

Ιόνιο Πανεπιστήμιο – Τμήμα Πληροφορικής  
Εισαγωγή στην Επιστήμη των Υπολογιστών  
2016-17

# Αλγόριθμοι και Δομές Δεδομένων (I)

(εισαγωγικές έννοιες)

<http://mixstef.github.io/courses/csintro/>

Μ.Στεφανιδάκης



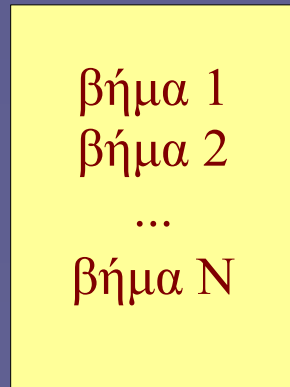
# Τι είναι αλγόριθμος;

- Αλγόριθμοι

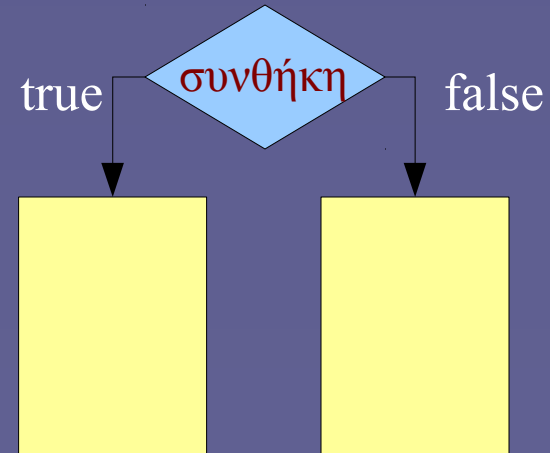
- “Βήμα προς βήμα μέθοδος για την επίλυση ενός προβλήματος”
  - Ανεξάρτητη από το υπολογιστικό σύστημα!
- Τυπικός ορισμός:
  - Μια διαδικασία με πεπερασμένο αριθμό διατεταγμένων βημάτων
    - πιθανώς με επαναλήψεις και συνθήκες
  - η οποία επιλύει ένα συγκεκριμένο πρόβλημα
    - μετασχηματισμού εισόδων σε εξόδους
  - σε πεπερασμένο χρόνο

# Βασικές αλγοριθμικές δομές

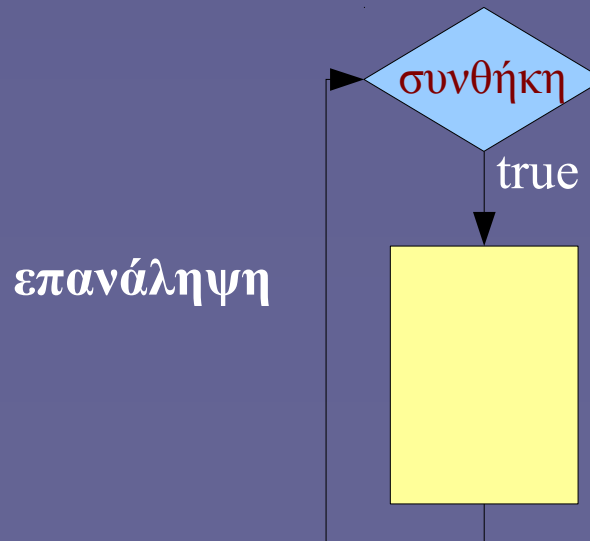
- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές



ακολουθία



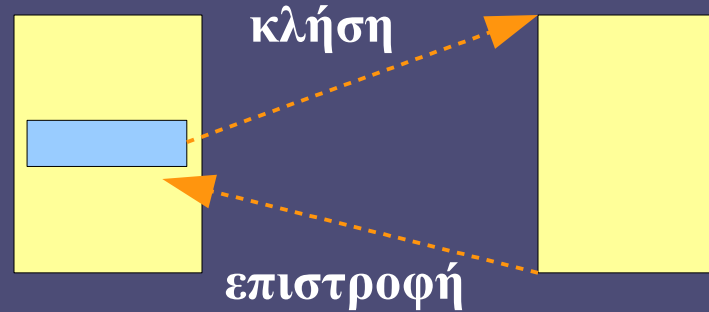
απόφαση  
υπό συνθήκη



επανάληψη

# Υποπρογράμματα (υποαλγόριθμοι)

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές



- Επαναχρησιμοποίηση
  - κλήση συναρτήσεων
- Ευκολότερη κατανόηση
  - σημαντικό όσο και η απόδοση!

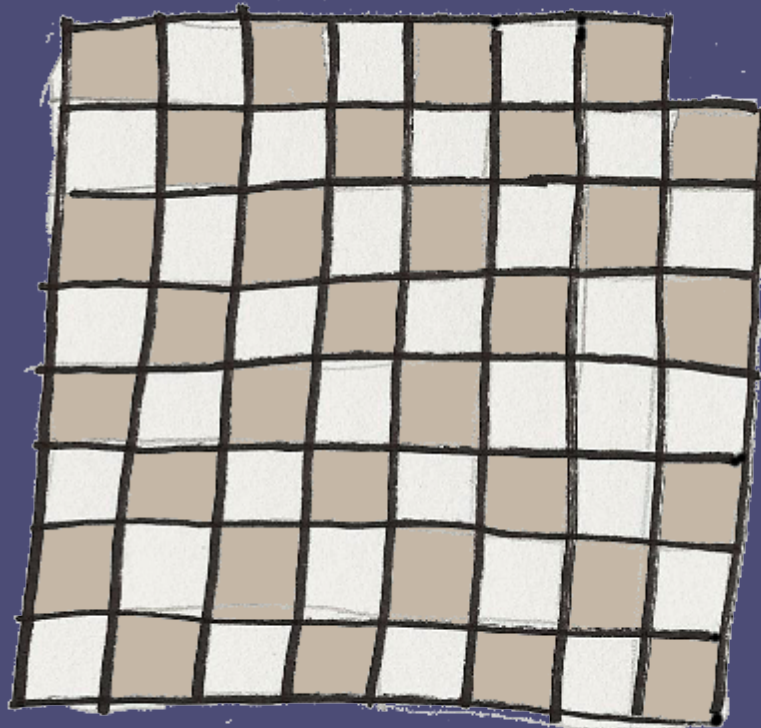
# Αναδρομή: ένα κομψό εργαλείο

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές

- Το γενικότερο πλαίσιο: μείωση ενός προβλήματος σε πολλά υποπροβλήματα
  - Και στη συνέχεια, συνδυασμός των μερικών λύσεων
- Αναδρομή
  - Μία συνάρτηση καλεί τον εαυτό της
    - δημιουργεί μικρότερα υποπροβλήματα
  - Μέχρι να φτάσει σε μια βασική περίπτωση
    - με άμεσο υπολογισμό του (μερικού) αποτελέσματος
  - Ακολουθούν επιστροφές με συνδυασμό των μερικών αποτελεσμάτων
    - μέχρι το τελικό αποτέλεσμα

# Αναδρομή: ένα παράδειγμα

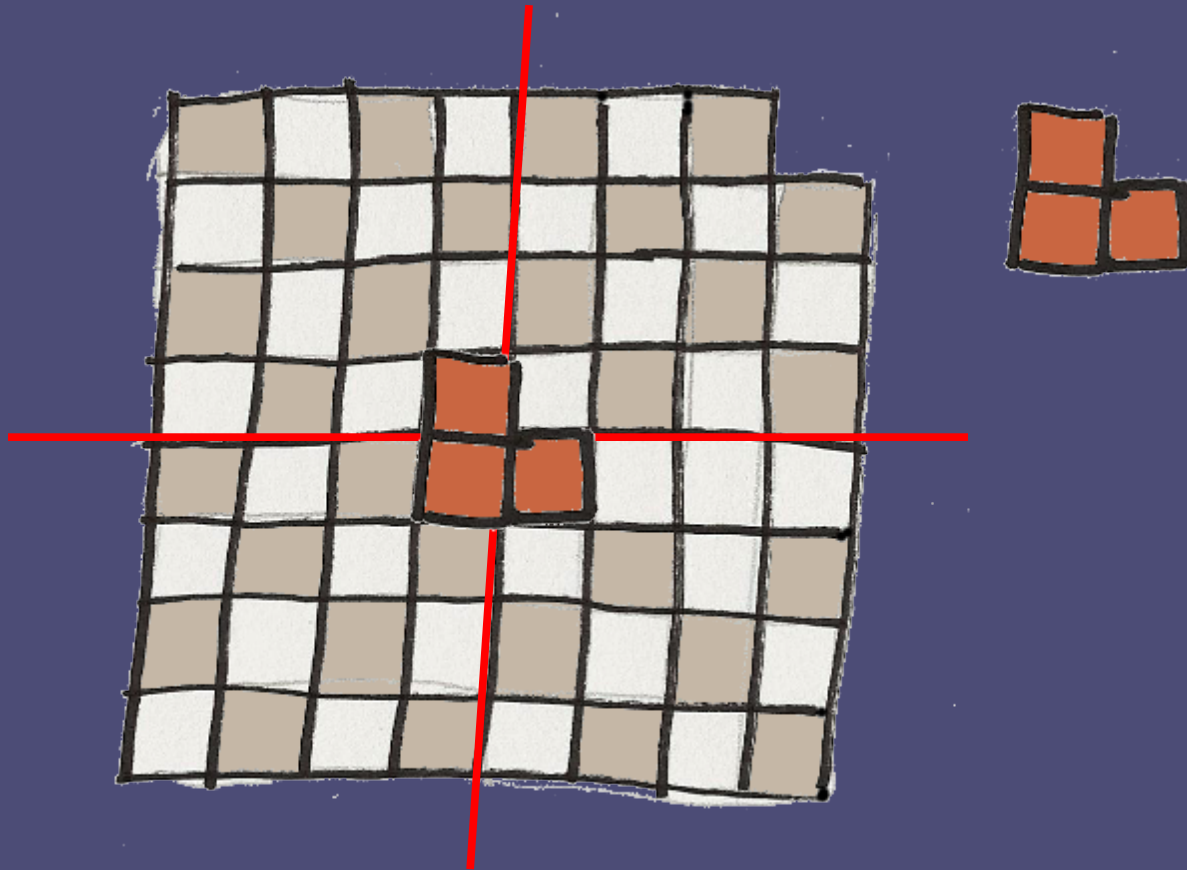
- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές



- Πώς θα καλύψουμε (αν μπορούμε) τη σκακιέρα με σχήματα τύπου L;

# Αναδρομή: ένα παράδειγμα

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές



- Βλέπετε τη λύση; Είναι η αναδρομή!

# Αναδρομή

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές

- Αλγοριθμικά “κομψή” λύση αλλά
  - Πιθανή επιβάρυνση κατά την κλήση των συναρτήσεων
  - Και μεγαλύτερη χρήση πόρων για διατήρηση προηγούμενων καταστάσεων
- Συχνά πρέπει να μετατρέψουμε την αναδρομή σε επανάληψη
  - Ευτυχώς αυτό είναι πάντα δυνατό
    - ακόμα κι αν οδηγεί σε λιγότερο κομψή διατύπωση του αλγορίθμου
    - με τη βοήθεια δομών δεδομένων που μιμούνται τη λειτουργία της αναδρομής



# Πολυπλοκότητα ενός αλγορίθμου

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα

- Για την κατανόηση της απόδοσής του
  - Χρειαζόμαστε ένα βασικό μέγεθος
  - Που θα εστιάζει στη μεγάλη εικόνα
    - Την αύξηση του χρόνου εκτέλεσης ανάλογα με το μέγεθος του προβλήματος (δεδομένων εισόδου)
    - Ανεξάρτητα από την ταχύτητα της γλώσσας προγραμματισμού
    - Ανεξάρτητα από την ταχύτητα του υλικού
  - Προσοχή: δεν ενδιαφερόμαστε για απόλυτους χρόνους
    - Αντιθέτως, υπολογίζουμε πόσες φορές εκτελείται μια βασική λειτουργία!
    - Σε σχέση με το μέγεθος των δεδομένων εισόδου

# Ο ασυμπτωτικός συμβολισμός $O(\dots)$

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα

- **Big O notation**

- Έκφραση κλάσης πολυπλοκότητας σε σχέση με το μέγεθος  $n$  των δεδομένων εισόδου
  - αφαιρώντας σταθερούς παράγοντες
- Τυπικά
  - $O(g(n))$ , είναι ένα σύνολο συναρτήσεων
  - $f(n)$  ανήκει στο  $O(g(n))$  εάν υπάρχει  $n_0$  και θετική σταθερά  $c$  έτσι ώστε  $f(n) \leq cg(n)$  για κάθε  $n \geq n_0$
- Πρακτικά
  - $O(g(n))$ , είναι οι συναρτήσεις που αυξάνονται με αργότερο ρυθμό από το  $g(n)$  - άρα είναι αποδοτικότερες!
- Παράδειγμα
  - Τα  $n^2$ ,  $3n^2$ ,  $85.8n^2 + 3.44$  ανήκουν όλα στο  $O(n^2)$

# Ο ασυμπτωτικός συμβολισμός $O(\dots)$

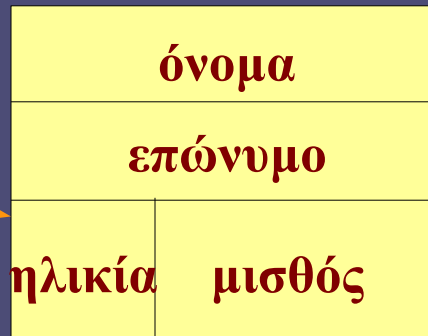
- Αλγόριθμοι
  - Βασικές αλγοριθμικές δομές
  - Πολυπλοκότητα
- Αν για παράδειγμα η πολυπλοκότητα ενός αλγορίθμου είναι  $O(n^2)$ 
    - Αυτό σημαίνει ότι για μέγεθος δεδομένων  $n$  θα εκτελεστεί αριθμός λειτουργιών της τάξης του  $n^2$
    - Και ότι ένας αλγόριθμος  $O(n)$  είναι αποδοτικότερος!
    - Πολυωνυμικά προβλήματα
      - $O(\log n)$ ,  $O(n)$ ,  $O(n^2)$ ,  $O(n^k)$
      - Συνήθως επιλύσιμα
    - Μη πολυωνυμικά προβλήματα
      - $O(k^n)$  ή  $O(n!)$
      - Γενικά, μη επιλύσιμα

# Η βασική μονάδα δεδομένων

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων

- Η “εγγραφή” (record ή structure)
  - Μία αυτοτελής ομάδα δεδομένων
  - Με συγκεκριμένη μορφή αποθήκευσης
    - Κάθε μέλος της ομάδας βρίσκεται σε καθορισμένη θέση (διεύθυνση) μέσα στην εγγραφή
  - Ως “κόμβος” δεδομένων σε πιο σύνθετες δομές δεδομένων

+200 bytes



όνομα	
επώνυμο	
ηλικία	μισθός

# Πίνακες (arrays)

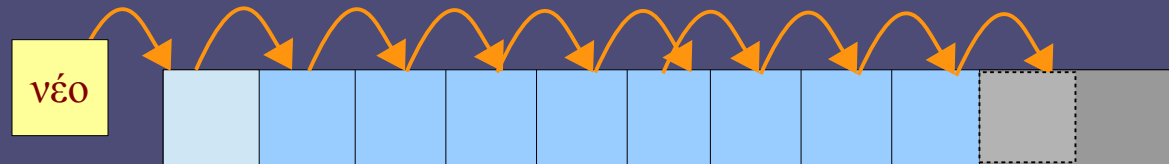
- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες

- Ακολουθία όμοιων εγγραφών
  - Σε συνεχόμενες θέσεις μνήμης
  - Πίνακας μιας διάστασης
  - Ο υπολογισμός της θέσης (διεύθυνσης) του  $i$ -οστού στοιχείου είναι άμεσος
    - Αρχή πίνακα +  $i$  \* μέγεθος εγγραφής
    - Θυμηθείτε: το  $i$  ξεκινά από το 0!
  - Είναι δυνατή η υλοποίηση πινάκων πιο πολλών διαστάσεων
    - Π.χ. για πίνακα δύο διαστάσεων, ως συνεχόμενες σειρές στη μνήμη
    - Πώς υπολογίζεται στην περίπτωση αυτή η διεύθυνση του στοιχείου  $_{i,j}$  ;

# Λειτουργίες σε πίνακες

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες

- Λήψη στοιχείου  $i$ 
  - Σταθερή πολυπλοκότητα  $O(1)$ 
    - Εφόσον ο υπολογισμός της θέσης του κάθε στοιχείου είναι άμεσος
- Προσθήκη στοιχείου (στο τέλος)
  - $O(1)$  – θεωρώντας ότι υπάρχει χώρος
    - Άμεσος υπολογισμός θέσης νέου στοιχείου
- Εισαγωγή στοιχείου (π.χ. στην αρχή)
  - $O(n)$  – θα πρέπει να μετατοπιστούν τα ήδη υπάρχοντα στοιχεία!



# Αναζήτηση σε πίνακα

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες
- Αναζήτηση

- Αναζήτηση στοιχείου με κάποιες ιδιότητες
  - Τιμές-κλειδιά
  - Εύρεση θέσης (i) στοιχείου – αν υπάρχει!
- Εάν τα στοιχεία δεν είναι ταξινομημένα...
  - Με βάση τις αναζητούμενες τιμές-κλειδιά
- ...τότε πρέπει να διασχίσουμε τον πίνακα σειριακά
  - από τη μία άκρη προς την άλλη (διάσχιση πίνακα)
  - συγκρίνοντας κάθε στοιχείο που συναντάμε
  - $O(n)$  – OK για μικρούς πίνακες



# Δυαδική Αναζήτηση (binary search)

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες
- Αναζήτηση

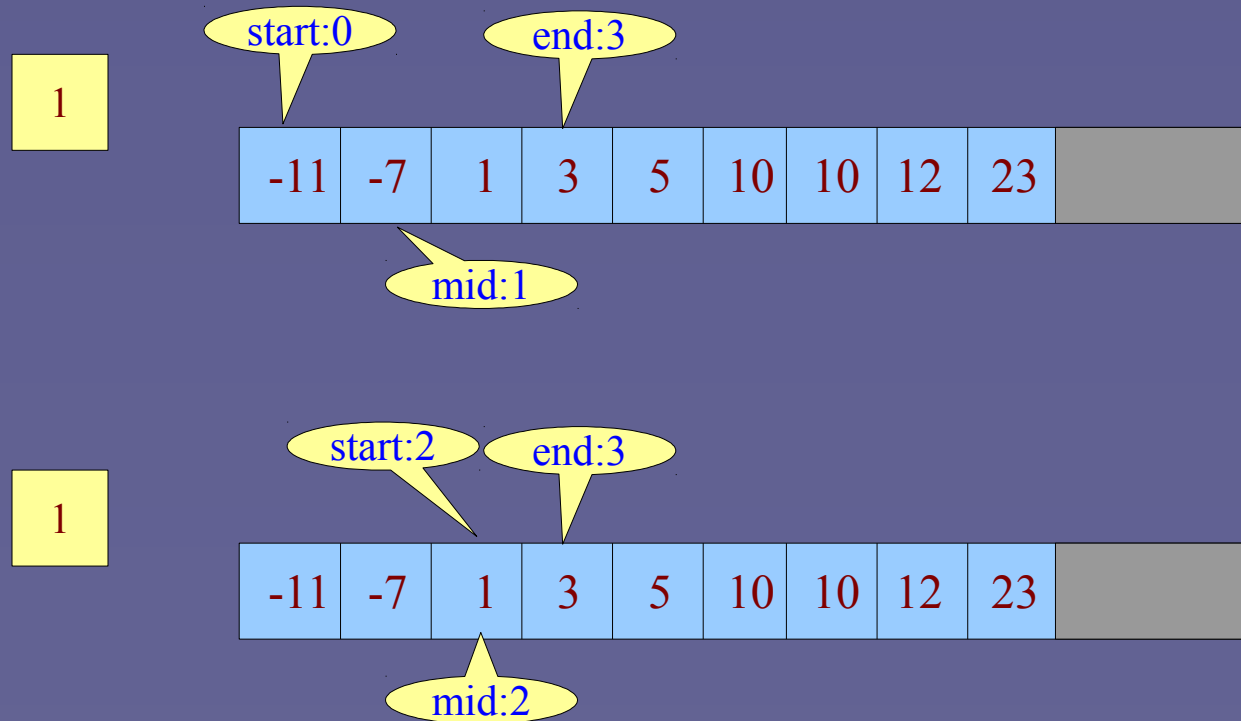
- Εάν τα στοιχεία είναι ταξινομημένα...
  - Με βάση τις αναζητούμενες τιμές-κλειδιά
- ...τότε η αναζήτηση γίνεται πολύ πιο αποδοτική!
  - Χωρίζοντας τον πίνακα στη μέση
  - Και ψάχνοντας μόνο ένα από τα δύο μέρη
    - Μικρότερο υποπρόβλημα!
  - Επαναλαμβάνουμε τη διαδικασία μέχρι την εύρεση
    - ή μέχρι να φανεί ότι δεν υπάρχει αυτό που ψάχνουμε
  - $O(\log_2 n)$ – η μόνη βιώσιμη λύση για μεγάλους πίνακες!





# Δυαδική Αναζήτηση (binary search)

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες
- Αναζήτηση



# Ταξινόμηση

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες
- Αναζήτηση
- Ταξινόμηση

- Συνήθης λειτουργία σε πίνακες
  - Με βάση τιμές κλειδιών (μέρους στοιχείου πίνακα)
  - Ως βάση για την εφαρμογή άλλων αλγορίθμων
    - Όπως η δυαδική αναζήτηση που είδαμε προηγουμένως
  - Δεν υπάρχει μοναδικός αλγόριθμος ταξινόμησης με την βέλτιστη πολυπλοκότητα
    - Εξαρτάται από τη μέχρι τώρα διάταξη των δεδομένων
- Στη συνέχεια θα δούμε ορισμένους μόνο από το σύνολο των αλγορίθμων ταξινόμησης

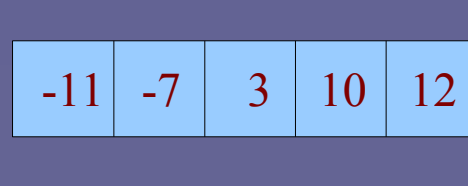
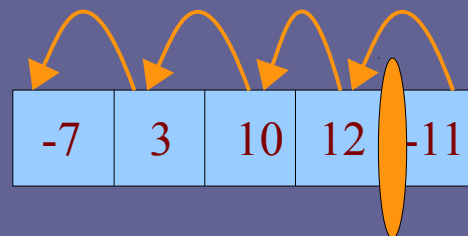
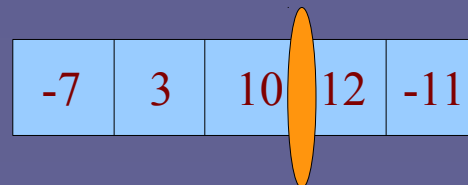
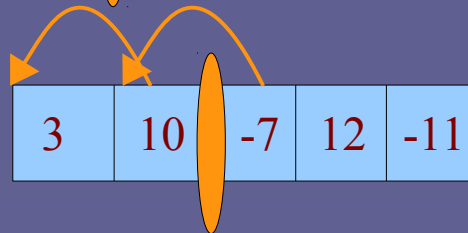
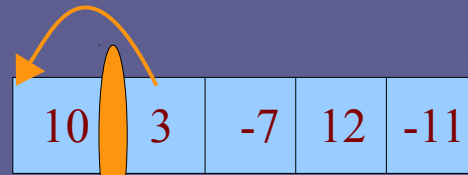
# Ταξινόμηση παρεμβολής (insertion sort)

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες
- Αναζήτηση
- Ταξινόμηση

- Ο πίνακας χωρίζεται σε δύο μέρη
  - Το ταξινομημένο μέρος
    - Αρχικά περιέχει το πρώτο στοιχείο του πίνακα
  - Και το αταξινόμητο
    - Τα υπόλοιπα στοιχεία
- Κάθε στοιχείο του αταξινόμητου
  - Προωθείται στη σωστή θέση μέσα στο ταξινομημένο μέρος
  - Το ταξινομημένο μέρος σταδιακά μεγαλώνει
- Πολυπλοκότητα
  - $O(n^2)$  στη χειρότερη περίπτωση
  - $O(n)$  στην καλύτερη (ήδη ταξινομημένα)

# Ταξινόμηση παρεμβολής (insertion sort)

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες
- Αναζήτηση
- Ταξινόμηση

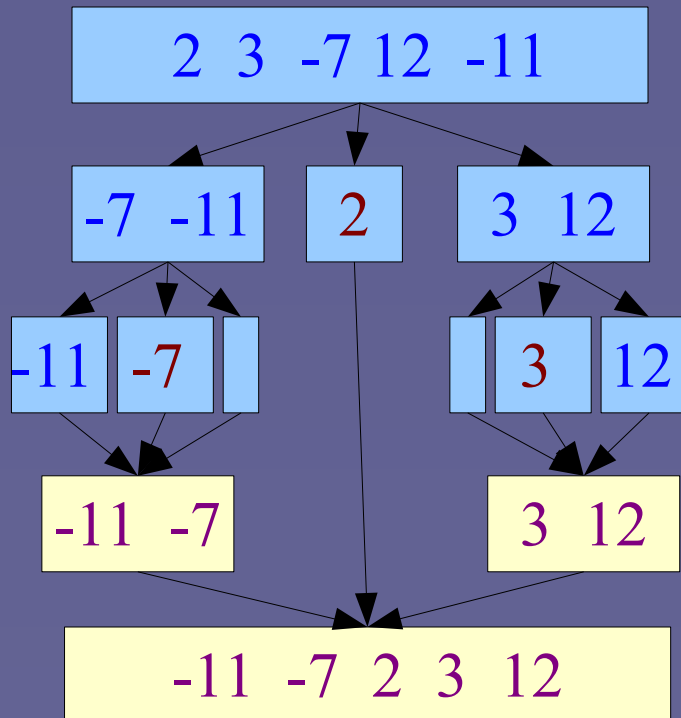


# Quicksort

- Αλγόριθμοι
  - Βασικές αλγοριθμικές δομές
  - Πολυπλοκότητα
  - Δομές δεδομένων
  - Πίνακες
  - Αναζήτηση
  - Ταξινόμηση
- Ένας από τους γνωστότερους αλγορίθμους ταξινόμησης
    - Ακολουθεί την τακτική της διαμέρισης του πίνακα σε δύο μέρη
      - Μικρότερα και μεγαλύτερα στοιχεία από (τυπικά) το πρώτο στοιχείο του πίνακα
    - Και στη συνέχεια ταξινομεί αναδρομικά τους δύο υποπίνακες
    - Στο τέλος συνενώνει τους πίνακες
      - Μικρότερα + στοιχείο διαμέρισης + μεγαλύτερα
  - Πολυπλοκότητα
    - $O(n^2)$  στη χειρότερη περίπτωση (ήδη ταξινομημένα)
    - $O(n \log_2 n)$  καλύτερη περίπτωση και κατά μέσο όρο

# Quicksort

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες
- Αναζήτηση
- Ταξινόμηση



# Διασυνδεδεμένες λίστες

- Αλγόριθμοι
- Βασικές αλγοριθμικές δομές
- Πολυπλοκότητα
- Δομές δεδομένων
- Πίνακες
- Αναζήτηση
- Ταξινόμηση
- Διασυνδεδεμένες λίστες

- Για αποθήκευση ακολουθίας στοιχείων
  - Εναλλακτικά των πινάκων
  - Αποτελείται από κόμβους
    - Στοιχεία (εγγραφές δεδομένων)
  - Κάθε κόμβος διασυνδέεται με τον επόμενο
    - Μονή ή διπλή φορά διασύνδεσης
  - Η εισαγωγή είναι  $O(1)$ 
    - Αλλαγή μόνο της διασύνδεσης των κόμβων
    - Θα πρέπει όμως να ξέρουμε το σημείο εισαγωγής..
  - Αλλά η προσπέλαση τυχαίου στοιχείου γίνεται τώρα  $O(n)$ 
    - Θα πρέπει να διασχίσουμε τη λίστα από κάποια άκρη της μέχρι να φτάσουμε στο στοιχείο που θέλουμε

αρχή

