

**2.**

Βασικές Έννοιες Αλγορίθμων

### Εισαγωγή



Αρχικά εξηγείται ο όρος αλγόριθμος και παραθέτονται τα σπουδαιότερα κριτήρια που πρέπει να πληρεί κάθε αλγόριθμος. Στη συνέχεια, η σπουδαιότητα των αλγορίθμων συνδυάζεται με την εξέλιξη της επιστήμης της Πληροφορικής. Η περιγραφή και αναπαράσταση των αλγορίθμων δίνεται αναλυτικά με χρήση των μεθόδων αναπαράστασης ελεύθερου κειμένου, διαγραμμάτων ροής, φυσικής γλώσσας και κωδικοποίησης με πρόγραμμα. Τα προγράμματα παρουσιάζονται με τη μορφή ψευδογλώσσας, που ορίζεται και τυποποιείται σε ένα σύνολο εντολών και προγραμματιστικών ακολουθιακών ενοτήτων. Στη συνέχεια, δίνονται παραδείγματα όπου εξετάζονται οι διάφορες συνιστώσες ενός αλγόριθμου, δηλαδή οι απαραίτητες εντολές που στηρίζουν το 'κτίσιμο' ενός αλγόριθμου. Συγκεκριμένα, παρουσιάζονται η δομή ακολουθίας, η δομή της επιλογής, οι επαναληπτικές διαδικασίες, οι διαδικασίες πολλαπλών επιλογών και οι εμφωλισμένες διαδικασίες. Για κάθε τύπο συνιστώσας δίνονται αναλυτικά παραδείγματα σε φυσική γλώσσα, σε ακολουθία διαδοχικών βημάτων και σε μορφή διαγραμμάτων ροής. Στο τέλος του κεφαλαίου παρουσιάζεται η ανάπτυξη και η αλγοριθμική προσέγγιση για την επίλυση ενός συνθετότερου προβλήματος, του προβλήματος του 'πολλαπλασιασμού αλά ρωσικά', όπου γίνεται χρήση και συνδυασμός αλγοριθμικών συνιστωσών.

### Διδακτικοί στόχοι



Στόχοι του κεφαλαίου αυτού είναι οι μαθητές:

- ⇒ να διατυπώνουν την έννοια του αλγορίθμου,
- ⇒ να αιτιολογούν τη σπουδαιότητα των αλγορίθμων,
- ⇒ να τεκμηριώνουν την αναγκαιότητα της αλγοριθμικής προσέγγισης κατά τη διαδικασία επίλυσης προβλημάτων,
- ⇒ να εφαρμόζουν τυποποιημένη επίλυση με αλγοριθμικές διαδικασίες,
- ⇒ να μπορούν να σχεδιάζουν αλγόριθμους με χρήση συγκεκριμένων τεχνικών.

### Προερωτήσεις



- ✓ Γνωρίζεις τι είναι αλγοριθμική προσέγγιση;
- ✓ Ξέρεις ότι ήδη έχεις χρησιμοποιήσει πολλούς αλγόριθμους;
- ✓ Γνωρίζεις, αν ο πολλαπλασιασμός δύο αριθμών μπορεί να γίνει με άλλο τρόπο;
- ✓ Τι θα κάνεις για να βρεις το άθροισμα  $3+6+9+\dots+999$ ;

## 2.1 Τι είναι αλγόριθμος

Η θεωρία των αλγορίθμων έχει μεγάλη παράδοση και η ηλικία μερικών αλγορίθμων αριθμεί χιλιάδες χρόνια, όπως για παράδειγμα ο αλγόριθμος του Ευκλείδη για την εύρεση του μέγιστου κοινού διαιρέτη δύο αριθμών ή το λεγόμενο κόσκινο του Ερατοσθένη για την εύρεση των πρώτων αριθμών από 1 ως  $n$ . Σήμερα το πεδίο της Θεωρίας Αλγορίθμων είναι ένα ιδιαίτερα ευρύ και πλούσιο πεδίο. Πληθώρα συγγραμμάτων έχει εμφανισθεί στη βιβλιογραφία, ενώ συνεχίζεται η περαιτέρω εμβάθυνση σε νέα σύγχρονα προβλήματα. Οι περισσότεροι από τους αλγορίθμους που συνήθως εξετάζονται στα σχετικά βιβλία έχουν προταθεί τα τελευταία 25 χρόνια, όση περίπου είναι και η ηλικία της Πληροφορικής ως μίας νέας αυθύπαρκτης επιστήμης.

### Ιστορικό σημείωμα

Η λέξη *αλγόριθμος* (algorithm) προέρχεται από μια μελέτη του Πέρση μαθηματικού Abu Ja'far Mohammed ibn Musa al Khowarizmi, που έζησε περί το 825 μ.Χ. Πέντε αιώνες αργότερα η μελέτη αυτή μεταφράστηκε στα λατινικά και άρχισε με τη φράση "Algoritmi dixit ..." (ο αλγόριθμος λέει ...). Η μελέτη του al Khowarizmi υπήρξε η πρώτη πλήρης πραγματεία άλγεβρας (όρος που και αυτός προέρχεται από το αραβικό al-jabr=αποκατάσταση), γιατί ένας από τους σκοπούς της άλγεβρας είναι και η αποκατάσταση της ισότητας μέσα σε μια εξίσωση. Ο όρος αλγόριθμος επέζησε επί χίλια χρόνια ως σπάνιος όρος, που σήμαινε κάτι σαν "συστηματική διαδικασία αριθμητικών χειρισμών". Τη σημερινή του αξία απέκτησε από την αρχή του 20ού αιώνα με την ανάπτυξη της ομώνυμης θεωρίας και φυσικά με την επικαιρότητα των ηλεκτρονικών υπολογιστών.



Ο όρος αλγόριθμος, λοιπόν, χρησιμοποιείται για να δηλώσει μεθόδους που εφαρμόζονται για την επίλυση προβλημάτων. Ωστόσο, ένας πιο αυστηρός ορισμός της έννοιας αυτής είναι ο εξής.

**Ορισμός:** Αλγόριθμος είναι μια πεπερασμένη σειρά ενεργειών, αυστηρά καθορισμένων και εκτελέσιμων σε πεπερασμένο χρόνο, που στοχεύουν στην επίλυση ενός προβλήματος.



Κάθε αλγόριθμος απαραίτητα ικανοποιεί τα επόμενα κριτήρια.

➡ **Είσοδος** (input). Καμία, μία ή περισσότερες τιμές δεδομένων πρέπει να δίνονται ως είσοδοι στον αλγόριθμο. Η περίπτωση που δεν δίνονται τιμές δεδομένων εμφανίζεται, όταν ο αλγόριθμος δημιουργεί και επεξερ-

γάζεται κάποιες πρωτογενείς τιμές με τη βοήθεια συναρτήσεων παραγωγής τυχαίων αριθμών ή με τη βοήθεια άλλων απλών εντολών.

- ⇒ **Έξοδος** (output). Ο αλγόριθμος πρέπει να δημιουργεί τουλάχιστον μία τιμή δεδομένων ως αποτέλεσμα προς το χρήστη ή προς έναν άλλο αλγόριθμο.
- ⇒ **Καθοριστικότητα** (definiteness). Κάθε εντολή πρέπει να καθορίζεται χωρίς καμία αμφιβολία για τον τρόπο εκτέλεσής της. Λόγου χάριν, μία εντολή διαίρεσης πρέπει να θεωρεί και την περίπτωση, όπου ο διαιρέτης λαμβάνει μηδενική τιμή.
- ⇒ **Περατότητα** (finiteness). Ο αλγόριθμος να τελειώνει μετά από πεπερασμένα βήματα εκτέλεσης των εντολών του. Μία διαδικασία που δεν τελειώνει μετά από ένα συγκεκριμένο αριθμό βημάτων δεν αποτελεί αλγόριθμο, αλλά λέγεται απλά *υπολογιστική διαδικασία* (computational procedure).
- ⇒ **Αποτελεσματικότητα** (effectiveness). Κάθε μεμονωμένη εντολή του αλγορίθμου να είναι απλή. Αυτό σημαίνει ότι μία εντολή δεν αρκεί να έχει ορισθεί, αλλά πρέπει να είναι και εκτελέσιμη.

Η έννοια του αλγορίθμου δεν συνδέεται αποκλειστικά και μόνο με προβλήματα της Πληροφορικής. Ας θεωρήσουμε, για παράδειγμα, ότι θέλουμε να γευματίσουμε και επομένως πρέπει να εκτελέσουμε τις επόμενες ενέργειες:

- ✓ να συγκεντρώσουμε τα υλικά,
- ✓ να προετοιμάσουμε τα σκεύη μαγειρικής,
- ✓ να παρασκευάσουμε το φαγητό,
- ✓ να ετοιμάσουμε τη σαλάτα,
- ✓ να στρώσουμε το τραπέζι,
- ✓ να γευματίσουμε,
- ✓ να καθαρίσουμε το τραπέζι, και
- ✓ να πλύνουμε τα πιάτα και τα κουζινικά.

Είναι ευνόητο ότι η προηγούμενη αλληλουχία των ενεργειών οδηγεί στο επιθυμητό αποτέλεσμα. Βέβαια, αυτή η αλληλουχία δεν είναι η μοναδική για την επίτευξη του σκοπού, αφού, για παράδειγμα, μπορούμε πρώτα να ετοιμάσουμε τη σαλάτα και μετά να παρασκευάσουμε το φαγητό, ενώ ακόμη μπορούμε πρώτα να πλύνουμε τα πιάτα και μετά να καθαρίσουμε το τραπέζι. Ωστόσο, το παράδειγμα θέλει να δείξει, ότι η θεώρηση μίας σύνθε-

της εργασίας με διακριτά βήματα που εκτελούνται διαδοχικά, είναι ένας πολύ χρήσιμος και πρακτικός τρόπος σκέψης για την επίλυση πολλών (αν όχι όλων) προβλημάτων.

## 2.2 Σπουδαιότητα αλγορίθμων

Η έννοια του αλγόριθμου είναι θεμελιώδης για την επιστήμη της Πληροφορικής. Η μελέτη των αλγορίθμων είναι πολύ ενδιαφέρουσα, γιατί είναι η πρώτη ύλη για τη μελέτη και εμβάθυνση, αν όχι σε όλες, τουλάχιστον σε πάρα πολλές γνωστικές περιοχές της επιστήμης αυτής.

Η Πληροφορική, λοιπόν, μπορεί να ορισθεί ως η επιστήμη που μελετά τους αλγορίθμους από τις ακόλουθες σκοπιές:

- ⇒ **Υλικού** (hardware). Η ταχύτητα εκτέλεσης ενός αλγορίθμου επηρεάζεται από τις διάφορες τεχνολογίες υλικού, δηλαδή από τον τρόπο που είναι δομημένα σε μία ενιαία αρχιτεκτονική τα διάφορα συστατικά του υπολογιστή (δηλαδή ανάλογα με το αν ο υπολογιστής έχει κρυφή μνήμη και πόση, ανάλογα με την ταχύτητα της κύριας και δευτερεύουσας μνήμης κ.ο.κ.).
- ⇒ **Γλωσσών Προγραμματισμού** (programming languages). Το είδος της γλώσσας προγραμματισμού που χρησιμοποιείται (δηλαδή, χαμηλότερου ή υψηλότερου επιπέδου) αλλάζει τη δομή και τον αριθμό των εντολών ενός αλγορίθμου. Γενικά μία γλώσσα που είναι χαμηλότερου επιπέδου (όπως η assembly ή η γλώσσα C) είναι ταχύτερη από μία άλλη γλώσσα που είναι υψηλότερου επιπέδου (όπως η Basic ή Pascal). Ακόμη, σημειώνεται ότι διαφορές συναντώνται μεταξύ των γλωσσών σε σχέση με το πότε εμφανίσθηκαν. Για παράδειγμα, παλαιότερα μερικές γλώσσες προγραμματισμού δεν υποστήριζαν την αναδρομή (έννοια που θα εξετάσουμε σε βάθος αργότερα).
- ⇒ **Θεωρητική** (theoretical). Το ερώτημα που συχνά τίθεται είναι, αν πράγματι υπάρχει ή όχι κάποιος αποδοτικός αλγόριθμος για την επίλυση ενός προβλήματος. Η εξέταση αυτού του ερωτήματος είναι δύσκολο να σχολιασθεί στα πλαίσια του βιβλίου αυτού, επειδή απαιτεί μεγάλη θεωρητική κατάρτιση. Ωστόσο η προσέγγιση αυτή είναι ιδιαίτερα σημαντική, γιατί προσδιορίζει τα όρια της λύσης που θα βρεθεί σε σχέση με ένα συγκεκριμένο πρόβλημα.
- ⇒ **Αναλυτική** (analytical). Μελετώνται οι υπολογιστικοί πόροι (computer resources) που απαιτούνται από έναν αλγόριθμο, όπως για παράδειγμα

μα το μέγεθος της κύριας και της δευτερεύουσας μνήμης, ο χρόνος για λειτουργίες CPU και για λειτουργίες εισόδου/εξόδου κ.λπ. Το αντικείμενο αυτό θα εξηγηθεί πληρέστερα στο Κεφάλαιο 5.

## 2.3 Περιγραφή και αναπαράσταση αλγορίθμων

Στη βιβλιογραφία συναντώνται διάφοροι τρόποι αναπαράστασης ενός αλγορίθμου:

- ⇒ με **ελεύθερο κείμενο** (free text), που αποτελεί τον πιο ανεπεξέργαστο και αδόμητο τρόπο παρουσίασης αλγορίθμου. Έτσι εγκυμονεί τον κίνδυνο ότι μπορεί εύκολα να οδηγήσει σε μη εκτελέσιμη παρουσίαση παραβιάζοντας το τελευταίο χαρακτηριστικό των αλγορίθμων, δηλαδή την αποτελεσματικότητα.
- ⇒ με **διαγραμματικές τεχνικές** (diagramming techniques), που συνιστούν ένα γραφικό τρόπο παρουσίασης του αλγορίθμου. Από τις διάφορες διαγραμματικές τεχνικές που έχουν επινοηθεί, η πιο παλιά και η πιο γνωστή ίσως, είναι το διάγραμμα ροής (flow chart). Ωστόσο η χρήση διαγραμμάτων ροής για την παρουσίαση αλγορίθμων δεν αποτελεί την καλύτερη λύση, γι'αυτό και εμφανίζονται όλο και σπανιότερα στη βιβλιογραφία και στην πράξη.
- ⇒ με **φυσική γλώσσα** (natural language) κατά βήματα. Στην περίπτωση αυτή χρειάζεται προσοχή, γιατί μπορεί να παραβιασθεί το τρίτο βασικό χαρακτηριστικό ενός αλγορίθμου, όπως προσδιορίσθηκε προηγουμένως, δηλαδή το κριτήριο του καθορισμού.
- ⇒ με **κωδικοποίηση** (coding), δηλαδή με ένα πρόγραμμα που όταν εκτελεσθεί θα δώσει τα ίδια αποτελέσματα με τον αλγόριθμο.

Όλοι οι αλγόριθμοι του βιβλίου αυτού είναι κωδικοποιημένοι σε μία υποθετική δομημένη ψευδογλώσσα, ωστόσο οι περισσότεροι από αυτούς μπορούν εύκολα σχετικά να προγραμματισθούν σε οποιαδήποτε γλώσσα προγραμματισμού.

## 2.4 Βασικές συνιστώσες/εντολές ενός αλγορίθμου

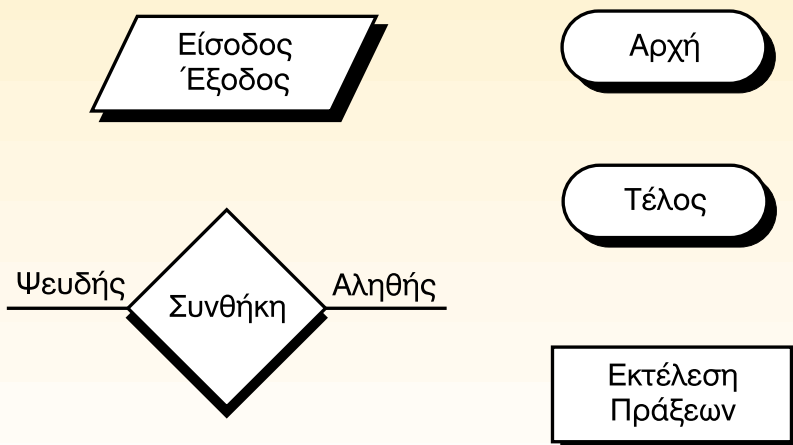
Στη συνέχεια δίνονται παραδείγματα αλγορίθμων όπου εξετάζονται οι

## Σύμβολα διαγράμματος ροής

Ένα διάγραμμα ροής αποτελείται από ένα σύνολο γεωμετρικών σχημάτων, όπου το καθένα δηλώνει μία συγκεκριμένη ενέργεια ή λειτουργία. Τα γεωμετρικά σχήματα ενώνονται μεταξύ τους με βέλη, που δηλώνουν τη σειρά εκτέλεσης των ενεργειών αυτών. Τα κυριότερα χρησιμοποιούμενα γεωμετρικά σχήματα είναι τα εξής:

- ⇒ *έλλειψη*, που δηλώνει την αρχή και το τέλος του κάθε αλγορίθμου,
- ⇒ *ρόμβος*, που δηλώνει μία ερώτηση με δύο ή περισσότερες εξόδους για απάντηση,
- ⇒ *ορθογώνιο*, που δηλώνει την εκτέλεση μίας ή περισσότερων πράξεων, και
- ⇒ *πλάγιο παραλληλόγραμμο*, που δηλώνει είσοδο ή έξοδο στοιχείων. Πολλές φορές το σχήμα αυτό μπορεί να διαφοροποιείται προκειμένου να προσδιορίζεται και το είδος της συσκευής απ' όπου γίνεται η είσοδος ή η έξοδος.

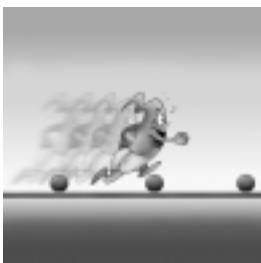
Το επόμενο σχήμα αποτυπώνει όλα αυτά τα σύμβολα..



Μερικά από τα χρησιμοποιούμενα γεωμετρικά σύμβολα στα διαγράμματα ροής.

διάφορες συνιστώσες ενός αλγορίθμου, δηλαδή οι απαραίτητες εντολές ξεκινώντας από τις απλούστερες και προχωρώντας προς τις συνθετότερες. Πιο συγκεκριμένα θα εξετασθούν περιπτώσεις σειριακών εντολών, αναθέσεων τιμών, επιλογής με βάση κριτήρια, διαδικασιών επανάληψης, ενεργειών πολλαπλών επιλογών καθώς και συνδυασμό εμφωλευμένων περιπτώσεων. Για κάθε περίπτωση παρουσιάζονται σχετικά παραδείγματα με την εκφώνηση (σε φυσική γλώσσα), την παρουσίαση των βημάτων που πρέπει να ακολουθηθούν καθώς και το αντίστοιχο διάγραμμα ροής.

### 2.4.1 Δομή ακολουθίας



Η ακολουθιακή δομή εντολών (σειριακών βημάτων) χρησιμοποιείται πρακτικά για την αντιμετώπιση απλών προβλημάτων, όπου είναι δεδομένη η σειρά εκτέλεσης ενός συνόλου ενεργειών. Ένα απλό παράδειγμα από την καθημερινή ζωή είναι η ακολουθία οδηγιών μίας συνταγής μαγειρικής με στόχο την κατασκευή ενός φαγητού. Τα βήματα και οι ποσότητες που πρέπει να ακολουθηθούν είναι συγκεκριμένα και οι οδηγίες απόλυτα καθορισμένες και σαφείς. Το παράδειγμα που ακολουθεί παρουσιάζει ένα απλό πρόβλημα που επιλύεται με σειριακή εκτέλεση εντολών.

#### Παράδειγμα 1. Ανάγνωση και εκτύπωση αριθμών

**Να διαβασθούν δύο αριθμοί, να υπολογισθεί και να εκτυπωθεί το άθροισμά τους.**



Διάβασε = εκτελεστέα εντολή

Αλγόριθμος = δηλωτική εντολή

Από την εκφώνηση προκύπτει αμέσως ο επόμενος αλγόριθμος

**Αλγόριθμος** Παράδειγμα\_1

**Διάβασε** a

**Διάβασε** b

$c \leftarrow a + b$

**Εκτύπωσε** c

**Τέλος** Παράδειγμα\_1

Ένας αλγόριθμος διατυπωμένος σε ψευδογλώσσα αρχίζει πάντα με τη λέξη *Αλγόριθμος* συνοδευόμενη με το όνομα του αλγορίθμου και τελειώνει με τη λέξη *Τέλος* συνοδευόμενη επίσης με το όνομα του αλγορίθμου. Η πρώτη ενέργεια που γίνεται, είναι η εισαγωγή των δεδομένων. Αυτό επιτυγχάνεται με τη χρήση του ρήματος *Διαβάζω* σε προστακτική. Η λέξη *Διάβασε* συνοδεύεται με το όνομα μίας ή περισσότερων μεταβλητών, όπως η a και εννοείται ότι μετά την ολοκλήρωση της ενέργειας αυτή, η μεταβλητή a θα έχει λάβει κάποια αριθμητική τιμή ως περιεχόμενο. Κάθε μία λέξη της



**Σταθερές** (constants). Με τον όρο αυτό αναφερόμαστε σε προκαθορισμένες τιμές που παραμένουν αμετάβλητες σε όλη τη διάρκεια της εκτέλεσης ενός αλγορίθμου. Οι σταθερές διακρίνονται σε

- ⇒ αριθμητικές, π.χ. 123, +5, -1,25
- ⇒ αλφαριθμητικές π.χ. “Τιμή”, “Κατάσταση αποτελεσμάτων”
- ⇒ λογικές που είναι ακριβώς δύο, Αληθής και Ψευδής

**Μεταβλητές** (variables). Μια μεταβλητή είναι ένα γλωσσικό αντικείμενο, που χρησιμοποιείται για να παραστήσει ένα στοιχείο δεδομένου. Στη μεταβλητή εκχωρείται μια τιμή, η οποία μπορεί να αλλάζει κατά τη διάρκεια εκτέλεσης του αλγορίθμου. Ανάλογα με το είδος της τιμής που μπορούν να λάβουν, οι μεταβλητές διακρίνονται σε αριθμητικές, αλφαριθμητικές και λογικές.

**Τελεστές** (operators). Πρόκειται για τα γνωστά σύμβολα που χρησιμοποιούνται στις διάφορες πράξεις. Οι τελεστές διακρίνονται σε αριθμητικούς, λογικούς και συγκριτικούς.

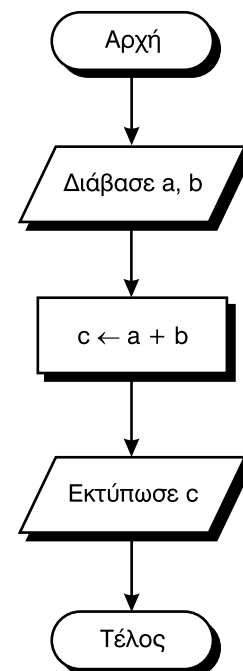
**Εκφράσεις** (expressions). Οι εκφράσεις διαμορφώνονται από τους τελεστές (operands), που είναι σταθερές και μεταβλητές και από τους τελεστές. Η διεργασία αποτίμησης μιας έκφρασης συνίσταται στην απόδοση τιμών στις μεταβλητές και στην εκτέλεση των πράξεων. Η τελική τιμή μιας έκφρασης εξαρτάται από την ιεραρχία των πράξεων και τη χρήση των παρενθέσεων. Μια έκφραση μπορεί να αποτελείται από μια μόνο μεταβλητή ή σταθερά μέχρι μια πολύπλοκη μαθηματική παράσταση.

χρησιμοποιούμενης ψευδογλώσσας, που προσδιορίζει μια σαφή ενέργεια, θα αποκαλείται στο εξής εντολή. Όλες οι εντολές σε έναν αλγόριθμο αποτυπώνονται με διαφορετικό χρώμα από το όνομα του αλγορίθμου και τις διάφορες σταθερές και μεταβλητές.

Μετά την ανάγνωση των τιμών των μεταβλητών  $a$  και  $b$  γίνεται ο υπολογισμός του αθροίσματος με την εντολή:  $c \leftarrow a + b$ . Η εντολή αυτή αποκαλείται εντολή εκχώρησης τιμής. Η γενική μορφή της είναι:

$\text{Μεταβλητή} \leftarrow \text{Έκφραση}$

και η λειτουργία της είναι “γίνονται οι πράξεις στην έκφραση και το αποτέλεσμα αποδίδεται, μεταβιβάζεται, εκχωρείται στη μεταβλητή”. Στην εντολή αυτή χρησιμοποιείται το αριστερό βέλος, προκειμένου να δείχνει τη φορά της εκχώρησης. Ας σημειωθεί ότι δεν πρόκειται για εξίσωση, παρ’όλο που σε άλλα βιβλία μπορεί να χρησιμοποιείται το σύμβολο ίσον “=” για τον



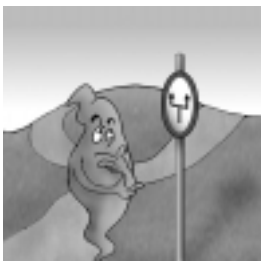
**Σχ. 2.1.** Ο αλγόριθμος του παραδείγματος 1 με διάγραμμα ροής

ίδιο σκοπό. Ας σημειωθεί επίσης ότι οι διάφορες γλώσσες προγραμματισμού χρησιμοποιούν διάφορα σύμβολα για το σκοπό αυτό.

Τέλος ο αλγόριθμος ολοκληρώνεται με την εντολή *Εκτύπωσε*, που αποτυπώνει το τελικό αποτέλεσμα στον εκτυπωτή. Η σύνταξη της εντολής αυτής είναι ανάλογη με αυτή της *Διάβασε*. Εναλλακτικά μπορεί να χρησιμοποιηθεί και η εντολή *Εμφάνισε*, που αποτυπώνει ένα αποτέλεσμα στην οθόνη.

Στον προηγούμενο αλγόριθμο οι μεταβλητές  $a$  και  $b$  είναι τα δεδομένα που αποτελούν την είσοδο, ενώ η μεταβλητή  $c$  αντιπροσωπεύει το αποτέλεσμα, δηλαδή την έξοδο του αλγορίθμου. Επιπλέον, ο αλγόριθμος έχει απολύτως καθορισμένη την κάθε εντολή (*καθοριστικότητα*), τελειώνει μετά από συγκεκριμένο αριθμό βημάτων (*περατότητα*), ενώ κάθε εντολή του είναι ιδιαίτερα σαφής και απλή (*αποτελεσματικότητα*). Επομένως ο αλγόριθμος αυτός πληροί τα κριτήρια που χαρακτηρίζουν τον ορισμό της έννοιας του αλγορίθμου, όπως αυτά περιγράφηκαν στην παράγραφο 2.1.

#### 2.4.2 Δομή Επιλογής



Στην πραγματικότητα πολύ λίγα προβλήματα μπορούν να επιλυθούν με τον προηγούμενο τρόπο της σειριακής/ακολουθιακής δομής ενεργειών. Συνήθως τα προβλήματα έχουν κάποιες ιδιαιτερότητες και δεν ισχύουν τα ίδια βήματα για κάθε περίπτωση. Η πλέον συνηθισμένη περίπτωση είναι να λαμβάνονται κάποιες αποφάσεις με βάση κάποια δεδομένα κριτήρια, που μπορεί να είναι διαφορετικά για κάθε διαφορετικό στιγμιότυπο ενός προβλήματος. Οι καθημερινές απλές μας ενέργειες περιέχουν αυτή τη διαδικασία επιλογής με βάση κάποια κατάσταση. Για παράδειγμα, το πρόβλημα της προετοιμασίας μας για έξοδο σχετίζεται με τις καιρικές συνθήκες. Έτσι λέμε ότι, “αν βρέχει, θα πάρω ομπρέλα, αλλιώς θα πάρω καπέλο”. Η συνθήκη εδώ είναι το “αν βρέχει”, ενώ η απόφαση είναι είτε να πάρω την “ομπρέλα” είτε το “καπέλο” με βάση την “τιμή” της συνθήκης.

Γενικά η διαδικασία της επιλογής περιλαμβάνει τον έλεγχο κάποιας συνθήκης που μπορεί να έχει δύο τιμές (Αληθής ή Ψευδής) και ακολουθεί η απόφαση εκτέλεσης κάποιας ενέργειας με βάση την τιμή της λογικής αυτής συνθήκης. Στη συνέχεια δίνονται δύο παραδείγματα ενεργειών με βάση κάποια συνθήκη επιλογής. Το πρώτο παράδειγμα αφορά στην εκτέλεση κάποιας ενέργειας όταν η συνθήκη είναι Αληθής, ενώ το δεύτερο παράδειγμα αφορά στην εκτέλεση μίας ενέργειας όταν η συνθήκη είναι Αληθής και κάποιας άλλης ενέργειας όταν η συνθήκη είναι Ψευδής.

## Παράδειγμα 2. Σύγκριση αριθμών με απλή επιλογή

**Να διαβαστεί ένας αριθμός και να εκτυπωθεί η απόλυτη τιμή του.**

Όπως είναι γνωστό, η απόλυτη τιμή ενός αριθμού είναι ο ίδιος ο αριθμός, αν αυτός είναι θετικός ή μηδέν και ο αντίθετός του, αν είναι αρνητικός. Έτσι προκειμένου να βρεθεί η απόλυτη τιμή, αρκεί να ελεγχθεί, αν τυχόν ο δεδομένος αριθμός είναι αρνητικός, οπότε στην περίπτωση αυτή πρέπει να βρεθεί ο αντίθετός του. Ο συλλογισμός αυτός οδηγεί στον επόμενο αλγόριθμο.

**Αλγόριθμος** Παράδειγμα\_2

**Διάβαση** a

**Αν**  $a < 0$  **τότε**  $a \leftarrow a * (-1)$

**Εκτύπωση** a

**Τέλος** Παράδειγμα\_2

Στην παράσταση αλγορίθμων με ψευδογλώσσα η επιλογή υλοποιείται με την εντολή **Αν...τότε**. Η σύνταξη της εντολής είναι:

**Αν** συνθήκη **τότε** εντολή

και η λειτουργία της είναι: Αν ισχύει η συνθήκη (δηλαδή αν είναι αληθής), τότε μόνο εκτελείται η εντολή. Σε κάθε περίπτωση εκτελείται στη συνέχεια η εντολή, που ακολουθεί.

Στην εντολή **Αν...τότε** είναι πιθανό, όταν ισχύει η συνθήκη, να απαιτείται η εκτέλεση περισσότερων από μία εντολές. Στην περίπτωση αυτή οι διαδοχικές εντολές γράφονται από κάτω και σε εσοχή, ενώ το σχήμα επιλογής κλείνει με τη λέξη **Τέλος\_αν**. Π.χ

**Αν** συνθήκη **τότε**

εντολή\_1

εντολή\_2

.....

εντολή\_ν

**Τέλος\_αν**

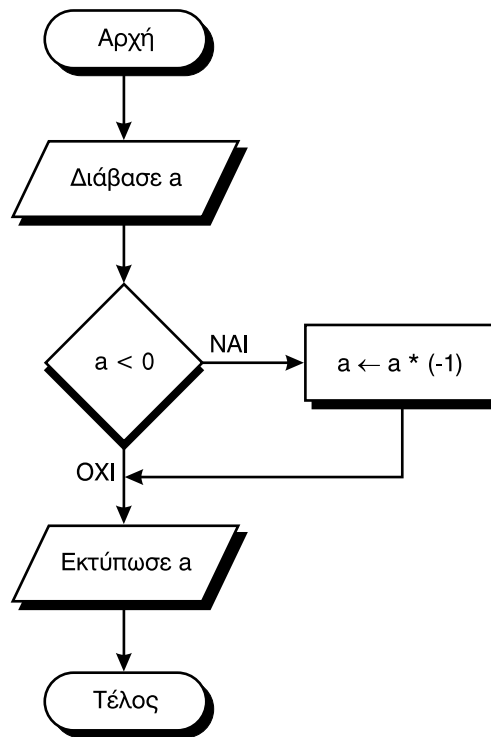
Όπως και στον αλγόριθμο του προηγούμενου παραδείγματος, εύκολα προκύπτει ότι η τιμή a είναι και είσοδος αλλά και έξοδος του αλγορίθμου. Επιπλέον, ο αλγόριθμος έχει καθορισμένη κάθε του εντολή (καθοριστικότητα), τελειώνει μετά από πεπερασμένο αριθμό βημάτων (περατότητα), ενώ κάθε εντολή του είναι ιδιαίτερα απλή κατά την εκτέλεσή της (αποτελεσματικότητα). Έτσι προκύπτει ότι ο αλγόριθμος αυτός πράγματι πληροί τα κριτήρια που περιγράφηκαν στην παράγραφο 2.1.



$$|+5|=5 \text{ και } |-5|=5$$



Η συνθήκη είναι μια λογική έκφραση.



Σχ. 2.2. Ο αλγόριθμος του παραδείγματος 2 με διάγραμμα ροής

### Παράδειγμα 3. Σύγκριση αριθμών με σύνθετη επιλογή

Να διαβασθούν δύο αριθμοί και σε περίπτωση που ο πρώτος αριθμός είναι μικρότερος του δεύτερου, να υπολογισθεί και να εκτυπωθεί το άθροισμά τους, διαφορετικά να υπολογισθεί και να εκτυπωθεί το γινόμενό τους.

**Αλγόριθμος** Παράδειγμα\_3

**Διάβασε** a, b

**Αν**  $a < b$  **τότε**

$c \leftarrow a + b$

**αλλιώς**

$c \leftarrow a * b$

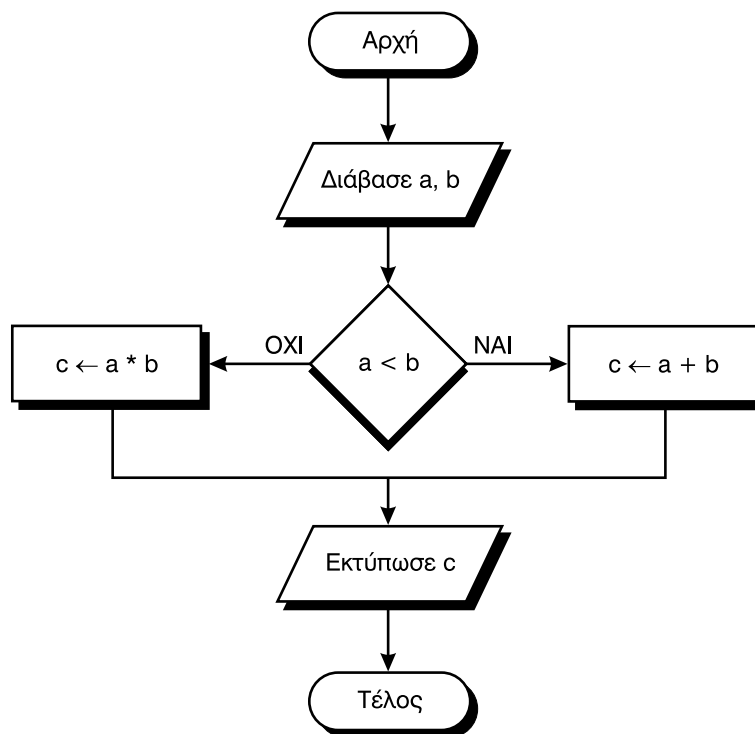
**Τέλος\_αν**

**Εκτύπωσε** c

**Τέλος** Παράδειγμα\_3

Στο παράδειγμα αυτό χρησιμοποιείται η γενική μορφή της εντολής επιλογής, που είναι:

**Αν** συνθήκη **τότε**  
 εντολή ή εντολές  
**αλλιώς**  
 εντολή ή εντολές  
**Τέλος\_αν**



Σχ. 2.3. Ο αλγόριθμος του παραδείγματος 3 με διάγραμμα ροής

### 2.4.3 Διαδικασίες πολλαπλών επιλογών

Οι διαδικασίες των πολλαπλών επιλογών εφαρμόζονται στα προβλήματα όπου μπορεί να ληφθούν διαφορετικές αποφάσεις ανάλογα με την τιμή που παίρνει μία έκφραση. Για παράδειγμα, κάθε γράμμα της αλφαβήτου μπορεί να αντιστοιχιστεί σε κάποιον ακέραιο αριθμό από το 1 μέχρι και 24, για τις ανάγκες κάποιας κωδικοποίησης. Στο παράδειγμα που ακολουθεί παρουσιάζεται μία περίπτωση πολλαπλών επιλογών με διαφορετική ακολουθία εντολών σε κάθε περίπτωση.

**Παράδειγμα 4. Ανάθεση γραμμάτων σε αριθμούς**

Να διαβασθεί ένας ακέραιος και να εκτυπωθεί το αντίστοιχο γράμμα της αλφαβήτου, αν ο ακέραιος έχει τιμή 1 ή 2 ή 3 διαφορετικά να εκτυπωθεί η λέξη “άγνωστος”.

**Αλγόριθμος** Παράδειγμα\_4

**Διάβασε** a

**Αν** a = 1 **τότε εκτύπωσε** 'Α'

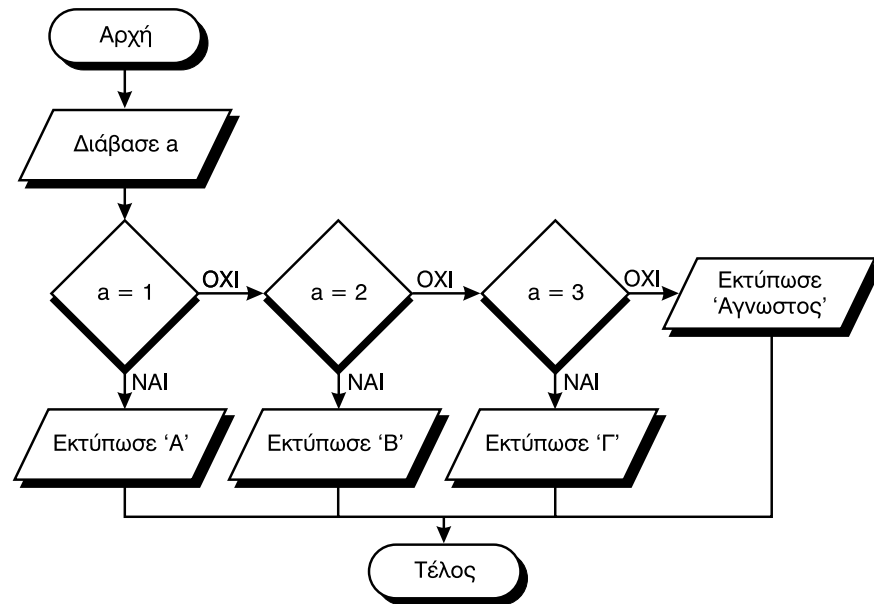
**αλλιώς\_αν** a = 2 **τότε εκτύπωσε** 'Β'

**αλλιώς\_αν** a = 3 **τότε εκτύπωσε** 'Γ'

**αλλιώς εκτύπωσε** 'άγνωστος'

**Τέλος\_αν**

**Τέλος** Παράδειγμα\_4



**Σχ. 2.4.** Ο αλγόριθμος του παραδείγματος 4 με διάγραμμα ροής

Αν οι διαφορετικές επιλογές είναι πολλές, τότε είναι προτιμότερο να χρησιμοποιηθεί το σχήμα πολλαπλής επιλογής *Επίλεξε...Τέλος επιλογών* (select case), όπως στο παράδειγμα που ακολουθεί.

### Παράδειγμα 5. Επιλογή ορίων

Να εισαχθεί ένας ακέραιος που αντιστοιχεί σε μια ηλικία και να βρεθεί σε ποια όρια εντάσσεται η δεδομένη ηλικία εμφανίζοντας σχετικό μήνυμα.

**Αλγόριθμος** Παράδειγμα\_5.

**Γράψε** "Σε ποια ηλικία άρχισες να μαθαίνεις προγραμματισμό ;"

**Διάβασε** age

**Επίλεξε**

**Περίπτωση** age < 0

**Εμφάνισε** "Είπαμε ηλικία ..."

**Περίπτωση**  $0 \leq \text{age} < 5$

**Εμφάνισε** "Μάλλον τα παραλές !!"

**Περίπτωση**  $5 \leq \text{age} < 60$

**Εμφάνισε** "Μπράβο"

**Περίπτωση**  $60 \leq \text{age} < 100$

**Εμφάνισε** "Ποτέ δεν είναι αργά"

**Περίπτωση** age > 100

**Εμφάνισε** "Κάλλιο αργά παρά ποτέ"

**Τέλος\_επιλογών**

**Τέλος** Παράδειγμα\_5

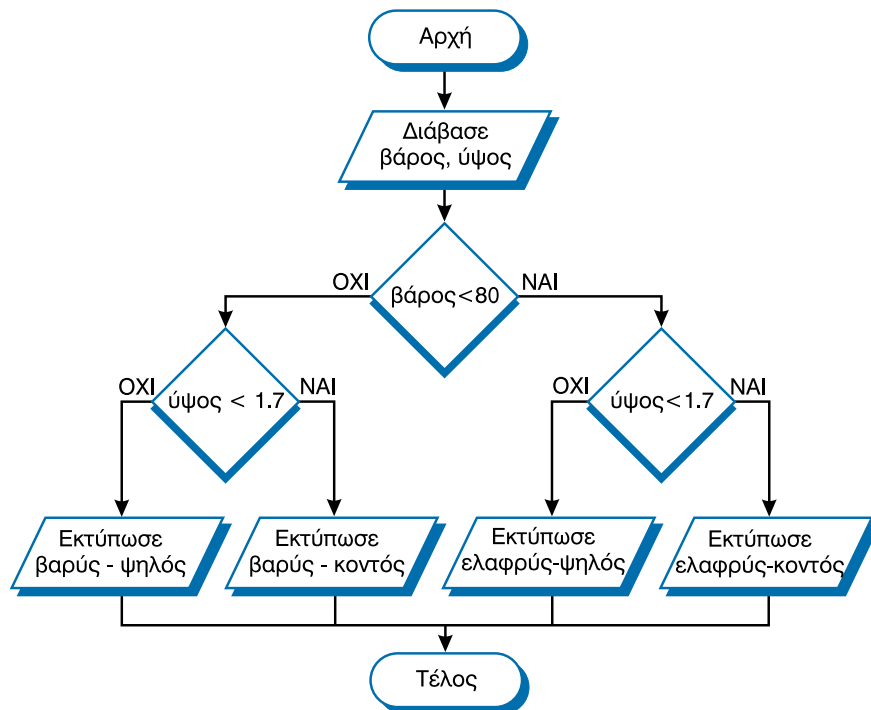
### 2.4.4 Εμφωλευμένες Διαδικασίες

Πολλαπλές επιλογές μπορούν να γίνουν και με μία εμφωλευμένη δομή. Το επόμενο παράδειγμα περιγράφει τον τρόπο με τον οποίο μία εντολή Αν...τότε είναι η εντολή που εκτελείται, όταν ισχύει (ή δεν) ισχύει η συνθήκη μίας άλλης εντολής Αν...τότε. Βέβαια η λογική αυτή μπορεί να επεκταθεί, δηλαδή να έχουμε νέα εμφωλευμένη δομή μέσα σε μία εμφωλευμένη δομή κοκ.

### Παράδειγμα 6. Χαρακτηρισμός ατόμων

Να διαβάζονται δύο αριθμοί που αντιστοιχούν στο ύψος και βάρος ενός άνδρα. Να εκτυπώνεται ότι ο άνδρας είναι "ελαφρύς", αν το βάρος του είναι κάτω από 80 κιλά, ή να εκτυπώνεται "βαρύς" στην αντίθετη περίπτωση. Επίσης να εκτυπώνεται "κοντός" αν το ύψος του είναι κάτω από 1.70, αλλιώς να εκτυπώνεται "ψηλός".

**Αλγόριθμος** Παράδειγμα\_6  
**Διάβασε** βάρος, ύψος  
**Αν** βάρος < 80 **τότε**  
    **Αν** ύψος < 1.70 **τότε**  
        **εκτύπωσε** 'Ελαφρύς, κοντός'  
    **αλλιώς**  
        **εκτύπωσε** 'ελαφρύς, ψηλός'  
    **Τέλος\_αν**  
**αλλιώς**  
    **Αν** ύψος < 1.70 **τότε**  
        **εκτύπωσε** 'Βαρύς, κοντός'  
    **αλλιώς**  
        **εκτύπωσε** 'βαρύς, ψηλός'  
    **Τέλος\_αν**  
**Τέλος\_αν**  
**Τέλος** Παράδειγμα\_5



Σχ. 2.5. Ο αλγόριθμος του παραδείγματος 6 με διάγραμμα ροής

Σε πολλές περιπτώσεις η συνθήκη είναι αρκετά πιο “δύσκολη”, δηλαδή

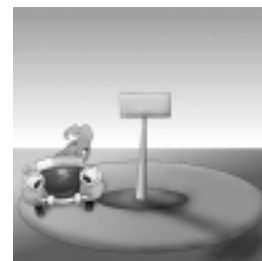


εμπεριέχει αποφάσεις που πιθανόν να βασίζονται σε περισσότερα από ένα κριτήρια. Ο συνδυασμός των κριτηρίων αυτών καθορίζει και τις “λογικές” πράξεις που μπορούν να γίνουν μεταξύ διαφορετικών συνθηκών. Πολύ συχνά στην καθημερινή ζωή κάποιες αποφάσεις βασίζονται σε συνδυασμούς κριτηρίων και λογικών πράξεων. Για παράδειγμα, το πρόβλημα της προετοιμασίας μας για έξοδο μπορεί να επεκταθεί ως εξής “αν βρέχει **ή** αν χιονίζει θα πάρω ομπρέλα”, είτε στην πρόταση “αν έχει ήλιο **και** αν έχει ζέστη θα πάρω καπέλο”, είτε στην πρόταση “αν **δεν** έχει ήλιο θα πάρω ομπρέλα”. Οι τρεις αυτές προτάσεις περιγράφουν και τις τρεις λογικές πράξεις που μπορεί να ισχύουν μεταξύ διαφορετικών συνθηκών. Η λογική πράξη **ή** είναι αληθής όταν οποιαδήποτε από τις δύο προτάσεις είναι αληθής. Η λογική πράξη **και** είναι αληθής όταν και οι δύο προτάσεις είναι αληθείς ενώ η λογική πράξη **όχι** (η λέξη “δεν” στο παράδειγμά μας) είναι αληθής όταν η πρόταση που την ακολουθεί είναι ψευδής. Ο επόμενος πίνακας δίνει τις τιμές των τριών αυτών λογικών πράξεων για όλους τους συνδυασμούς τιμών.

Πρόταση Α	Πρόταση Β	Α ή Β	Α και Β	όχι Α
Αληθής	Αληθής	Αληθής	Αληθής	Ψευδής
Αληθής	Ψευδής	Αληθής	Ψευδής	Ψευδής
Ψευδής	Αληθής	Αληθής	Ψευδής	Αληθής
Ψευδής	Ψευδής	Ψευδής	Ψευδής	Αληθής

#### 2.4.5 Δομή Επανάληψης

Η διαδικασία της επανάληψης είναι ιδιαίτερα συχνή, αφού πλήθος προβλημάτων μπορούν να επιλυθούν με κατάλληλες επαναληπτικές διαδικασίες. Η λογική των επαναληπτικών διαδικασιών εφαρμόζεται στις περιπτώσεις, όπου μία ακολουθία εντολών πρέπει να εφαρμοσθεί σε ένα σύνολο περιπτώσεων, που έχουν κάτι κοινό. Για παράδειγμα, όλες οι τράπεζες κάθε εξάμηνο αποδίδουν τόκους των καταθέσεων ταμειευτηρίου. Ο υπολογισμός των τόκων πρέπει να γίνει για όλους τους λογαριασμούς της τράπεζας, άρα η πράξη



$$\text{τόκος} = \text{ποσό} * \text{επιτόκιο}$$

πρέπει να εκτελεσθεί για όλους τους τραπεζικούς λογαριασμούς. Οι επαναληπτικές διαδικασίες μπορεί να έχουν διάφορες μορφές και συνήθως εμπεριέχουν και συνθήκες επιλογών (όπως αυτές περιγράφηκαν στην προηγούμενη υποπαράγραφο). Στη συνέχεια δίνεται ένα σύνολο παρα-

δειγμάτων που εντάσσονται στις πλέον γνωστές κατηγορίες επαναληπτικών διαδικασιών.

### Παράδειγμα 7. Εκτύπωση διαδοχικών αριθμών με επαναληπτική εντολή: όσο...επανάλαβε

**Να γραφεί αλγόριθμος που να εμφανίζει τους αριθμούς από 1 έως 100.**



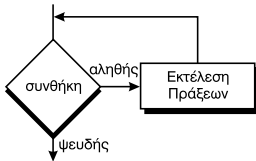
$i \leftarrow i + 1 \Rightarrow$  “η νέα τιμή της μεταβλητής  $i$  είναι η προηγούμενη συν ένα”

Στον αλγόριθμο αυτό επιζητείται η παρουσίαση μίας σειράς αριθμών. Αν οι αριθμοί αυτοί ήσαν λίγοι, τότε αυτό θα μπορούσε να γίνει με την παράθεση αντίστοιχων εντολών εμφάνισης. Το ίδιο θα συμβεί και στην περίπτωση που οι αριθμοί είναι περισσότεροι, αλλά δεν έχουν καμία σχέση μεταξύ τους π.χ. 5, 207, -32 κ.ο.κ. Όμως στο ζητούμενο αλγόριθμο παρατηρούμε ότι κάθε αριθμός παράγεται από τον προηγούμενό του με απλό τρόπο δηλαδή προσθέτοντας κάθε φορά το 1. Μπορεί λοιπόν να χρησιμοποιηθεί μια μεταβλητή, έστω  $i$ , η οποία αρχίζει από το 1 και καταλήγει στο 100 αυξανόμενη κατά 1. Η εκάστοτε αύξηση της μεταβλητής αυτής μπορεί να γίνει με τη χρήση της εντολής εκχώρησης

$i \leftarrow i + 1$

Η αρχική τιμή της μεταβλητής  $i$  ορίζεται εύκολα με την εντολή  $i \leftarrow 1$ . Το ζητούμενο είναι να εκτελεστεί 100 φορές η εντολή  $i \leftarrow i + 1$ . Αυτό επιτυγχάνεται με τη χρήση της εντολής Όσο ...επανάλαβε. Η σύνταξη της εντολής αυτής είναι:

**Όσο** συνθήκη **επανάλαβε**  
εντολές  
**Τέλος\_επανάληψης**



Η λειτουργία της εντολής είναι η εξής: Επαναλαμβάνεται η εκτέλεση των εντολών, όσο η συνθήκη είναι αληθής. Όταν η συνθήκη γίνει ψευδής, τότε ο αλγόριθμος συνεχίζεται με την εντολή που ακολουθεί το ‘Τέλος\_επανάληψης’. Με την εισαγωγή της εντολής αυτής η σχεδίαση του ζητούμενου αλγορίθμου είναι:

**Αλγόριθμος** Παράδειγμα\_7

$i \leftarrow 1$

**Όσο**  $i \leq 100$  **επανάλαβε**

**Εμφάνισε**  $i$

$i \leftarrow i + 1$

**Τέλος\_επανάληψης**

**Τέλος** Παράδειγμα\_7



Το τμήμα του αλγορίθμου που επαναλαμβάνεται, δηλαδή από την εντολή Όσο μέχρι το Τέλος\_επανάληψης αποκαλείται **βρόχος**.

### Παράδειγμα 8: Επαναληπτική είσοδος στοιχείων

**Να γραφεί αλγόριθμος που να διαβάζει ένα άγνωστο πλήθος αριθμών και να εμφανίζει τον κάθε αριθμό.**

Το πρόβλημα αυτό παρουσιάζει την εξής ιδιομορφία: ενώ φαίνεται ότι θα χρησιμοποιηθεί για τη λύση του κάποια επαναληπτική διαδικασία, δεν προσδιορίζεται ο τρόπος τερματισμού της. Κατ'αρχήν, λοιπόν, ας εξετάσουμε τον αλγόριθμο που εκτελεί ένας άνθρωπος, όταν αντιγράφει κάποιους αριθμούς, όπως για παράδειγμα όταν συγκεντρώνονται τα έξοδα από διάφορους λογαριασμούς. Ο αλγόριθμος αυτός είναι:

**Βήμα 1.** Διάβασε έναν αριθμό

**Βήμα 2.** Γράψε τον αριθμό

**Βήμα 3.** Επανάλαβε τη διαδικασία από το βήμα 1.

Ο αλγόριθμος αυτός έχει μια ατέλεια, δεν διαθέτει τρόπο τερματισμού (ατέρμων βρόχος). Η έλλειψη αυτή είναι φυσική, εφόσον ο αλγόριθμος εκτελείται από έναν άνθρωπο. Αυτός θα σταματήσει να γράφει, όταν δεν υπάρχουν πλέον άλλοι αριθμοί. Ωστόσο είναι δυνατόν να διορθωθεί αυτή η ατέλεια, αν το βήμα 3 λάβει την εξής μορφή:

**Βήμα 3.** Αν υπάρχουν άλλοι αριθμοί, επανάλαβε τη διαδικασία από το βήμα 1, αλλιώς σταμάτησε.

Ο αλγόριθμος τώρα είναι σωστός και μπορεί να εκτελεστεί και από μία μηχανή. Όμως έχει ένα άλλο μειονέκτημα: ο τερματισμός γίνεται μέσα από την εντολή Αν ... τότε ... αλλιώς ..., πράγμα που δεν συνιστάται και πρέπει να αποφεύγεται, γιατί εύκολα μπορεί να χάσει ο προγραμματιστής τον έλεγχο της ροής του προγράμματος και να οδηγηθεί σε λάθος. Για την άρση του μειονεκτήματος αυτού πρέπει να χρησιμοποιηθεί μία εντολή επαναληπτικής διαδικασίας, όπως η εντολή Όσο...επανάλαβε. Ο τελικός αλγόριθμος είναι ο εξής:

**Αλγόριθμος** Παράδειγμα\_8

**Διάβασε**  $x$

**Όσο**  $x > 0$  **επανάλαβε**

**Εμφάνισε**  $x$

**Διάβασε**  $x$

**Τέλος\_επανάληψης**

**Τέλος** Παράδειγμα\_8



*Ο βρόχος επανάληψης μπορεί να μην εκτελεσθεί καμία φορά, αν η πρώτη τιμή που διαβάζεται είναι αρνητική.*

Στον προηγούμενο αλγόριθμο η επαναληπτική διαδικασία τερματίζεται, όταν διαβασθεί ένας αρνητικός ή μηδενικός αριθμός. Δηλαδή, θεωρείται ότι

οι εισαγόμενοι αριθμοί πρέπει να είναι θετικοί. Αν αυτό δεν συμβαίνει, τότε μπορεί να χρησιμοποιηθεί ως συνθήκη τερματισμού οποιαδήποτε συγκεκριμένη τιμή έχει συμφωνηθεί, ότι θα χρησιμοποιείται για το σκοπό αυτό, π.χ. η 999999. Προφανώς αυτή η τιμή δεν μπορεί να ανήκει στις εισαγόμενες τιμές. Στην περίπτωση αυτή η εντολή Όσο...επανάλαβε θα γραφεί ως εξής:

**Όσο**  $x \neq 999999$  **επανάλαβε**

### Παράδειγμα 9. Εκτύπωση θετικών αριθμών με εντολή: αρχή\_επανάληψης...μέχρις\_ότου

Να διαβάζονται και να εκτυπώνονται όσοι θετικοί αριθμοί δίνονται από το πληκτρολόγιο. Ο αλγόριθμος τελειώνει, όταν δοθεί ένας αρνητικός αριθμός.

**Αλγόριθμος** Παράδειγμα\_9

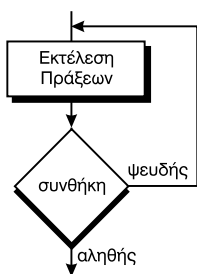
**Αρχή\_επανάληψης**

Διάβασε  $x$

Εμφάνισε  $x$

**Μέχρις\_ότου**  $x < 0$

**Τέλος** Παράδειγμα\_9



Η εντολή Αρχή\_επανάληψης...Μέχρις\_ότου εκτελείται οπωσδήποτε μια φορά

Ας σημειωθεί ότι, στο παράδειγμα αυτό ο βρόχος επανάληψης θα εκτελεσθεί οπωσδήποτε τουλάχιστον μία φορά ακόμα και αν η αρχική τιμή της μεταβλητής  $x$  είναι αρνητική. Η βασική διαφοροποίηση αυτής της μορφής επαναληπτικής διαδικασίας σε σχέση με την επαναληπτική διαδικασία που παρουσιάστηκε στο προηγούμενο παράδειγμα, οφείλεται στη θέση της λογικής συνθήκης στη ροή εκτέλεσης των εντολών.

### Παράδειγμα 10. Υπολογισμός αθροίσματος αριθμών με επαναληπτική εντολή: για...από...μέχρι

Να βρεθεί και να εκτυπωθεί το άθροισμα των 100 ακεραίων από το 1 μέχρι το 100.

Όταν ο αριθμός των φορών που θα εκτελεστεί μια επαναληπτική διαδικασία είναι γνωστός εκ των προτέρων, τότε είναι προτιμότερο να χρησιμοποιείται η εντολή Για...από...μέχρι. Έτσι ο ζητούμενος αλγόριθμος είναι.

**Αλγόριθμος** Παράδειγμα\_10

Sum  $\leftarrow$  0

**Για** i από 1 **μέχρι** 100

Sum  $\leftarrow$  Sum + i

**Τέλος\_επανάληψης**

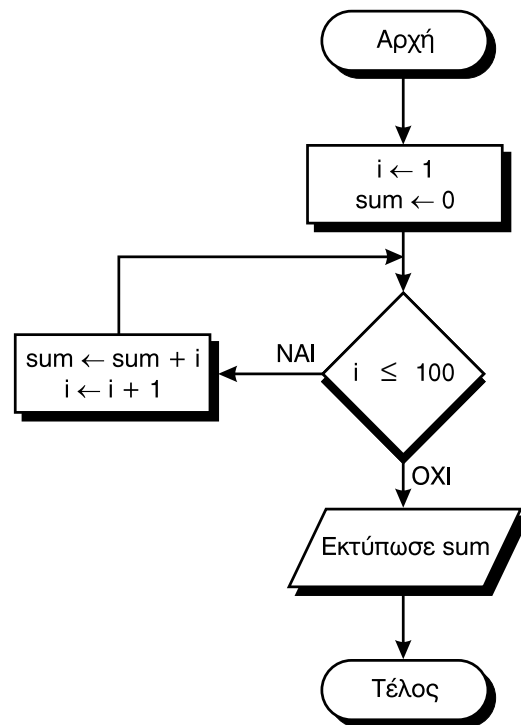
**Εκτύπωσε** Sum

**Τέλος** Παράδειγμα\_10



$Sum \leftarrow Sum + i \Rightarrow$  Η νέα τιμή του Sum είναι η παληά συν i

Όπως γίνεται φανερό, η εντολή Για...από...μέχρι περιλαμβάνει όλα τα απαιτούμενα στοιχεία για την επανάληψη, δηλαδή αρχική τιμή της μεταβλητής i (=1) και τελική τιμή (=100). Το βήμα μεταβολής της μεταβλητής i είναι 1, το οποίο υπονοείται και δεν σημειώνεται, όταν είναι 1. Η μεταβλητή Sum που υποδέχεται το άθροισμα των διαδοχικών αριθμών, πρέπει να εκκινήσει με τιμή 0, ενώ το εκάστοτε μερικό άθροισμα υπολογίζεται με την εντολή εκχώρησης εντός του βρόχου. Στο τέλος η μεταβλητή Sum θα περιέχει το τελικό άθροισμα.



Σχ. 2.6. Ο αλγόριθμος του παραδείγματος 10 με διάγραμμα ροής

**Παράδειγμα 11. Υπολογισμός αθροίσματος με επαναληπτική εντολή: για...από...μέχρι...βήμα**

**Να βρεθεί και να εκτυπωθεί το άθροισμα των άρτιων αριθμών από το 1 μέχρι το 100.**

Η λύση αυτού του προβλήματος είναι παρόμοια με αυτή του προηγούμενου. Η μόνη αλλαγή είναι στην εντολή επανάληψης όπου προσδιορίζεται η ποσότητα βήμα, η οποία κάθε φορά προστίθεται στην τιμή της μεταβλητής  $i$ . Έτσι έχουμε

**Αλγόριθμος** Παράδειγμα\_11

άθροισμα  $\leftarrow 0$

**Για**  $i$  **από** 2 **μέχρι** 100 **με\_βήμα** 2

    άθροισμα  $\leftarrow$  άθροισμα +  $i$

**Τέλος\_επανάληψης**

**Εκτύπωσε** άθροισμα

**Τέλος** Παράδειγμα\_11

Από τα προηγούμενα δύο παραδείγματα γίνεται φανερός ο τρόπος χρήσης της εντολής Για...από...μέχρι. Ας σημειωθεί ωστόσο, ότι υπάρχουν κάποιες δεσμεύσεις μεταξύ των τιμών από, μέχρι και βήμα. Έτσι το βήμα δεν μπορεί να είναι μηδέν, γιατί τότε ο βρόχος εκτελείται επ' άπειρον. Είναι δυνατόν όμως το βήμα να έχει αρνητική τιμή, αρκεί η τιμή από να είναι μεγαλύτερη από την τιμή μέχρι, όπως για παράδειγμα στην επόμενη εντολή:

**Για**  $k$  **από** 100 **μέχρι** 0 **με\_βήμα** -1

Επίσης οι τιμές από, μέχρι και βήμα δεν είναι απαραίτητο να είναι ακέραιες. Μπορούν λάβουν οποιαδήποτε πραγματική τιμή. Για παράδειγμα, όταν ζητείται να βρεθούν διαδοχικές τιμές μιας συνάρτησης  $f(x)$  για  $x$  από 0 έως 1, τότε μπορεί να γραφεί η επόμενη εντολή

**Για**  $x$  **από** 0 **μέχρι** 1 **με\_βήμα** 0,01

**Παράδειγμα 12. Πολλαπλασιασμός αλά ρωσικά**

Στη συνέχεια προχωρούμε στην ανάπτυξη ενός συνθετότερου προβλήματος, όπου για την αλγοριθμική του επίλυση γίνεται χρήση αρκετών από τις προηγούμενες δομές. Ας θεωρήσουμε την πράξη του πολλαπλασιασμού δύο ακεραίων αριθμών και ας θυμηθούμε πως αυτή υλοποιείται χειρωνακτικά. Τοποθετούμε, λοιπόν, τους δύο αριθμούς τον ένα κάτω από τον άλλο και πολλαπλασιάζουμε κάθε ψηφίο του κάτω αριθμού με όλα ψηφία του επάνω αριθμού. Πιο συγκεκριμένα, για κάθε ψηφίο του κάτω αριθμού παράγεται ένα μερικό γινόμενο, ενώ τα μερικά γινόμενα τοποθετούνται το



Ο βρόχος **Για**  $k$  **από** 5 **μέχρι** 5 εκτελείται ακριβώς μία φορά

Ο βρόχος **Για**  $k$  **από** 5 **μέχρι** 1 δεν εκτελείται καμία φορά



ένα κάτω από το άλλο με μία μετατόπιση από τα δεξιά προς τα αριστερά καθώς θεωρούμε διαδοχικά τα ψηφία των μονάδων, των δεκάδων, των εκατοντάδων κ.λπ. Στη συνέχεια γίνεται η πρόσθεση των επιμέρους γινομένων, αφού τα τοποθετήσουμε στην κατάλληλη διάταξη όπως φαίνεται στο σχήμα 2.7.

$$\begin{array}{r} 45 \\ \times 19 \\ \hline 405 \\ + 45 \\ \hline 855 \end{array}$$

**Σχ. 2.7.**  
Χειρωνακτικός τρόπος  
πολλαπλασιασμού.

Ωστόσο, η πράξη του πολλαπλασιασμού δεν εκτελείται από τον υπολογιστή με τον τρόπο αυτό. Πιο συγκεκριμένα, ο χρησιμοποιούμενος τρόπος είναι ο λεγόμενος *πολλαπλασιασμός αλά ρωσικά*. Χωρίς βλάβη της γενικότητας θεωρούμε ότι οι ακέραιοι είναι θετικοί (μεγαλύτεροι του μηδενός), αλλά η μέθοδος μπορεί εύκολα να μετατραπεί, ώστε να περιγράψει και την περίπτωση των αρνητικών ακεραίων. Πως ακριβώς λειτουργεί η μέθοδος, θα φανεί με το επόμενο παράδειγμα, όπου περιγράφεται ο αλγόριθμος με ελεύθερο κείμενο.

Έστω, λοιπόν, ότι δίνονται δύο θετικοί ακέραιοι αριθμοί, οι αριθμοί 45 και 19. Οι αριθμοί γράφονται δίπλα-δίπλα και ο πρώτος διπλασιάζεται αγνοώντας το δεκαδικό μέρος, ενώ ο δεύτερος υποδιπλασιάζεται. Στο σχήμα 2.8 παρουσιάζεται η επαναλαμβανόμενη διαδικασία, που συνεχίζεται μέχρις ότου στη δεύτερη στήλη να προκύψει μονάδα. Τελικώς, το γινόμενο ισούται με το άθροισμα των στοιχείων της πρώτης στήλης, όπου αντίστοιχα στη δεύτερη στήλη υπάρχει περιττός αριθμός. Για το παράδειγμά μας, τα στοιχεία αυτά παρουσιάζονται στην τρίτη στήλη.

45	19	45
90	9	90
180	4	
360	2	
720	1	720
Άθροισμα =		855

**Σχ. 2.8.** Πολλαπλασιασμός αλά ρωσικά.

### Ολίσθηση (shift)

Στα κυκλώματα του υπολογιστή τα δεδομένα αποθηκεύονται με δυαδική μορφή, δηλαδή 0 και 1, ανεξάρτητα από το πως τα ορίζει ο προγραμματιστής, όπως ακεραίους ή πραγματικούς σε δεκαδικό σύστημα, ή ακόμη χαρακτήρες κ.λπ. Έτσι ο αριθμός 17 του δεκαδικού συστήματος ισοδυναμεί με τον αριθμό 00010001 του δυαδικού συστήματος, ο οποίος μπορεί να αποθηκευθεί σε ένα byte. Αν μετακινήσουμε τα ψηφία αυτά κατά μία θέση προς τα αριστερά, δηλαδή αν προσθέσουμε ένα 0 στο τέλος του αριθμού και αγνοήσουμε το αρχικό 0, τότε προκύπτει ο αριθμός 00100010 του δυαδικού συστήματος, που ισοδυναμεί με το αριθμό 34 του δεκαδικού συστήματος. Επίσης, με παρόμοιο τρόπο, αν μετακινήσουμε τα ψηφία κατά μία θέση δεξιά, δηλαδή αποκόψουμε το τελευταίο ψηφίο 1 και θεωρήσουμε ένα ακόμη αρχικό 0, τότε προκύπτει ο αριθμός 00001000 του δυαδικού συστήματος, που ισοδυναμεί με τον αριθμό 8 του δεκαδικού συστήματος. Άρα η ολίσθηση προς τα αριστερά ισοδυναμεί με πολλαπλασιασμό επί δύο, ενώ η ολίσθηση προς τα δεξιά ισοδυναμεί με την ακέραια διαίρεση διά δύο.

## Στοιχεία ψευδογλώσσας

### 1. Σταθερές

Αριθμητικές: χρησιμοποιούνται οι αριθμητικοί χαρακτήρες, το +, το - και το κόμμα ως δεκαδικό σημείο,

Αλφαριθμητικές: σχηματίζονται από οποιουδήποτε χαρακτήρες εντός διπλών εισαγωγικών,

Λογικές: υπάρχουν δύο, οι Αληθής και Ψευδής.

### 2. Μεταβλητές

Για τη σύνθεση του ονόματος μιας μεταβλητής χρησιμοποιούνται οι αριθμητικοί χαρακτήρες, οι αλφαριθμητικοί χαρακτήρες πεζοί και κεφαλαίοι, καθώς και ο χαρακτήρας \_ (underscore). Οι μεταβλητές μπορούν επίσης να είναι αριθμητικές, αλφαριθμητικές και λογικές.

### 3. Τελεστές

Αριθμητικοί: +, -, \*, /, ^

Συγκριτικοί: ≤, <, =, ≠, >, ≥

Λογικοί: και (σύζευξη), ή (διάζευξη), όχι (άρνηση).

### 4. Εκφράσεις

Σχηματίζονται από σταθερές, μεταβλητές, συναρτήσεις, τελεστές και παρενθέσεις.

### 5. Εντολή εκχώρησης

Μεταβλητή ← έκφραση

### 6. Σχήματα λογικών υποθέσεων

**Αν** <συνθήκη> **τότε** <εντολή>

**Αν** <συνθήκη> **τότε**  
<διαδικασία\_1>

**αλλιώς**  
<διαδικασία\_2>

**Τέλος\_αν**

**Επίλεξε** έκφραση

**Περίπτωση** 1

Διαδικασία\_1

.....

**Περίπτωση** ν

Διαδικασία\_ν

**Περίπτωση αλλιώς**

Διαδικασία\_αλλιώς

**Τέλος\_επιλογών**

**Αν** <συνθήκη\_1> **τότε**  
<διαδικασία\_1>

**αλλιώς\_αν** <συνθήκη\_2> **τότε**  
<διαδικασία\_2>

.....

**αλλιώς\_αν** <συνθήκη\_ν> **τότε**  
<διαδικασία\_ν>

**αλλιώς**

<διαδικασία\_αλλιώς>

**Τέλος\_αν**



όπου ως διαδικασία λαμβάνεται ένα σύνολο εντολών

### 7. Επαναληπτικές διαδικασίες

⇒ Επαναληπτικό σχήμα με έλεγχο επανάληψης στην αρχή

**Όσο** <συνθήκη> **επανάλαβε**

Διαδικασία

**Τέλος\_επανάληψης**

⇒ Επαναληπτικό σχήμα με έλεγχο επανάληψης στο τέλος

**Αρχή\_επανάληψης**

Διαδικασία

**Μέχρις\_ότου** <συνθήκη>

⇒ Επαναληπτικό σχήμα ορισμένων φορών επανάληψης

**Για** μεταβλητή **από** τ1 **μέχρι** τ2 **με\_βήμα** β

Διαδικασία

**Τέλος\_επανάληψης**

### 8. Ρήματα σε προστακτική

Για παράδειγμα, “Διάβασε”, “Γράψε”, “Εκτέλεσε” κ.λπ.

### 9. Ουσιαστικά

Σε ορισμένες περιπτώσεις όταν οι ζητούμενες ενέργειες είναι πολλές ή προφανείς, καθορίζονται με τη χρήση ουσιαστικών αντί ρημάτων, όπως “εισαγωγή δεδομένων”, “εμφάνιση πεδίων στην οθόνη” κ.λπ.

### 10. Σχόλια

Προκειμένου να διαχωρίζονται οι επεξηγηματικές φράσεις από τις λέξεις-κλειδιά του αλγορίθμου, στις πρώτες προτάσσεται το σύμβολο !, για παράδειγμα !Σχόλια.

**11. Πρώτη και τελευταία γραμμή** ενός αλγορίθμου είναι αντίστοιχα

**Αλγόριθμος** <όνομα\_αλγορίθμου> και **Τέλος** <όνομα\_αλγορίθμου>

### 12. Δεδομένα και αποτελέσματα

Τα δεδομένα εισόδου (αν υπάρχουν) περιγράφονται στη δεύτερη γραμμή του αλγορίθμου εντός των συμβόλων // ... //. Αντίστοιχα τα αποτελέσματα εξόδου δίνονται στην προτελευταία γραμμή του αλγορίθμου εντός των συμβόλων // ... //.

Η μέθοδος αυτή χρησιμοποιείται πρακτικά στους υπολογιστές, γιατί υλοποιείται πολύ πιο απλά απ' ό,τι ο γνωστός μας χειρωνακτικός τρόπος πολλαπλασιασμού. Πιο συγκεκριμένα, απαιτεί πολλαπλασιασμό επί δύο, διαίρεση διά δύο και πρόσθεση. Σε αντίθεση η γνωστή μας διαδικασία πολλαπλασιασμού απαιτεί πολλαπλασιασμό με οποιοδήποτε ακέραιο και πρόσθεση. Σε επίπεδο, λοιπόν, κυκλωμάτων υπολογιστή ο πολλαπλασιασμός επί δύο και η διαίρεση διά δύο μπορούν να υλοποιηθούν ταχύτατα με μία απλή εντολή ολίσθησης (shift), σε αντίθεση με τον πολλαπλασιασμό με οποιοδήποτε ακέραιο που θεωρείται πιο χρονοβόρα διαδικασία. Το τελευταίο γεγονός είναι ο λόγος που ο πολλαπλασιασμός αλά ρωσικά είναι προτιμότερος απ' ό,τι ο χειρωνακτικός τρόπος πολλαπλασιασμού δύο ακεραίων.

Στη συνέχεια παρουσιάζεται ο αλγόριθμος πολλαπλασιασμού ακεραίων αλά ρωσικά με φυσική γλώσσα κατά βήματα.

Αλγόριθμος: Πολλαπλασιασμός δύο θετικών ακεραίων (αλά ρωσικά)	
Είσοδος:	Δύο ακέραιοι M1 και M2, όπου $M1, M2 \geq 1$
Έξοδος:	Το γινόμενο $P = M1 * M2$
Βήμα 1	Θέσε $P = 0$
Βήμα 2	Αν $M2 > 0$ , τότε πήγαινε στο Βήμα 3, αλλιώς πήγαινε στο Βήμα 7
Βήμα 3	Αν ο M2 είναι περιττός, τότε θέσε $P = P + M1$
Βήμα 4	Θέσε $M1 = M1 * 2$
Βήμα 5	Θέσε $M2 = M2 / 2$ (θεώρησε μόνο το ακέραιο μέρος)
Βήμα 6	Πήγαινε στο Βήμα 2
Βήμα 7	Τύπωσε τον P.

Ακολουθεί ο αλγόριθμος σε ψευδοκώδικα για το ίδιο πρόβλημα του πολλαπλασιασμού αλά ρωσικά.

**Αλγόριθμος** Πολλαπλασιασμός\_αλά\_ρωσικά

**Δεδομένα** // M1, M2 ακέραιοι //

$P \leftarrow 0$

**Όσο** M2 > 0 **επανάλαβε**

**Αν** M2 mod 2 = 1 **τότε**  $P \leftarrow P + M1$

$M1 \leftarrow M1 * 2$

$M2 \leftarrow [M2 / 2]$

**Τέλος\_επανάληψης**

**Αποτελέσματα** // P, το γινόμενο των ακεραίων M1, M2 //

**Τέλος** Πολλαπλασιασμός\_αλά\_ρωσικά

## Ανακεφαλαίωση

Στο κεφάλαιο αυτό έγινε η πρώτη γνωριμία με την έννοια του αλγορίθμου. Δόθηκαν οι απαραίτητοι ορισμοί που συνοδεύτηκαν με αρκετά παραδείγματα. Αλγόριθμος είναι η διαδικασία της λύσης ενός προβλήματος. Η παράσταση των αλγορίθμων μπορεί να γίνει με αρκετούς τρόπους, ωστόσο η έμφαση δόθηκε στην παράσταση με χρήση ψευδογλώσσας. Στο κεφάλαιο αυτό αναπτύχθηκαν οι κύριες αλγοριθμικές δομές, δηλαδή η ακολουθία, η επιλογή και η επανάληψη ή ανακύκλωση, που θα χρησιμοποιηθούν στους αλγορίθμους των επόμενων κεφαλαίων.



## Λέξεις κλειδιά

Αλγόριθμος, ακολουθία, επιλογή, επανάληψη, διάγραμμα ροής, ψευδογλώσσα, εμφωλευμένος, βρόχος.



## Ερωτήσεις - Θέματα για συζήτηση

1. Να δοθεί ο ορισμός του όρου αλγόριθμος.
2. Ποιά είναι τα κριτήρια που πρέπει να ικανοποιεί κάθε αλγόριθμος;
3. Υπό ποία πρίσματα η Πληροφορική επιστήμη μελετά τους αλγορίθμους;
4. Ποιά η διαφορά της θεωρητικής από την αναλυτική προσέγγιση στην επίλυση ενός προβλήματος με χρήση αλγορίθμου;
5. Περιγράψτε τους τρόπους περιγραφής και αναπαράστασης των αλγορίθμων.
6. Ποιές είναι οι βασικοί τύποι συνιστωσών/εντολών ενός αλγορίθμου ;
7. Να περιγραφεί η δομή της ακολουθίας και να δοθεί σε διάγραμμα ροής ένα παράδειγμα αυτής της αλγοριθμικής προσέγγισης.
8. Να περιγραφεί η δομή της επιλογής και να δοθεί με ακολουθία βημάτων ένα παράδειγμα αυτής της αλγοριθμικής προσέγγισης.
9. Να περιγραφεί η δομή των επαναληπτικών διαδικασιών και να δοθεί με ακολουθία βημάτων και με διάγραμμα ροής ένα παράδειγμα αυτής της αλγοριθμικής προσέγγισης.
10. Να περιγραφεί η δομή των διαδικασιών πολλαπλών επιλογών και να δοθεί με ακολουθία βημάτων και με διάγραμμα ροής ένα παράδειγμα αυτής της αλγοριθμικής προσέγγισης.
11. Να περιγραφεί η δομή των εμφωλευμένων διαδικασιών και να δοθεί με ακολουθία βημάτων και με διάγραμμα ροής ένα παράδειγμα αυτής



της αλγοριθμικής προσέγγισης.

12. Να περιγραφεί με ακολουθία βημάτων το πρόβλημα του 'πολλαπλασιασμού αλά ρωσικά'.
13. Ποιά η πρακτική σημασία του αλγορίθμου του 'πολλαπλασιασμού αλά ρωσικά' ; Πότε γίνεται χρήση αυτού του τρόπου πολλαπλασιασμού δύο ακεραίων ;

### Βιβλιογραφία

1. Ν.Ιωαννίδης, Κ.Μαρινάκης, Σπ.Μπακογιάννης, *Δομημένη Σχεδίαση Προγράμματος*, Εκδόσεις Ελιξ, Αθήνα 1991.
2. Χρήστος Κοίλιας, *Δομές Δεδομένων και Οργανώσεις Αρχείων*, Εκδόσεις Νέων Τεχνολογιών, 1993, Αθήνα.
3. Ιωάννης Μανωλόπουλος, *Δομές Δεδομένων – μία Προσέγγιση με Pascal*, Εκδόσεις Art of Text, Θεσσαλονίκη, 1998.
4. Σκανδάλης κ.α. *Στοιχεία Θεωρίας Αλγορίθμων*, Πανεπιστημιακές Εκδόσεις Κρήτης, Κρήτη, 1990.
5. D. Brunskill and J. Turner, *Understanding Algorithms and Data Structures*, McGraw-Hill, 1996.
6. D. E. Knuth, *The Art of Computer Programming: Fundamental Algorithms*, Vol.1, 3rd edition, Addison Wesley, 1997.
7. M.A. Weiss, *Data Structures and Algorithm Analysis*, 2<sup>nd</sup> edition, Benjamin/Cummings, 1995

### Διευθύνσεις Διαδικτύου

⇒ <http://hissa.ncsl.nist.gov/~black/CRCDict/>

Κόμβος με ευρετήριο όρων για αλγορίθμους, Δομές Δεδομένων και Προβλήματα (Algorithms, Data Structures, and Problems Terms and Definitions for the CRC Dictionary of Computer Science, Engineering and Technology)

⇒ [http://www.ee.uwa.edu.au/~plsd210/ds/ds\\_ToC.html](http://www.ee.uwa.edu.au/~plsd210/ds/ds_ToC.html)

Κόμβος ενός πρότυπου μαθήματος ακαδημαϊκού επιπέδου για Δομές Δεδομένων και Αλγορίθμους με παρουσίαση, εξηγήσεις και κώδικα προγραμμάτων για τις κυριότερες κατηγορίες προβλημάτων.