# Κεφ2: Βασικές έννοιες αλγορίθμων

## Αλγόριθμος

(Ορισμός) Είναι μία

* πεπερασμένη σειρά ενεργειών,
* αυστηρά καθορισμένων και
* εκτελέσιμων σε πεπερασμένο χρόνο, που
* στοχεύουν στην επίλυση ενός προβλήματος

Η σειρά – αλληλουχία των ενεργειών δεν είναι μοναδική.

Η έννοια του αλγορίθμου δεν συνδέεται αποκλειστικά με έννοιες της πληροφορικής.

## Απαραίτητα κριτήρια αλγορίθμων

(πρέπει να τα ικανοποιεί κάθε αλγόριθμος):

* **Είσοδος:** Καμία , μία ή περισσότερες τιμές δεδομένων πρέπει να δίνονται ως είσοδος. Η περίπτωση που δεν δίνονται τιμές δεδομένων εμφανίζεται όταν: ο αλγόριθμος δημιουργεί και επεξεργάζεται κάποιες πρωτογενείς τιμές με την βοήθεια συναρτήσεων παραγωγής τυχαίων αριθμών ή με την βοήθεια άλλων απλών εντολών.
* **Έξοδος:** Ο αλγόριθμος πρέπει να δημιουργεί τουλάχιστον ένα αποτέλεσμα προς τον χρήστη ή προς ένα άλλο αλγόριθμο.
* **Καθοριστικότητα**: Κάθε εντολή πρέπει να καθορίζεται χωρίς αμφιβολία για τον τρόπο εκτέλεσής της. Πχ μία εντολή ζ  χ / ψ πρέπει να λαμβάνει υπ’ όψιν της το γεγονός ότι μπορεί το ψ να είναι μηδέν.
* **Περατότητα:** Ο αλγόριθμος πρέπει να τελειώνει μετά από πεπερασμένα βήματα. Αν δεν τελειώνει μετά από ένα συγκεκριμένο αριθμό βημάτων δεν είναι αλγόριθμος αλλά υπολογιστική διαδικασία.
* **Αποτελεσματικότητα:** Κάθε εντολή ενός αλγορίθμου πρέπει να είναι απλή και εκτελέσιμη (δεν αρκεί να έχει οριστεί).

## Σκοπιές μελέτης των αλγορίθμων

* **Υλικού** (Hardware): Μελετάται το πως επηρεάζεται η ταχύτητα εκτέλεσης ενός αλγορίθμου από τις διάφορες τεχνολογίες υλικού. Δηλαδή από τον τρόπο που τα συστατικά του υπολογιστή είναι δομημένα σε μία ενιαία αρχιτεκτονική (πχ έχει αν κρυφή μνήμη και πόση, ανάλογα με την ταχύτητα της κύριας και δευτερεύουσας μνήμης)
* **Γλωσσών προγραμματισμού** (programming languages): Μελετάται το πώς επηρεάζεται ένας αλγόριθμος από το είδος της γλώσσας προγραμματισμού που χρησιμοποιείται. Πχ το αν θα χρησιμοποιηθεί γλώσσα χαμηλού ή υψηλού επιπέδου αλλάζει την δομή , τον αριθμό των εντολών , ακόμη και την ταχύτητα εκτέλεσης του. Γενικά οι γλώσσες χαμηλότερου επιπέδου είναι ταχύτερες. {ένας αλγόριθμος μπορεί να γραφεί σε διάφορες γλώσσες προγραμματισμού}
* **Θεωρητική** (theoretical): Μελετά το αν υπάρχει ή όχι κάποιος αποδοτικός αλγόριθμος για την επίλυση ενός προβλήματος. Η μελέτη αυτή είναι ιδιαίτερα σημαντική γιατί προσδιορίζει τα όρια της λύσης που θα βρεθεί σε ένα πρόβλημα.
* **Αναλυτική** (analytical): Μελετά τους υπολογιστικούς πόρους που χρειάζεται ένας αλγόριθμος. (πχ το μέγεθος της κύριας-δευτερεύουσας μνήμης που χρειάζεται κλπ)

## Περιγραφή και αναπαράσταση αλγορίθμων

(στην βιβλιογραφία υπάρχουν διάφοροι τρόποι){στο σχολικό τέσσερις}

* **Ελεύθερο κείμενο** (Free text): Αποτελεί τον πιο ανεπεξέργαστο και αδόμητο τρόπο παρουσίασης. Υπάρχει κίνδυνος να οδηγήσει σε μη εκτελέσιμη παρουσίαση, παραβιάζοντας το τελευταίο χαρακτηριστικό των αλγορίθμων το κριτήριο της αποτελεσματικότητας.
* **Διαγραμματικές τεχνικές** (diagramming techniques): Αποτελούν ένα γραφικό τρόπο αναπαράστασης. Μια τεχνική (από τις πιο παλιές και γνωστές ) είναι το διάγραμμα ροής (Flow Chart). Η χρήση διαγρ. τεχνικών δεν είναι η καλύτερη λύση, γι’ αυτό εμφανίζονται όλο και σπανιότερα στην βιβλιογραφία και στην πράξη.
* **Φυσική γλώσσα κατά βήματα** (natural language): Μοιάζει με το Ελεύθερο κείμενο απλά είναι κατά βήματα. Κίνδυνος να παραβιαστεί το κριτήριο της καθοριστικότητας.
* **Κωδικοποίηση** (coding): Δηλαδή με ένα πρόγραμμα γραμμένο είτε με μία ψευδογλώσσα είτε σε κάποιο προγραμματιστικό περιβάλλον που όταν εκτελεσθεί θα δώσει τα ίδια αποτελέσματα με τον αλγόριθμο.

## Διάγραμμα ροής

Είναι στην ουσία μία διαγραμματική τεχνική {βλέπε παραπάνω}.

Αποτελείται από ένα σύνολο γεωμετρικών σχημάτων, όπου το καθένα δηλώνει μία συγκεκριμένη ενέργεια ή λειτουργία.

Τα σχήματα ενώνονται μεταξύ τους με βέλη που δηλώνουν την σειρά εκτέλεσης των ενεργειών αυτών.

Τα κυριότερα σχήματα είναι τα εξής:

* **Έλλειψη**: Η αρχή και το τέλος του αλγορίθμου.
* **Ρόμβος**: Μία συνθήκη με δύο εξόδους ανάλογα με την τιμή της. {αληθής - ψευδής}
* **Ορθογώνιο**: Η εκτέλεση μίας ή περισσοτέρων πράξεων.
* **Πλάγιο παρ/μο**: Η είσοδος και η έξοδος στοιχείων του αλγορίθμου.

## Στοιχεία αλγορίθμου

Ένας αλγόριθμος διαμορφώνεται από τα παρακάτω :

* **Τελεσταίοι**
* **Τελεστές**
* **Εκφράσεις**
* **Εντολές**

{Αναλυτικότερα:}

## Τελεσταίοι (operands)

* **Σταθερές(constants)**: Προκαθορισμένες τιμές που παραμένουν αμετάβλητες σε όλη τη διάρκεια της εκτέλεσης ενός αλγορίθμου.
* **Μεταβλητές(variables)**: Είναι ένα αντικείμενα {δηλαδή λέξεις..} που χρησιμοποιούνται για να παραστήσουν στοιχεία δεδομένων. Η διαφορά τους από τις σταθερές είναι ότι δέχονται μία τιμή που μπορεί να αλλάξει κατά την εκτέλεση ενός αλγορίθμου.

Οι τελεσταίοι (μεταβλητές-σταθερές) διακρίνονται σε τρεις κατηγορίες ανάλογα με το είδος των τιμών που μπορούν να λάβουν.

{βλέπε για αντιπαραβολή το κεφάλαιο 7 όπου οι κατηγορίες είναι τέσσερις}

* **Αριθμητικοί**:   
  Είναι ακέραιες τιμές ή πραγματικές   
  πχ 12, 3.14, -19.99 κλπ.
* **Αλφαριθμητικοί**:   
  Αποτελούνται από σειρές χαρακτήρων μέσα σε εισαγωγικά. Οι χαρακτήρες μπορεί να είναι γράμματα, ψηφία, σημεία στίξης κλπ  
  πχ ‘γειά σας’, ‘μεσος1’, ‘12’ κλπ
* **Λογικοί**: Η τιμές που μπορούν να πάρουν είναι ακριβώς δύο !: αληθής, ψευδής

## Τελεστές (operators)

Είναι σύμβολα που χρησιμοποιούνται στις διάφορες πράξεις. Διακρίνονται σε τρεις κατηγορίες:

|  |  |  |
| --- | --- | --- |
| **Αριθμητικοί:** | **Λογικοί:** | **Συγκριτικοί:** |
| ^ | Oxi (άρνησης) | = (ίσο) |
| \* / div mod | Και (σύζευξης) | <> (διάφορο) |
| + - | Η (διάξευξης) | >= (μεγαλύτερο ή ίσο) κλπ |

## Εκφράσεις (expressions)

Όταν μία τιμή προκύπτει από υπολογισμό τότε αναφερόμαστε σε εκφράσεις. Οι τελεστές μαζί με τους τελεσταίους διαμορφώνουν τις εκφράσεις. Η διεργασία αποτίμησης μίας έκφρασης συνίσταται στην απόδοση τιμών στις μεταβλητές και στην εκτέλεση των πράξεων. Η εκτέλεση των πράξεων εξαρτάται από την ιεραρχία των πράξεων και την χρήση παρενθέσεων. Μία έκφραση μπορεί να αποτελείται από μία μόνο μεταβλητή ή σταθερά μέχρι μία πολύπλοκη μαθηματική παράσταση.

{

Χωρίζονται σε δύο κατηγορίες : τις αριθμητικές (Σχολ. Κεφ. 7) και τις λογικές (Σχολ. Κεφ. 8)

SOS: Οι λογικές εκφράσεις ονομάζονται αλλιώς και: Συνθήκες

Παραδείγματα εκφράσεων

|  |  |
| --- | --- |
| Αριθμητικές : | Λογικές |
| Χ  Χ+2  Χ+C/2^3 | X+2>0  X+2>0 και Χ<> 8  Αληθής και όχι (Χ=3) |

}

## Εντολές

Αποκαλείται κάθε μία λέξη που προσδιορίζει μία σαφή ενέργεια.

Οι εντολές μπορούν να διακριθούν στις τρεις βασικές δομές του δομημένου προγραμματισμού:

{βλέπε και κεφάλαιο 6ο (δομημένος προγραμματισμός)}

* Δομή ακολουθίας
* Δομή επιλογής
* Δομή επανάληψης

Οι εντολές διακρίνονται επίσης σε **δηλωτικές** και σε **εκτελέσιμες**. {δεν αναφέρει περισσότερα το σχολικό περί της διάκρισης αυτής}

Πχ

|  |  |
| --- | --- |
| Δηλωτικές | Εκτελέσιμες |
| Αλγόριθμος | Διάβασε χ |

Άλλες δηλωτικές εντολές:

// Δεδομένα // : Τα δεδομένα εισόδου αν υπάρχουν περιγράφονται στην δεύτερη γραμμή του αλγορίθμου (μετά την εντολή αλγόριθμος)

// Αποτελέσματα // : Τα αποτελέσματα εισόδου δίνονται στην προτελευταία γραμμή (πριν την εντολή τέλος )

## Δομή ακολουθίας

Ονομάζεται και σειριακή ή ακολουθιακή δομή. Αποτελείται από ένα σύνολο εντολών που τοποθετούνται η μία κάτω από την άλλη. Χρησιμοποιείται (από μόνη της) για την επίλυση πολύ απλών προβλημάτων όπου η σειρά εκτέλεσης ενός συνόλου ενεργειών είναι δεδομένη. Χρησιμοποιείται ευρύτατα σε συνδυασμό με άλλες δομές (επιλογής, επανάληψης). Στη δομή αυτή ανήκουν οι εντολές:

* **Εισόδου:** Διάβασε. Η εντολή αυτή συνοδεύεται με το όνομα μίας ή περισσοτέρων μεταβλητών Η λειτουργία της είναι: μετά την ολοκλήρωσή της η μεταβλητή/μεταβλητές θα έχει λάβει τιμή ως περιεχόμενο. Πχ : διάβασε χ,ψ
* **Εξόδου:** Εμφάνισε – Εκτύπωσε. Οι εντολές αυτές εμφανίζουν τα αποτελέσματα στην οθόνη και στον εκτυπωτή αντίστοιχα. Η σύνταξή της εντολής αυτής είναι ανάλογη με του διάβασε.
* **Εκχώρησης τιμής**:  Η γενική μορφή της εντολής είναι : Μεταβλητήέκφραση. Η λειτουργία της είναι: γίνονται οι πράξεις στην έκφραση και το αποτέλεσμά της αποδίδεται , μεταβιβάζεται, εκχωρείται στη μεταβλητή. Ας σημειωθεί ότι δεν πρόκειται για εξίσωση και ότι οι διάφορες γλώσσες προγραμματισμού χρησιμοποιούν διάφορα σύμβολα αντί για το βέλος.
* **Δηλωτικές** : Αλγόριθμος , Τέλος. Ένας αλγόριθμος διατυπωμένος σε ψευδογλώσσα αρχίζει πάντα με τη λέξη Αλγόριθμος συνοδευόμενη με το όνομα του και τελειώνει με την λέξη Τέλος συνοδευόμενη επίσης με το όνομά του.

{Για περισσότερες λεπτομέρειες για τις τρεις πρώτες εντολές ανατρέξτε καλύτερα στο κεφάλαιο 7}

## Δομή επιλογής

Αποτελείται από ένα σύνολο εντολών που εκτελούνται κατά περίπτωση. Η διαδικασία της επιλογής περιλαμβάνει τον έλεγχο κάποιας συνθήκης με δύο δυνατές τιμές (αληθής, ψευδής) και στη συνέχεια την απόφαση εκτέλεσης κάποιας εντολής ανάλογα με τη συνθήκη.

{Για περισσότερα βλέπε κεφάλαιο 8}

## Δομή επανάληψης

Η δομή επανάληψης εφαρμόζεται στις περιπτώσεις όπου μία ακολουθία εντολών πρέπει να γίνει περισσότερες από μία φορές. Εφαρμόζεται σε ένα σύνολο περιπτώσεων που έχουν κάτι κοινό. Οι επαναληπτικές διαδικασίες έχουν τρεις μορφές και συχνά εμπεριέχουν συνθήκες επιλογών.

{βλέπε κεφάλαιο 8}

## Ολίσθηση

Στα κυκλώματα του υπολογιστή τα δεδομένα αποθηκεύονται με δυαδική μορφή , δηλαδή 0 και 1. Έτσι ο αριθμός 5 του δεκαδικού συστήματος αντιστοιχεί στον αριθμό 00000101 του δυαδικού συστήματος (με χρήση 8 bits). Η ολίσθηση διακρίνεται σε ολίσθηση προς τα αριστερά και προς τα δεξιά.

* Προς τα αριστερά : Ο αριθμός 00000101 γράφεται 00001010 (μετακινείται προς τα αριστερά).

Ο ισοδύναμος αριθμός στο δεκαδικό είναι ο 10 (δέκα). Άρα ολίσθηση προς τα αριστερά ισοδυναμεί με διπλασιασμό του αριθμού.

Πχ προς τα αριστερά :

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 5: | 0 | 0 | 0 | 0 | 0 | **1** | **0** | **1** |
| 10: | 0 | 0 | 0 | 0 | **1** | **0** | **1** | 0 |

* Προς τα δεξιά: Ο αριθμός 00000101 γράφεται 00000010 (μετακινείται προς τα δεξιά). Ο ισοδύναμος αριθμός στο δεκαδικό είναι ο 2 (δύο) Άρα ολίσθηση προς τα δεξιά ισοδυναμεί με υποδιπλασιασμό του αριθμού.

Πχ προς τα δεξιά:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 5: | 0 | 0 | 0 | 0 | 0 | **1** | **0** | **1** |
| 2: | 0 | 0 | 0 | 0 | **1** | **0** | **1** | **0** |

## Πολλαπλασιασμός αλά ρωσικά

Η πράξη του πολλαπλασιασμού δύο αριθμών δεν εκτελείται από τον υπολογιστή με τον τρόπο που την εκτελούμε εμείς. Έστω ότι θέλω να πολλαπλασιάσω τους αριθμούς 45 και 19 Οι αριθμοί που πρέπει να πολλαπλασιαστούν γράφονται δίπλα δίπλα και ο πρώτος διπλασιάζεται ενώ ο δεύτερος υποδιπλασιάζεται. Η διαδικασία αυτή επαναλαμβάνεται ώσπου ο δεύτερος να γίνει μονάδα :

|  |  |  |
| --- | --- | --- |
| 45 | 19 | *45* |
| 90 | 9 | *90* |
| 180 | 4 |  |
| 360 | 2 |  |
| 720 | 1 | *720* |

Τελικά το ζητούμενο γινόμενο ισούται με το άθροισμα των στοιχείων της πρώτης στήλης όπου αντίστοιχα στην δεύτερη στήλη υπάρχει περιττός αριθμός. Στο παράδειγμα τα στοιχεία αυτά παρουσιάζονται στην τρίτη στήλη.

Δηλαδή είναι 45+90+720=855

{βλέπε σχολικό για τον αντίστοιχο αλγόριθμο}

1. **Ο υπολογιστής εκτελεί τον πολλαπλασιασμό αλά ρωσικά γιατί:**

Είναι υλοποιείται πιο απλά και γρήγορα με την βοήθεια κυκλωμάτων. Σε αντίθεση με την γνωστή μας διαδικασία πολ/μου απαιτεί μόνο διπλασιασμούς και υποδιπλασιασμούς. Σε επίπεδο λοιπόν κυκλωμάτων ο διπλασιασμός μπορεί να γίνει με μία εντολή ολίσθησης προς τα αριστερά ενώ ο υποδιπλασιασμός με ολίσθηση προς τα δεξιά.

## Δομή ενός αλγόριθμου σε ψευδογλώσσα

Αλγόριθμος όνομα

//Δεδομένα //

Εντολές

//Αποτελέσματα //

Τέλος όνομα

{Στα //δεδομένα// αναγράφουμε δίπλα όλες τις μεταβλητές(ή πίνακες) που εμφανίζονται σε εντολές εισόδου, ενώ στα //αποτελέσματα// όλες τις μεταβλητές (ή πίνακες) που εμφανίζονται σε εντολές εξόδου.

Πχ Να γραφεί αλγόριθμος σε ψευδογλώσσα που υπολογίζει και εμφανίζει το εμβαδά ενός τριγώνου.

Αλγόριθμος Τρίγωνο

//Δεδομένα βάση,ύψος //

Διάβασε βάση,ύψος

Εμβαδόβάση\*ύψος/2

Εμφάνισε Εμβαδό

//Αποτελέσματα Εμβαδό //

Τέλος όνομα

}