ΠΑΝΕΠΙΣΤΗΜΙΟ
ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΣΤΑΤΙΣΤΙΚΗΣ

Αλγόριθμοι

Εισαγωγή

Διδάσκων: Τσούρος Δήμος

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΣΤΑΤΙΣΤΙΚΗΣ

Γενικές Πληροφορίες

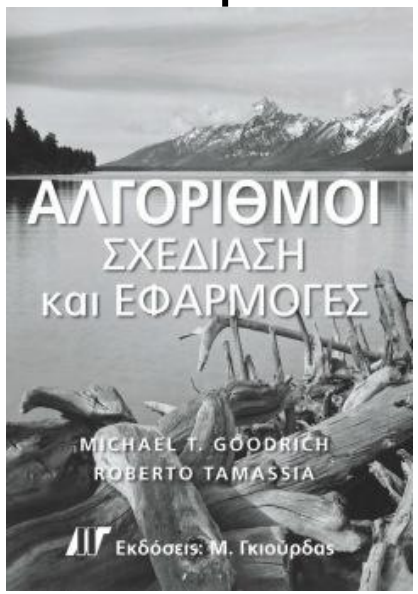
- Διδάσκων: Τσούρος Δήμος
  - Mail: dtsouros@uowm.gr

- Ώρες μαθήματος: Τετάρτη 12:00 – 15:00

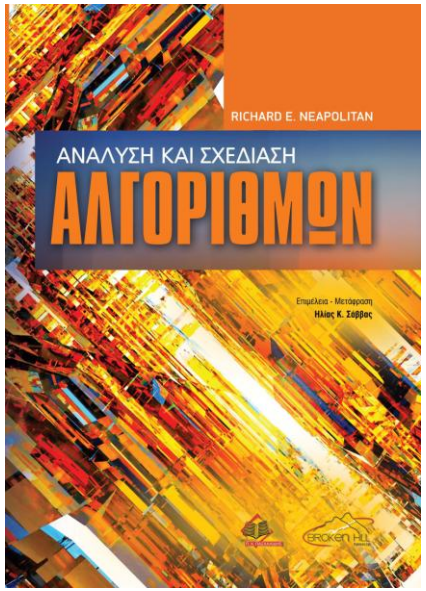
- Ώρες γραφείου:
  - Τρίτη 11:00 – 12:00, 15:00 – 16:00
  - Τετάρτη 15:00 – 16:00

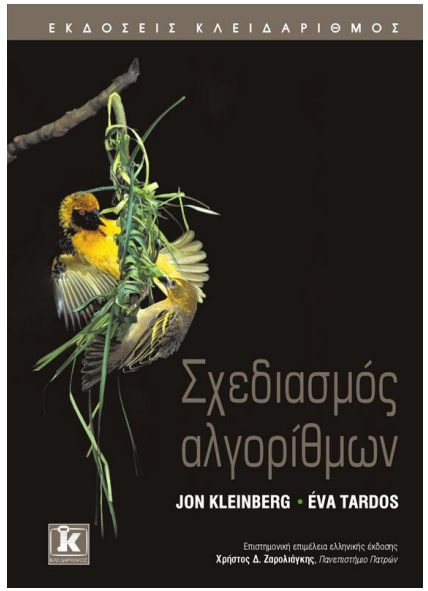
Ύλη μαθήματος:

- Προτεινόμενο Βιβλίο:



- Άλλα βιβλία:





- Διαφάνειες: (θα ανεβαίνουν στο ***eclass.uowm.gr***)

Τρόπος διδασκαλίας

- Διαλέξεις θεωρίας
- Παραδείγματα υλοποίησης και εφαρμογής
- Ασκήσεις σε συγκεκριμένες μεθόδους
- Ερωτήσεις!!!!!!

Στόχος η κατανόηση των εννοιών και των μεθόδων!! Όχι η αποστήθιση..

Τρόπος αξιολόγησης

- **Γραπτή τελική εξέταση (70%)**
  - θα περιλαμβάνει
    - Πολλαπλής επιλογής (με χαμηλή αρνητική βαθμολογία για αποφυγή τυχαίας επιλογής)
    - Ερωτήσεις κατανόησης
    - Ερωτήσεις ανάπτυξης
    - Ασκήσεις
  - Στο τελευταίο μάθημα θα δούμε παραδείγματα ερωτήσεων και ασκήσεων
- **Εργασίες (30%)**
- (Σε συνεννόηση μπορούμε να το κάνουμε 50%-50%)

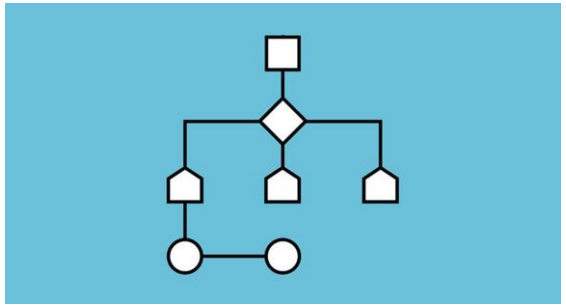
Στόχος η κατανόηση των εννοιών και των μεθόδων!! Όχι η αποστήθιση..

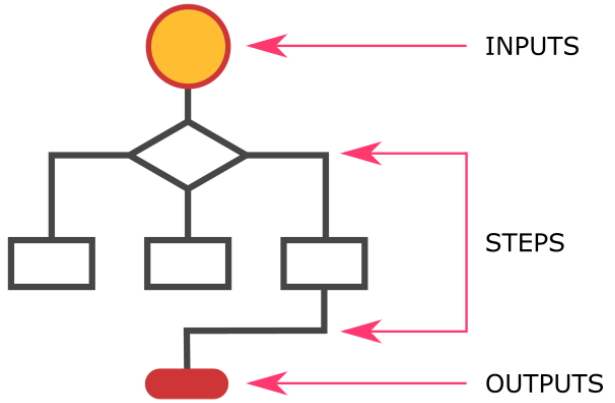
Στόχος

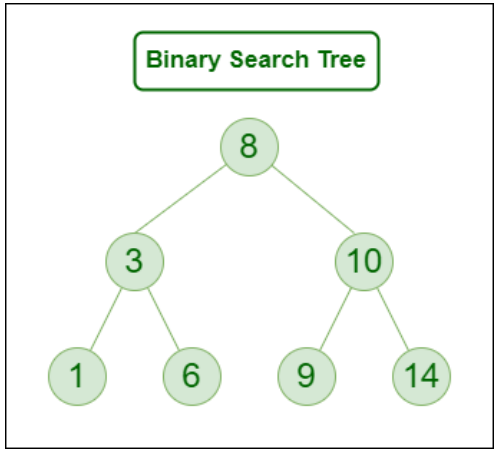
- Κατανόηση των βασικών εννοιών του προγραμματισμού και των αλγορίθμων.
- Εξοικείωση με τις κύριες τεχνικές σχεδίασης αλγορίθμων.
- Ανάλυση αλγορίθμων
- Ποσόστωση επιτυχίας στις εξετάσεις:
  - Μακάρι 100%!

Περιεχόμενα μαθήματος

- Τι είναι ένας αλγόριθμος;
- Ανάλυση αποτελεσματικότητας αλγορίθμων
- Δομές δεδομένων
- Τεχνικές σχεδίασης αλγορίθμων
  - Διαίρει και βασίλευε
  - Άπληστοι αλγόριθμοι
  - Δυναμικός προγραμματισμός
- Δέντρα







Εισαγωγή

Αλγόριθμοι: τρόποι επίλυσης (υπολογιστικών) προβλημάτων

Αλγόριθμοι υπάρχουν παντού

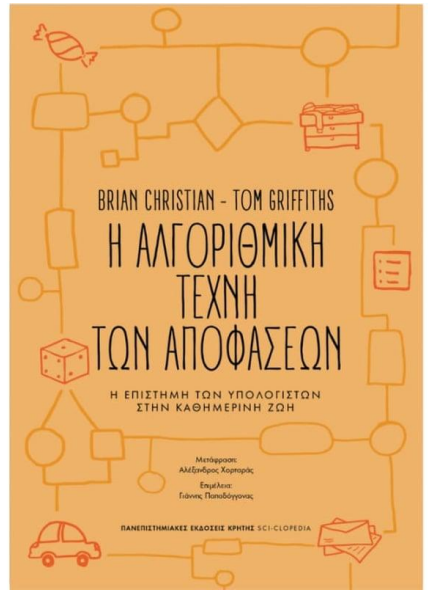
Καθημερινά παραδείγματα:

- χάρτες/πλοήγηση, εύρεση βέλτιστης διαδρομής
- αναζήτηση web,
- προτάσεις ταινιών/βίντεο/...,
- ταξινόμηση email, αναζήτηση

Οργανωτικά παραδείγματα:

- πρόγραμμα εξετάσεων,
- διαχείριση αποθήκης
- Βάρδιες σε νοσοκομεία, εργοστάσια ...

οι αλγόριθμοι είναι τρόποι σκέψης, όχι απλά κώδικας



Μελέτη περίπτωσης: Πώς βρίσκει το web «τις καλύτερες απαντήσεις»

Γενικά βήματα:

- **συλλογή σελίδων,**
- **Αναζήτηση στη βάση δεδομένων,**
- **αντιστοίχιση ερωτήματος,**
- **Κατάταξη**

Περιορισμοί: χρόνος απόκρισης, ακρίβεια, ...

- Χρειάζονται **καλοί** αλγόριθμοι

Αλγόριθμοι: τρόποι επίλυσης (υπολογιστικών) προβλημάτων

Πρόβλημα
- Έννοια βασική, κάτι στο οποίο χρειαζόμαστε μια λύση
- π.χ. το οικονομικό πρόβλημα

Υπολογιστικά Προβλήματα (ΥΠ):
- Αυτά που λύνονται με χειρισμούς αριθμών (αριθμητικές πράξεις, συγκρίσεις, εγγραφές και διαγραφές αριθμών, κλπ)
- Π.χ. το πρόβλημα ταξινόμησης ενός πίνακα
  - Όχι το οικονομικό πρόβλημα

Μαθηματική περιγραφή ΥΠ:
- Γίνεται συνήθως με **παραμέτρους εισόδου** και **άγνωστες μεταβλητές**
- Ζητείται ο υπολογισμός των αγνώστων ως συνάρτηση των παραμέτρων

Κατηγορίες υπολογιστικών προβλημάτων

- **Κατηγορίες ΥΠ**
  - Προβλήματα αναζήτησης (search problems)
  - Προβλήματα δόμησης (structuring problems)
  - Προβλήματα κατασκευής (construction problems)
  - Προβλήματα απόφασης (decision problems)
  - Προβλήματα βελτιστοποίησης (optimization problems)
  - Προβλήματα προσαρμογής (adaptation problems)

Δυσκολίες επίλυσης προβλημάτων

Εννοιολογικά δύσκολο ΥΠ

- Δεν υπάρχει αλγόριθμος για αυτό το πρόβλημα γιατί δεν το κατανοούμε καλά

Αναλυτικά δύσκολο ΥΠ.

- Υπάρχει αλγόριθμος για αυτό το πρόβλημα, αλλά δεν ξέρουμε πόσος χρόνος απαιτείται για την επίλυσή του

Υπολογιστικά δύσκολο πρόβλημα

- Υπάρχει αλγόριθμος για αυτό το πρόβλημα, αλλά ο χρόνος επίλυσης είναι πολύ μεγάλος

Υπολογιστικά άλυτο πρόβλημα

- Δεν υπάρχει αλγόριθμος για αυτό το πρόβλημα, γιατί δεν μπορεί να υπάρξει τέτοιος αλγόριθμος

Αφαίρεση και μοντελοποίηση

- **Μοντελοποίηση προβλήματος**
  - **Πρόβλημα** (ανεπίσημη περιγραφή) → **Προδιαγραφή** (επίσημη περιγραφή)
  - **Αφαίρεση:** Κρατάμε μόνο ό,τι χρειάζεται! Αγνόησε τις μη χρήσιμες λεπτομέρειες
- **Ορίζουμε**:
  - είσοδο
  - έξοδο
  - τι θεωρείται έγκυρη είσοδος, περιορισμοί
  - Σχέση μεταξύ εισόδου και εξόδου (τι επεξεργασία χρειάζεται)
- Παράδειγμα: «Βρες το μέγιστο σε λίστα A[1..n]»
  - είσοδος: A[1..n],
  - έξοδος: μια τιμή,
  - επεξεργασία: η έξοδος είναι το μέγιστο της εισόδου

Η μοντελοποίηση είναι το πρώτο βήμα πριν γράψουμε αλγόριθμο ή πρόγραμμα.

Αφαίρεση και μοντελοποίηση: Παράδειγμα 1

Ανάθεση καθηγητών σε μαθήματα

- **Είσοδος:** λίστα μαθημάτων, διαθέσιμοι καθηγητές, γνώσεις καθηγητών, ώρες διδασκαλίας
- **Έξοδος:** αντιστοίχιση μαθημάτων–καθηγητών
- **Επεξεργασία:** εύρεση ανάθεσης ώστε
  - κάθε καθηγητής να διδάσκει μαθήματα που γνωρίζει
  - και να διδάσκει το πολύ ένα μάθημα ανά ώρα

Αφαίρεση: Κρατάμε μόνο ό,τι χρειάζεται! Δεν μας νοιάζουν τα ονόματα των καθηγητών, οι τίτλοι των μαθημάτων, το όνομα του πανεπιστημίου, ...

Αφαίρεση και μοντελοποίηση: Παράδειγμα 2

Προγραμματισμός εξετάσεων

- **Είσοδος:** μαθήματα, φοιτητές, μαθήματα κάθε εξαμήνου, αίθουσες, διαθέσιμες ώρες
- **Έξοδος:** πρόγραμμα εξετάσεων
- **Επεξεργασία:** ανάθεση ημερομηνιών και αιθουσών ώστε
  - Να εξετάζεται ένα μάθημα ανά αίθουσα κάθε ώρα
  - Να εξετάζεται ένα μάθημα κάθε εξαμήνου κάθε ώρα

Αφαίρεση: Κρατάμε μόνο ό,τι χρειάζεται! Δεν μας νοιάζουν τα ονόματα των φοιτητών, οι τίτλοι των μαθημάτων, το όνομα του πανεπιστημίου, ...

Στιγμιότυπο ή περίπτωση (instance) ΥΠ

Ένα πρόβλημα περιγράφεται γενικά, π.χ.

- το πρόβλημα ανάθεσης καθηγητών σε μαθήματα
- το πρόβλημα ταξινόμησης ενός πίνακα

Στιγμιότυπο (ή περίπτωση) ενός Υπολογιστικού Προβλήματος (ΥΠ) ονομάζεται ο συνδυασμός του προβλήματος με μια συγκεκριμένη είσοδο/αρχικοποίηση των παραμέτρων του, δηλαδή τα δεδομένα που το καθορίζουν.

Για την επίλυση ενός στιγμιότυπου του προβλήματος εφαρμόζεται ένας αλγόριθμος (μέθοδος επίλυσης) που, με βάση τη συγκεκριμένη είσοδο, παράγει μια λύση ή αποφαίνεται ότι δεν υπάρχει λύση.

Ο αλγόριθμος είναι και αυτός **γενικός!** Μπορεί να εφαρμοστεί για την παραγωγή λύσης σε οποιοδήποτε στιγμιότυπο του προβλήματος που καλείται να λύσει

Πρόβλημα και στιγμιότυπο: Παράδειγμα

Πρόβλημα: Ταξινόμηση πίνακα αριθμών

Παράμετροι εισόδου:
- πίνακας A[1..n]

Άγνωστες μεταβλητές:
- νέος πίνακας A'[1..n]

Ζητούμενο:
- υπολογισμός A' που περιέχει τα ίδια στοιχεία με τον A σε αύξουσα σειρά.

Παραδείγματα στιγμιότυπων και λύσεων τους**:**
- **Στιγμιότυπο 1:** A=[5,2,9,1] – **Λύση**: A'=[1,2,5,9]
- **Στιγμιότυπο 2:** A=[2,6,5,3,8,1,2] – **Λύση**: A'=[1,2,2,3,5,6,8]

Πρόβλημα και στιγμιότυπο: Παράδειγμα

Πρόβλημα: Αναζήτηση σε πίνακα αριθμών

Παράμετροι εισόδου:
- πίνακας A[1..n], στοιχείο-στόχος x

Άγνωστες μεταβλητές:
- Μεταβλητή i με τιμή τέτοια ώστε A[i] = x (ή -1 αν δεν υπάρχει)

Ζητούμενο:
- Αναζήτηση θέσης του στοιχείου x στον πίνακα A

Παραδείγματα στιγμιότυπων και λύσεων τους**:**
- **Στιγμιότυπο 1:** A=[5,2,9,1], x=2 – **Λύση**: i=2
- **Στιγμιότυπο 2:** A=[5,2,9,1], x=8 – **Λύση**: i=-1
- **Στιγμιότυπο 3:** A=[2,6,5,3,8,1,2], x=8 – **Λύση**: i=5

Τι είναι Αλγόριθμος

Αλγόριθμος είναι ένας ***συστηματικός*** τρόπος επίλυσης προβλημάτων

- Επίλυση οποιουδήποτε στιγμιότυπου ενός προβλήματος με τον ίδιο τρόπο.
  - Παράγει **αποτέλεσμα** (έξοδο) από **δεδομένα** (είσοδο).
- Βήμα-βήμα διαδικασία που πρέπει να ακολουθηθεί για την επίλυση ενός προβλήματος.
  - Η **λογική επίλυσης** του.
  - Κάθε βήμα πρέπει να είναι σαφές και εκτελέσιμο.
- Οι **αλγόριθμοι** (algorithms) ή αλγοριθμική (algorithmics) είναι κάτι παραπάνω από ένας απλός κλάδος της πληροφορικής: είναι ο πυρήνας της.
- Υπάρχει από την αρχαιότητα (π.χ. Ευκλείδειος αλγόριθμος για το μέγιστο κοινό διαιρέτη-ΜΚΔ).

Τι είναι Αλγόριθμος

Αλγόριθμος: Διαδικασία επίλυσης (υπολογιστικών) προβλημάτων διατυπωμένη σε μαθηματική μορφή

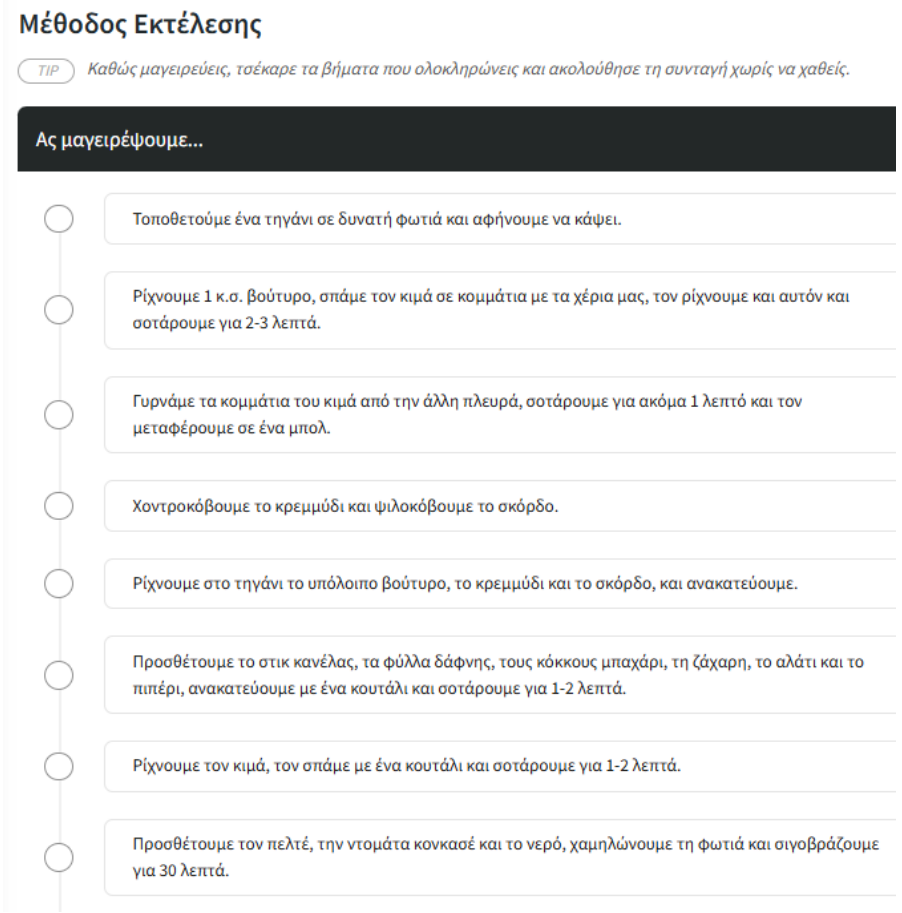
Ένας αλγόριθμος έχει συγκεκριμένες ιδιότητες – χαρακτηριστικά. Στη βιβλιογραφία υπάρχουν πολλοί ισοδύναμοι ορισμοί. Τα βασικά χαρακτηριστικά που ξεχωρίζουμε:

- Δεν περιέχει **λογικές αντιφάσεις**
- Λύνει **κάθε** στιγμιότυπο του προβλήματος
- Τελειώνει μετά από **πεπερασμένο** αριθμό πράξεων
- Δεν περιέχει εκφράσεις, οι οποίες ερμηνεύονται με **υποκειμενικό** τρόπο – Όλα τα βήματα είναι σαφή

Αλγόριθμος: Παράδειγμα

Πρόβλημα: Φτιάξε μακαρόνια με κιμά

- Δεν είναι υπολογιστικό πρόβλημα!
- Όμως η μέθοδος επίλυσής του είναι ένας αλγόριθμος.
- Είσοδος: Για πόσους
- Έξοδος: φαΐ



...

Αλγόριθμος: Παράδειγμα

Πρόβλημα: Ταξινόμηση πίνακα αριθμών

Παράμετροι εισόδου:
- πίνακας A[1..n]

Άγνωστες μεταβλητές:
- νέος πίνακας A'[1..n]

Ζητούμενο:
- υπολογισμός A' που περιέχει τα ίδια στοιχεία με τον A σε αύξουσα σειρά.

Αλγόριθμος

1. Πάρε τον πίνακα A[1..n].
2. Για κάθε στοιχείο από τη δεύτερη θέση μέχρι το τέλος (θέση i):
    1. Σύγκρινε το με τα στοιχεία που βρίσκονται πριν από αυτό (A[j], όπου j<i).
        1. Όσο το προηγούμενο στοιχείο είναι μεγαλύτερο:
        2. Μετακίνησε το προηγούμενο στοιχείο μία θέση δεξιά.
        3. Μείωσε το j κατά 1.
        4. Όταν βρεις μικρότερο ή φτάσεις στην αρχή, τοποθέτησε το A[i] στη σωστή θέση.
    2. Συνέχισε μέχρι να περάσουν όλα τα στοιχεία.

Αλγόριθμος: Παράδειγμα

Πρόβλημα: Αναζήτηση σε πίνακα αριθμών

Παράμετροι εισόδου:
- πίνακας A[1..n], στοιχείο-στόχος x

Άγνωστες μεταβλητές:
- Μεταβλητή i με τιμή τέτοια ώστε A[i] = x (ή -1 αν δεν υπάρχει)

Ζητούμενο:
- Αναζήτηση θέσης του στοιχείου x στον πίνακα A

Αλγόριθμος

1. Πάρε τον πίνακα A[1..n] και το στοιχείο-στόχο x.
2. Για κάθε θέση i από 1 μέχρι n:
    1. Σύγκρινε το A[i] με το x.
    2. Αν είναι ίσα:
        1. Επιστροφή i
3. Αν τελειώσει ο πίνακας χωρίς να βρεθεί το στοιχείο:
    1. Επιστροφή -1.

Βασικές ιδιότητες αξιολόγησης: Ορθότητα, περατότητα και μια ιδέα για «κόστος»

- Ορθότητα: για κάθε έγκυρη είσοδο, η έξοδος είναι σωστή
- Περατότητα: «κάποια στιγμή σταματά», για όλες τις *έγκυρες* εισόδους
- Κόστος (διαισθητικά): πόσα βήματα χρειάζεται για n στοιχεία;
- Παράδειγμα:
  - γραμμική αναζήτηση σε πίνακα ≈ ελέγχει έως n θέσεις **«στην χειρότερη περίπτωση»**
  - Αν έχουμε ταξινομημένο πίνακα, πως μπορούμε να κάνουμε «καλύτερη»/πιο γρήγορη αναζήτηση;

Διαφορά Αλγορίθμου – Προγράμματος

Αλγόριθμος:

- Αφηρημένη, βήμα-βήμα διαδικασία για την επίλυση ενός προβλήματος.
- Περιγράφεται με φυσική γλώσσα, ψευδοκώδικα ή διαγράμματα.
- Είναι ανεξάρτητος από υπολογιστή ή γλώσσα προγραμματισμού.

Πρόγραμμα:

- Συγκεκριμένη υλοποίηση ενός αλγορίθμου σε γλώσσα προγραμματισμού.
- Αποτελείται από εντολές που εκτελεί ένας υπολογιστής.
- Συνδέεται με λεπτομέρειες (π.χ. σύνταξη γλώσσας, χειρισμός μνήμης, αποτελεσματικότητα τρόπων υλοποίησης).

Αλγόριθμος και πρόγραμμα: Παράδειγμα

Αλγόριθμος για ταξινόμηση πίνακα

1. Πάρε τον πίνακα A[1..n].
2. Για κάθε στοιχείο από τη δεύτερη θέση μέχρι το τέλος (θέση i):
    1. Σύγκρινε το με τα στοιχεία που βρίσκονται πριν από αυτό (A[j], όπου j<i).
        1. Όσο το προηγούμενο στοιχείο είναι μεγαλύτερο:
        2. Μετακίνησε το προηγούμενο στοιχείο μία θέση δεξιά.
        3. Μείωσε το j κατά 1.
        4. Όταν βρεις μικρότερο ή φτάσεις στην αρχή, τοποθέτησε το A[i] στη σωστή θέση.
    2. Συνέχισε μέχρι να περάσουν όλα τα στοιχεία.

Πρόγραμμα στη γλώσσα προγραμματισμού python

```
def insertion_sort(A):
    # Για κάθε στοιχείο από τη 2η θέση ως το τέλος
    for i in range(1, len(A)):
        key = A[i]              # θυμόμαστε την τιμή
        j = i - 1
        # Όσο το προηγούμενο στοιχείο είναι μεγαλύτερο
        while j >= 0 and A[j] > key:
            A[j+1] = A[j]       # μετακίνηση δεξιά
            j -= 1
        A[j+1] = key            # τοποθέτηση στη σωστή θέση
    return A
```

Αλγόριθμος και πρόγραμμα: Παράδειγμα

Αλγόριθμος για αναζήτηση σε πίνακα

1. Πάρε τον πίνακα A[1..n] και το στοιχείο-στόχο x.
2. Για κάθε θέση i από 1 μέχρι n:
    1. Σύγκρινε το A[i] με το x.
    2. Αν είναι ίσα:
        1. Επιστροφή i
3. Αν τελειώσει ο πίνακας χωρίς να βρεθεί το στοιχείο:
    1. Επιστροφή -1.

Πρόγραμμα στη γλώσσα προγραμματισμού python

```python
def linear_search(A, x):
    # Για κάθε θέση του πίνακα
    for i in range(len(A)):
        if A[i] == x:          # αν βρέθηκε το στοιχείο
            return i           # επιστρέφουμε τη θέση
    return -1                  # δεν βρέθηκε
```

Σημαντικά ερωτήματα

- Ποιες είναι οι βασικές έννοιες των αλγορίθμων και του προγραμματισμού;
- Ποιοι είναι οι τρόποι αναπαράστασης των αλγορίθμων;
  - Είδαμε μια βασική βηματική αναπαράσταση.
  - Υπάρχουν διάφορες τυπικές μορφές αναπαράστασης.
- Πως συγκρίνουμε αλγορίθμους;
  - Μπορούμε να ξέρουμε αν ένας αλγόριθμος είναι καλύτερος από έναν άλλο χωρίς να τους προγραμματίσουμε;
- Ποιος είναι ο καλύτερος αλγόριθμος για την επίλυση ενός ΥΠ;
  - Υπάρχει ένας καλύτερος αλγόριθμος για κάθε συγκεκριμένο ΥΠ;
- Υπάρχουν βασικές αρχές για τη σχεδίαση (αποδοτικών) αλγορίθμων;

Η ύλη του μαθήματος καλύπτει τα παραπάνω