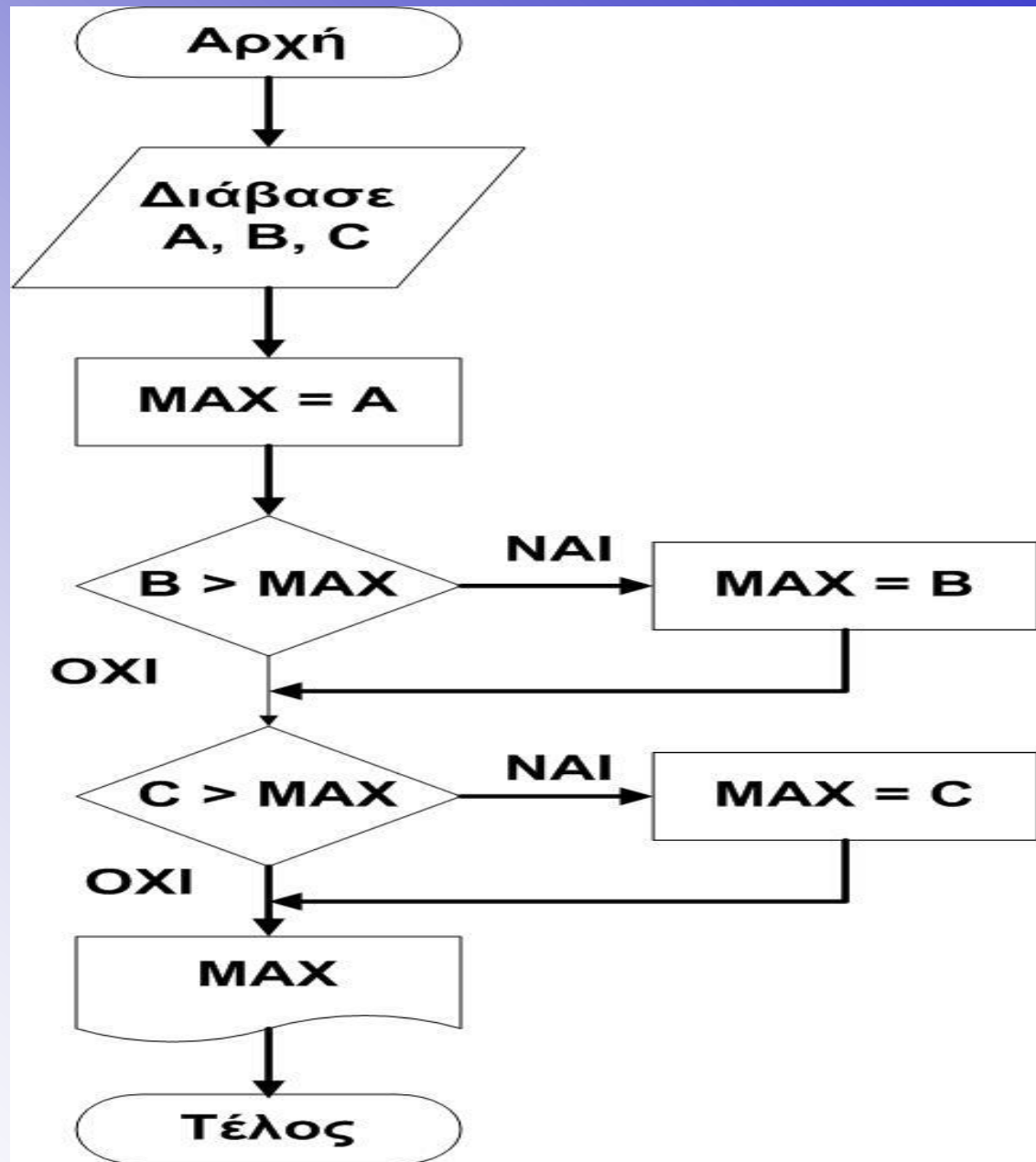
3: Αν $B > MAX$ Τότε Θέσε **MAX = B**

4: Αν $C > MAX$ Τότε Θέσε **MAX = C**

5: Τύπωσε την τιμή του MAX

6: Τέλος

Πρόβλημα 2:
Ο αλγόριθμος
σε μορφή
διαγράμματος
ροής .



Πρόβλημα 3: Να τυπωθεί η φράση «ΚΑΛΗΜΕΡΑ ΖΩΗ» N φορές

Είσοδος: Ο αριθμός N

Έξοδος: Η φράση “ΚΑΛΗΜΕΡΑ ΖΩΗ!” N φορές

Πρόβλημα 3: Αλγόριθμος σε ψευδοκώδικα

1: Διάβασε N

2: Μηδένισε μετρητή, $I = 0$

3: Όσο ο μετρητής $I < N$

3.1 Αύξησε μετρητή κατά ένα, $I = I + 1$

3.2 Τύπωσε “ΚΑΛΗΜΕΡΑ ΖΩΗ!”

Τέλος Όσο

4: Τέλος

Πρόβλημα 3: Αλγόριθμος σε ψευδοκώδικα

1: Διάβασε N
2: Μηδένισε μετρητή I
3: Όσο I < N
 3.1 Αύξησε μετρητή I
 3.2 Τύπωσε ΚΑΛΗΜΕΡΑ ΖΩΗ!
Τέλος Όσο
4: Τέλος

Εκτέλεση αλγορίθμου για **N=5**

ΚΑΛΗΜΕΡΑ ΖΩΗ!

ΚΑΛΗΜΕΡΑ ΖΩΗ!

ΚΑΛΗΜΕΡΑ ΖΩΗ!

ΚΑΛΗΜΕΡΑ ΖΩΗ!

ΚΑΛΗΜΕΡΑ ΖΩΗ!

Πρόβλημα 3: Βελτίωση αλγορίθμου

1: Διάβασε N

2: Όσο $N \leq 0$

2.1 Τύπωσε “Δώσε θετικό πλήθος N ”

2.2 Διάβασε N

Τέλος Όσο

3: Μηδένισε μετρητή, $I = 0$

4: Όσο $I < N$

4.1 Αύξησε μετρητή κατά ένα, $I = I + 1$

4.2 Τύπωσε “ΚΑΛΗΜΕΡΑ ΖΩΗ!”

Τέλος Όσο

5: Τέλος

Πρόβλημα 4: Για N τυχαίους αριθμούς που θα διαβαστούν από το πληκτρολόγιο να υπολογιστεί το άθροισμα, το γινόμενο και ο μέσος όρος τους.

Είσοδος: Οι N αριθμοί

Έξοδος: Το άθροισμα (S), το γινόμενο (G) και ο μέσος όρος τους (MO).

Πρόβλημα 4:

1: Διάβασε N

2: Όσο $N \leq 0$

2.1 Τύπωσε “Δώσε θετικό πλήθος N”

2.2 Διάβασε N

Τέλος Όσο

3: Μηδένισε μετρητή αριθμών, $J = 0$

4: Θέσε αρχικές τιμές στα S και G, $S=0$, $G=1$

5: Όσο $J < N$

5.1 Αύξησε μετρητή κατά ένα, $J = J + 1$

5.2 Τύπωσε “Δώσε νέο αριθμό X”

5.3 Διάβασε X

5.4 Υπολόγισε $S+X$ και φύλαξε το αποτέλεσμα στο S

5.5 Υπολόγισε $G \cdot X$ και φύλαξε το αποτέλεσμα στο G

Τέλος Όσο

6. Υπολόγισε μέσο όρο, $MO=S/N$

7: Τύπωσε S, G, MO

8: Τέλος

Πρόβλημα 4:

Το διάγραμμα
ροής του
αλγορίθμου.

