



ΑΕΝ ΗΠΕΙΡΟΥ ΠΛΗΡΟΦΟΡΙΚΗ Ι

ΣΤΟΙΧΕΙΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

ΣΤΟΙΧΕΙΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

- Γλώσσες προγραμματισμού
- Αλγόριθμος
- Διάγραμμα ροής

ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

- Για να μπορέσουμε να επικοινωνήσουμε με έναν υπολογιστή, να τον ελέγξουμε και να εκτελέσουμε τις επιθυμητές ενέργειες, χρειαζόμαστε έναν **κώδικα επικοινωνίας**.
- Έτσι, δημιουργήθηκαν οι Γλώσσες Προγραμματισμού (Programming Languages).

ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

- Πρόκειται για **τεχνητές γλώσσες** που έχουν τους δικούς τους συντακτικούς και εννοιολογικούς κανόνες και χρησιμοποιούνται για την οργάνωση και διαχείριση πληροφοριών και τη διατύπωση αλγορίθμων.
- Επομένως, τα **προγράμματα** που εκτελούνται από έναν υπολογιστή είναι ουσιαστικά ένα σύνολο εντολών γραμμένες σε κάποια από τις γλώσσες προγραμματισμού που υπάρχουν.

ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

- Κάθε γλώσσα έχει τις δικές της **προδιαγραφές** οι οποίες αφορούν στο **λεξιλόγιο**, στο **συντακτικό** και στη **σημασιολογία** της.
- Ανάλογα με τον **σκοπό** για τον οποίο γράφτηκε μια γλώσσα και το μέσο όπου απευθύνεται, οι γλώσσες προγραμματισμού διακρίνονται στις τρεις παρακάτω κατηγορίες:
 - Γλώσσα μηχανής
 - Γλώσσα χαμηλού επιπέδου
 - Γλώσσα υψηλού επιπέδου

ΓΛΩΣΣΑ ΜΗΧΑΝΉΣ

- Περιλαμβάνει εντολές γραμμένες σε **μορφή ακολουθιών bit** (συμβολοσειρές από 0 και 1) και γίνεται άμεσα αντιληπτή από τον επεξεργαστή του υπολογιστή.
- Σε οποιαδήποτε γλώσσα και αν είναι γραμμένο ένα πρόγραμμα, τελικά θα μεταφραστεί μέσω ενός **συμβολομεταφραστή** (assembler) σε γλώσσα μηχανής για να μπορέσει να γίνει **κατανοητό** από την κεντρική μονάδα επεξεργασίας.
- Πρόκειται για μία πολύ **δυσνόητη** για τον άνθρωπο γλώσσα και ο προγραμματισμός σε αυτήν απαιτεί εξαιρετικά καλές γνώσεις της γλώσσας, αλλά και της λειτουργίας του υπολογιστή.

ΓΛΩΣΣΑ ΜΗΧΑΝΉΣ

Assembly Language	Machine Code
add \$t1, \$t2, \$t3	04CB: 0000 0100 1100 1011
addi \$t2, \$t3, 60	16BC: 0001 0110 1011 1100
and \$t3, \$t1, \$t2	0299: 0000 0010 1001 1001
andi \$t3, \$t1, 5	22C5: 0010 0010 1100 0101
beq \$t1, \$t2, 4	3444: 0011 0100 0100 0100
bne \$t1, \$t2, 4	4444: 0100 0100 0100 0100
j 0x50	F032: 1111 0000 0011 0010
lw \$t1, 16(\$s1)	5A50: 0101 1010 0101 0000
nop	0005: 0000 0000 0000 0101
nor \$t3, \$t1, \$t2	029E: 0000 0010 1001 1110
or \$t3, \$t1, \$t2	029A: 0000 0010 1001 1010
ori \$t3, \$t1, 10	62CA: 0110 0010 1100 1010
srl \$t2, \$t1, 2	0455: 0000 0100 0101 0101
srl \$t2, \$t1, 1	0457: 0000 0100 0101 0111
sw \$t1, 16(\$t0)	7050: 0111 0000 0101 0000
sub \$t2, \$t1, \$t0	0214: 0000 0010 0001 0100

ΓΛΩΣΣΑ ΧΑΜΗΛΟΎ ΕΠΙΠΈΔΟΥ

- Βρίσκεται πολύ κοντά στη γλώσσα μηχανής, αλλά για τις εντολές εδώ χρησιμοποιούνται **συμβολικά ονόματα** και όχι συμβολοσειρές από 0 και 1.
- Είναι **άμεσα συνδεδεμένη με το μηχάνημα** για το οποίο γράφτηκε και δεν μπορεί να εκτελεστεί σε κάποιο άλλο, ακόμα και αν είναι του ίδιου κατασκευαστή.
- Δεν χρειάζεται, όμως, μεταγλωττιστή και ο επεξεργαστής για τον οποίο γράφτηκε μπορεί να την τρέξει όπως είναι γραμμένη.

ΓΛΩΣΣΑ ΥΨΗΛΟΎ ΕΠΙΠΈΔΟΥ:

- Πρόκειται για τις γλώσσες που επιτρέπουν τη **μεταφορά** ενός προγράμματος από έναν υπολογιστή σε έναν άλλο.
- Είναι πιο κοντά στη **φυσική γλώσσα** του ανθρώπου και γι'αυτό είναι εύκολα **κατανοητή** από τους προγραμματιστές.
- Για την εκτέλεση των προγραμμάτων τους, όμως, απαιτείται η χρήση **μεταγλωττιστή** (interpreter, compiler) για τη μετάφραση σε γλώσσα μηχανής.

ΓΛΩΣΣΑ ΥΨΗΛΟΎ ΕΠΙΠΈΔΟΥ:

- Αξίζει να σημειωθεί ότι με την εξέλιξη του προγραμματισμού και την ανάπτυξη γραφικού περιβάλλοντος εργασίας, μπορούμε να χρησιμοποιούμε και γλώσσες προγραμματισμού σε γραφικό περιβάλλον (π.χ. Visual C++, Visual Basic).
- Έτσι, σχεδιάζουμε όλο το περιβάλλον μιας εφαρμογής (μενού, κουμπιά κ.λπ.), και οι εντολές εκτελούνται επιλέγοντάς τες απλά από το μενού.

ΑΛΓΟΡΙΘΜΟΣ

- Ο αλγόριθμος είναι μία **καθορισμένη σειρά** σαφών εντολών, με αρχή, μέση και τέλος, που επιλύει **ορθά** ένα συγκεκριμένο πρόβλημα σε **πεπερασμένο χρόνο**.
- Ένας αλγόριθμος θα πρέπει να είναι **αποτελεσματικός** και να έχει τη δυνατότητα **επέκτασης**.

ΑΛΓΟΡΙΘΜΟΣ

- Για να δημιουργήσουμε έναν αλγόριθμο, ακολουθούμε μία **συγκεκριμένη σειρά βημάτων**:

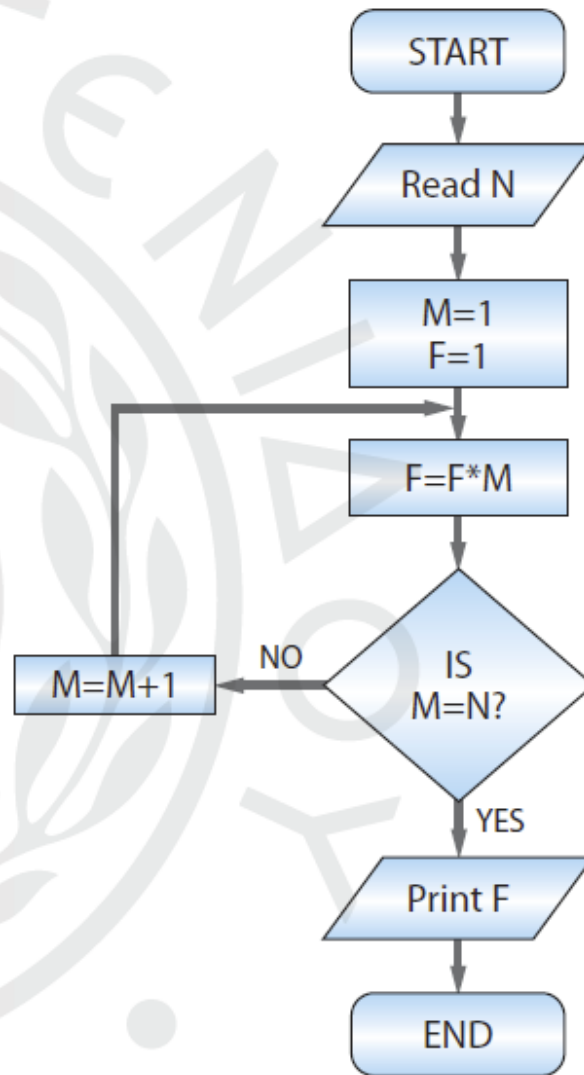
1. Διατυπώνουμε το πρόβλημα
2. Κατανοούμε το πρόβλημα
3. Λύνουμε το πρόβλημα
4. Διατυπώνουμε τον αλγόριθμο και
5. Ελέγχουμε τη λύση.

Έτσι, έχουμε **είσοδο** δεδομένων, **επεξεργασία** και **έξοδο** αποτελεσμάτων.

ΑΛΓΟΡΙΘΜΟΣ

- Για να υλοποιηθεί ένας αλγόριθμος στον υπολογιστή, χρειάζεται να γραφούν οι απαραίτητες **εντολές** με τη χρήση μιας γλώσσας προγραμματισμού.
- Οι εντολές αυτές σε συνδυασμό με τα απαραίτητα δεδομένα (data), αποτελούν ένα ολοκληρωμένο πλέον σύνολο, **το πρόγραμμα** (program).

ΔΙΑΓΡΑΜΜΑ ΡΟΪΗΣ



Σχ. 1.25

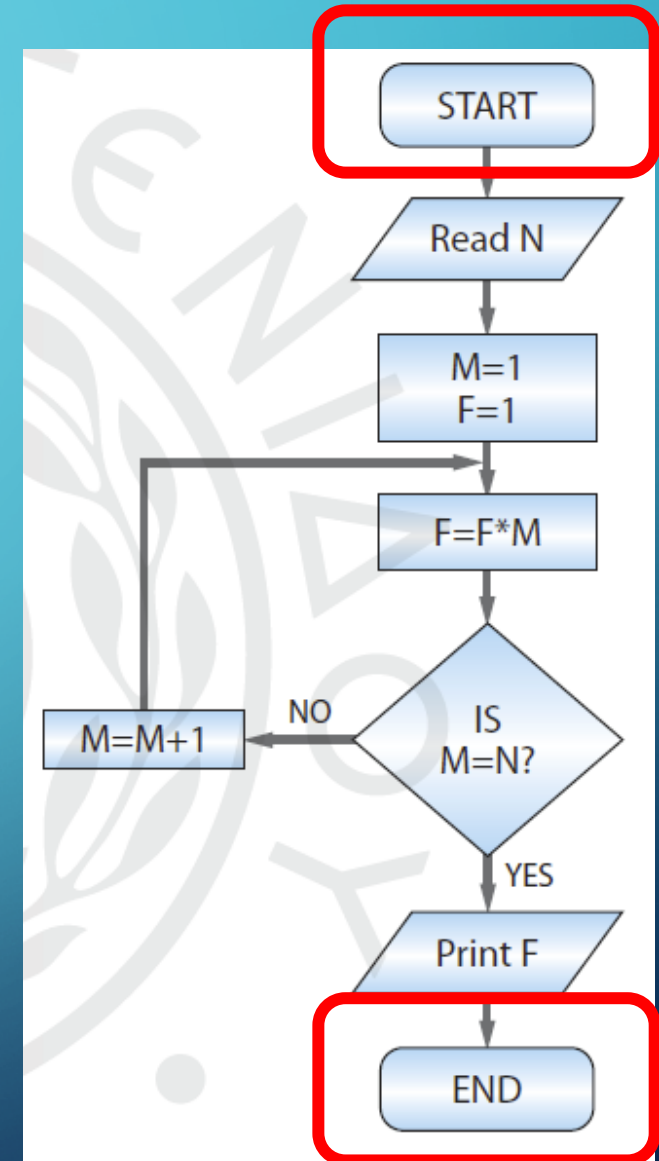
Διάγραμμα ροής για τον υπολογισμό
του N παραγοντικό ($N!$)

ΔΙΑΓΡΑΜΜΑ ΡΟΪΗΣ

- Διάγραμμα ροής (flowchart) είναι ένα διάγραμμα που **αναπαριστά έναν αλγόριθμο ή μία διαδικασία**, δείχνοντας τα βήματα ως διάφορα σχήματα που συνδέονται μεταξύ τους με βέλη.
- Αυτή η διαγραμματική παρουσίαση μπορεί να δώσει λύση βήμα προς βήμα σε ένα πρόβλημα. **Τα δεδομένα αναπαρίστανται σε σχήματα** (συνήθως παραλληλόγραμμα) και **τα βέλη δείχνουν τη ροή των δεδομένων**.
- Τα διαγράμματα ροής χρησιμοποιούνται στην ανάλυση, στην τεκμηρίωση και στον έλεγχο μίας διαδικασίας ή ενός προγράμματος.

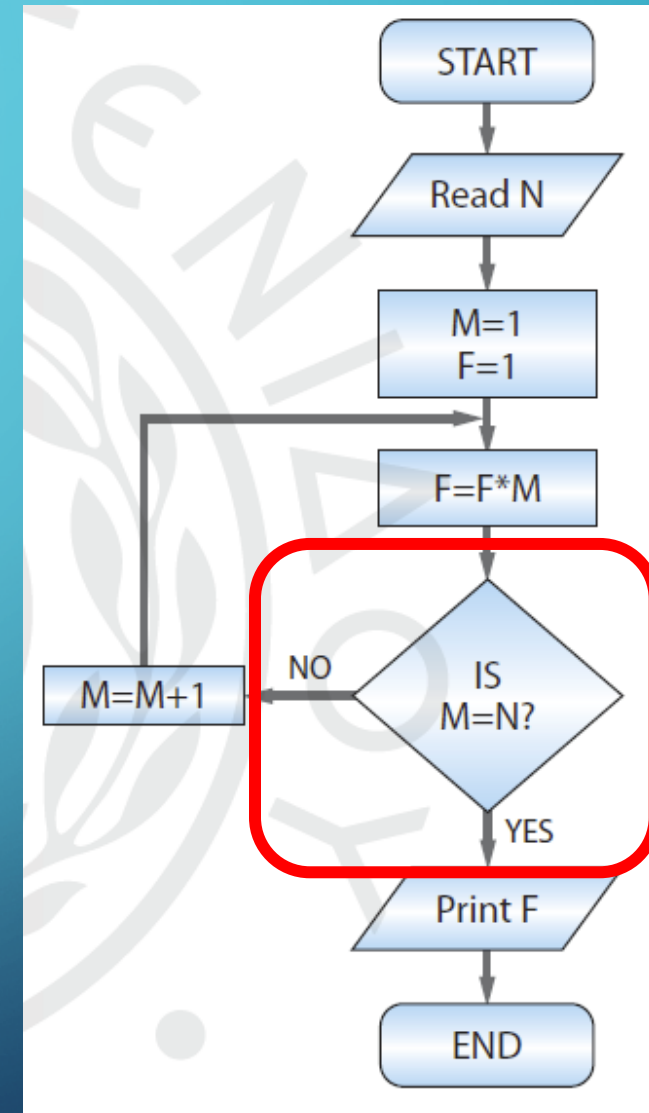
ΔΙΑΓΡΑΜΜΑ ΡΟΪΗΣ

Η **έλλειψη** που δηλώνει την αρχή και το τέλος του κάθε αλγορίθμου.



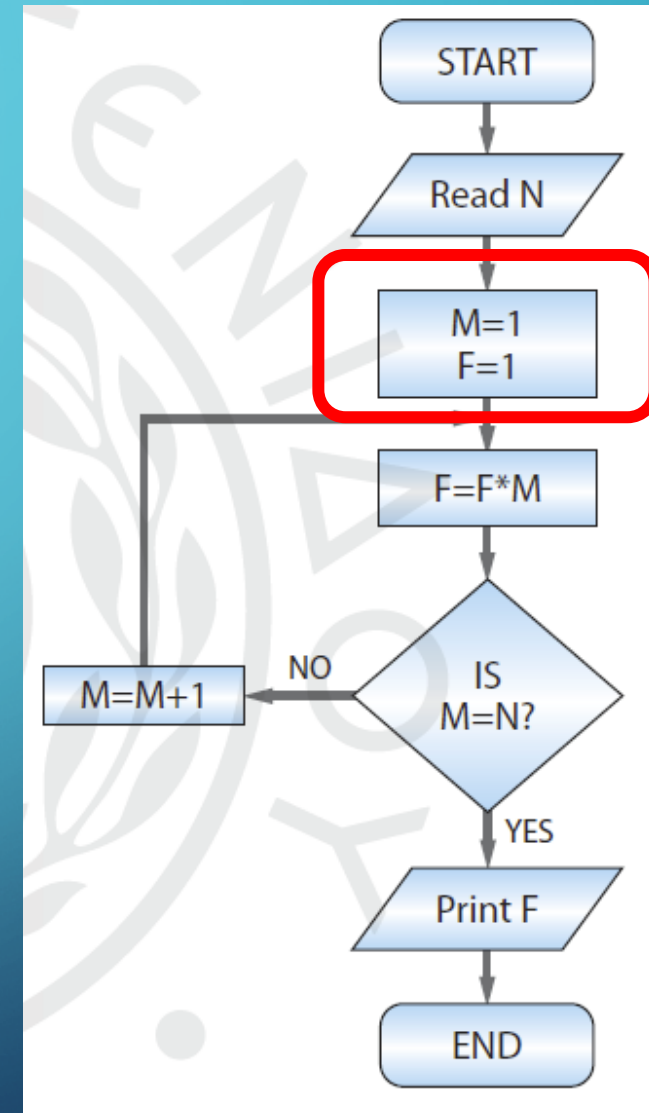
ΔΙΑΓΡΑΜΜΑ ΡΟΪΗΣ

Ο **ρόμβος** που δηλώνει μία ερώτηση με δύο ή περισσότερες εξόδους ως απάντηση.



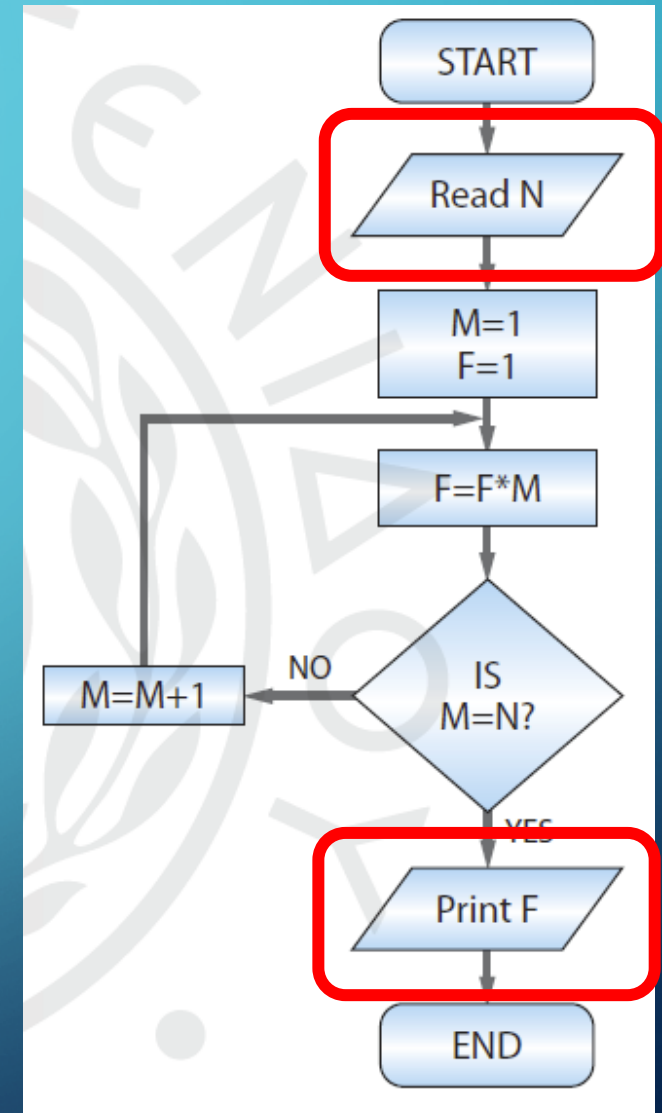
ΔΙΑΓΡΑΜΜΑ ΡΟΪΗΣ

Το **ορθογώνιο** που δηλώνει την εκτέλεση μίας ή περισσότερων πράξεων.



ΔΙΑΓΡΑΜΜΑ ΡΟΪΗΣ

Το **πλάγιο παραλληλόγραμμα** που δηλώνει είσοδο ή έξοδο στοιχείων.



ΔΙΑΓΡΑΜΜΑ ΡΟΪΗΣ

- Στην περίπτωση που ένα βέλος καταλήξει σε ένα προηγούμενο στάδιο του αλγορίθμου, έχουμε μια **επαναληπτική διαδικασία (loop)**.
- Σε κάθε περίπτωση οι διαδικασίες ξεκινούν να εκτελούνται από **πάνω προς τα κάτω και από δεξιά προς τα αριστερά**.