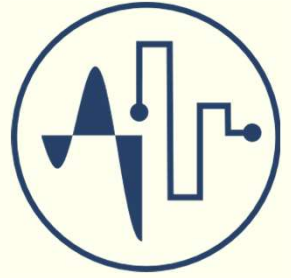


# Αντικειμενοστρεφής Προγραμματισμός



## Ταξινόμηση (Sorting)

### Μέθοδοι Ανταλλαγής

(Bubblesort, Insertion Sort, Selection Sort)



**Παναγιώτης Αδαμίδης**

**adamidis@ihu.gr**



ΔΙΕΘΝΕΣ  
ΠΑΝΕΠΙΣΤΗΜΙΟ  
ΤΗΣ ΕΛΛΑΔΟΣ

# Ταξινόμηση (Sorting)

- Ταξινόμηση είναι η διευθέτηση ενός συνόλου **ομοειδών** στοιχείων σε αύξουσα ή φθίνουσα σειρά, σύμφωνα με κάποιο κριτήριο.
- Υπάρχουν δύο κατηγορίες αλγορίθμων ταξινόμησης: αλγόριθμοι ταξινόμησης δεδομένων που είναι αποθηκευμένα στην κύρια μνήμη (πχ. πίνακες) και αλγόριθμοι ταξινόμησης δεδομένων που είναι αποθηκευμένα στη δευτερεύουσα μνήμη (π.χ. αρχεία). Η κύρια διαφορά τους είναι ότι μόνο στην πρώτη περίπτωση είναι διαθέσιμα **όλα** τα δεδομένα και μπορούν να συγκριθούν ή να γίνει ανταλλαγή της θέσης τους.



# Αλγόριθμοι Ταξινόμησης

## (Sorting algorithms)

- Τρεις γενικές κατηγορίες μεθόδων ταξινόμησης:
  - ♦ Ανταλλαγής (Exchange)
  - ♦ Αναδρομικές (Recursive)
  - ♦ Tree-based
- Συνήθεις μέθοδοι ταξινόμησης με **ανταλλαγή**:
  - ♦ Bubble sort
  - ♦ Ταξινόμηση με επιλογή (Selection sort)
  - ♦ Ταξινόμηση με εισαγωγή (Insertion sort)
- Συνήθεις **αναδρομικές** μέθοδοι ταξινόμησης:
  - ♦ Ταξινόμηση με συγχώνευση (Merge Sort)
  - ♦ Γρήγορη ταξινόμηση (Quick Sort)



# Λειτουργία Αλγόριθμων Ταξινόμησης με Ανταλλαγή

- Θεωρείστε 20 κάρτες (σε τυχαία σειρά) οι οποίες αναγράφουν τις τιμές 1-20. Για να ταξινομηθούν με την μέθοδο:
- **bubble sort**, απλώστε τις κάρτες και συνεχίστε με την ανταλλαγή των καρτών οι οποίες είναι σε λάθος θέση (σύγκριση **μόνο** δύο διαδοχικών καρτών) μέχρι να ταξινομηθούν όλες.
- **εισαγωγής**, κρατείστε όλες τις κάρτες στο χέρι. Τοποθετείστε μία-μία τις κάρτες στο τραπέζι, τοποθετώντας κάθε μία στην σωστή θέση.
- **επιλογής**, απλώστε τις κάρτες και πάρτε στο χέρι την κάρτα με την μικρότερη τιμή. Συνεχίστε με την κάρτα που έχει την μικρότερη τιμή από τις υπόλοιπες κ.ο.κ.



# Bubble Sort: Βασική Ιδέα

- Απλούστερος και λιγότερο αποτελεσματικός αλγόριθμος ταξινόμησης.
- Προσπέλαση των στοιχείων από την αρχή μέχρι το τέλος με συνεχή σύγκριση τιμών **δύο διαδοχικών** στοιχείων.
- Ανταλλαγή των στοιχείων εάν είναι σε λάθος σειρά.



# Bubble Sort: Λειτουργία

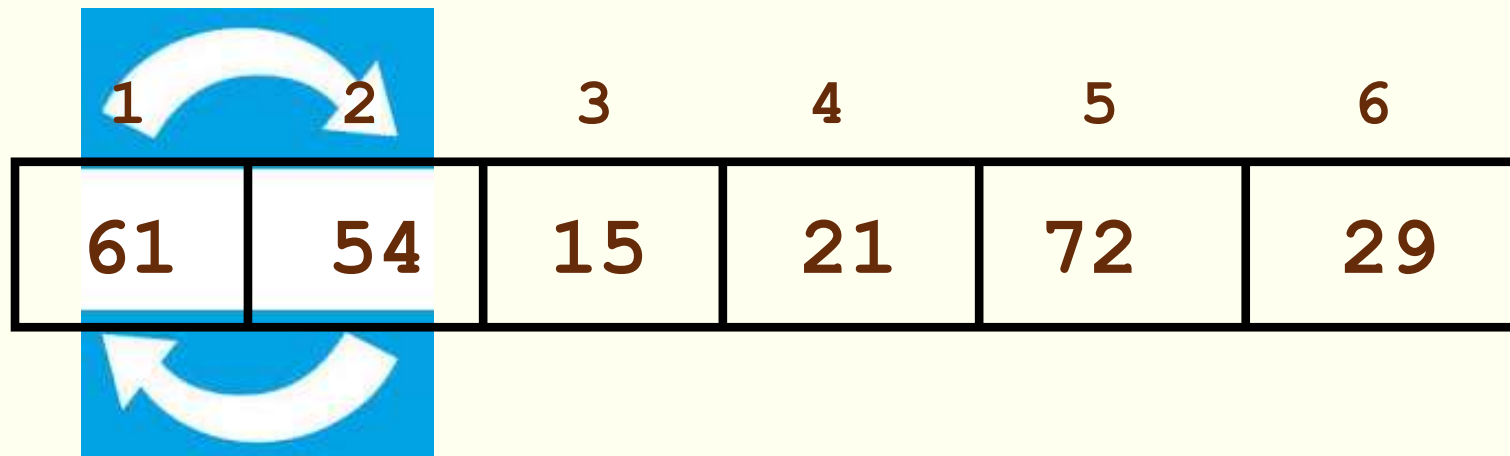
- Ξεκινάμε με τον έλεγχο των δύο πρώτων στοιχείων. Ανταλλαγή των δύο στοιχείων εάν δεν είναι σε σωστή σειρά.
- Συνεχίζουμε με την σύγκριση του 2ου και του 3ου στοιχείου, του 3ου με το 4ο κ.ο.κ. μέχρι τα δύο τελευταία στοιχεία. Έτσι ολοκληρώνεται το πρώτο πέρασμα των στοιχείων.
- ◆ Μετά το 1ο πέρασμα, το μεγαλύτερο στοιχείο έχει μεταφερθεί στο τέλος. Έτσι, στο 2ο πέρασμα δεν είναι απαραίτητος ο έλεγχος του τελευταίου στοιχείου (δηλ. η σύγκριση των 2 τελευταίων στοιχείων).
- ◆ Με το 2ο πέρασμα μεταφέρεται στην προ-τελευταία θέση και το στοιχείο που είναι δεύτερο σε μέγεθος.
- ◆ Για να ταξινομηθούν  $n$  στοιχεία χρειάζονται το πολύ  $n-1$  περάσματα του πίνακα.
- ◆ Ο έλεγχος τέλους της ταξινόμησης μετά από κάθε πέρασμα επιταχύνει τον αλγόριθμο.



# Πρώτο πέρασμα

## Ανάδυση του μεγαλύτερου στοιχείου

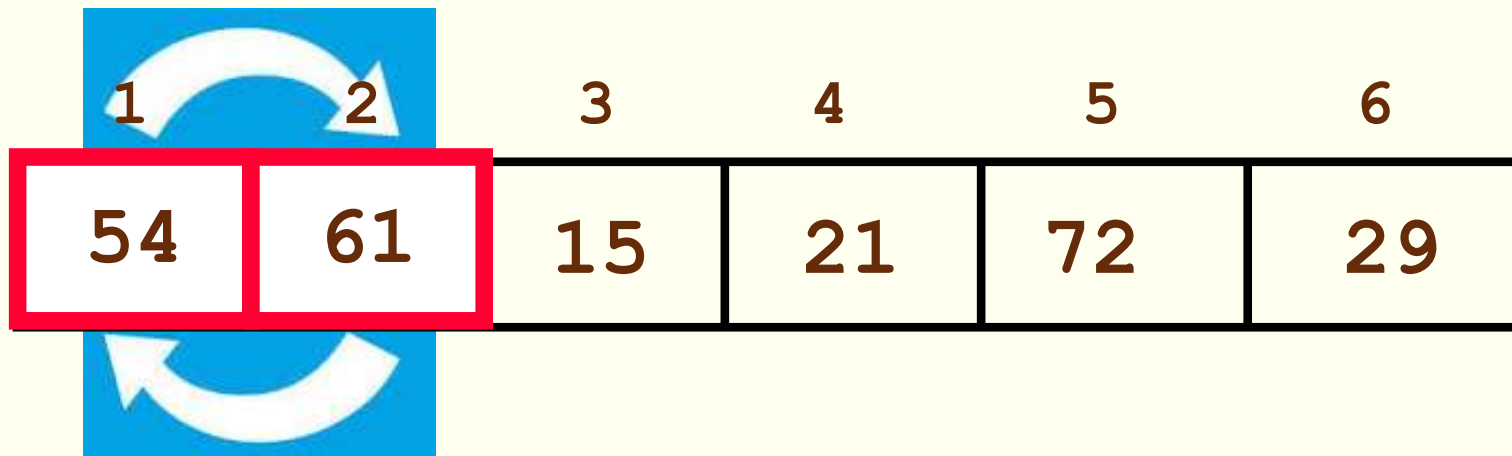
- Διάσχιση (πέρασμα) ενός συνόλου στοιχείων σειριακά από την αρχή μέχρι το τέλος τους
- Ανάδυση σαν φυσαλίδα της μεγαλύτερης τιμής στο τέλος χρησιμοποιώντας συγκρίσεις και ανταλλαγή διαδοχικών στοιχείων



# Πρώτο πέρασμα

## Ανάδυση του μεγαλύτερου στοιχείου

- Διάσχιση (πέρασμα) ενός συνόλου στοιχείων σειριακά από την αρχή μέχρι το τέλος τους
- Ανάδυση σαν φυσαλίδα της μεγαλύτερης τιμής στο τέλος χρησιμοποιώντας συγκρίσεις και ανταλλαγή διαδοχικών στοιχείων

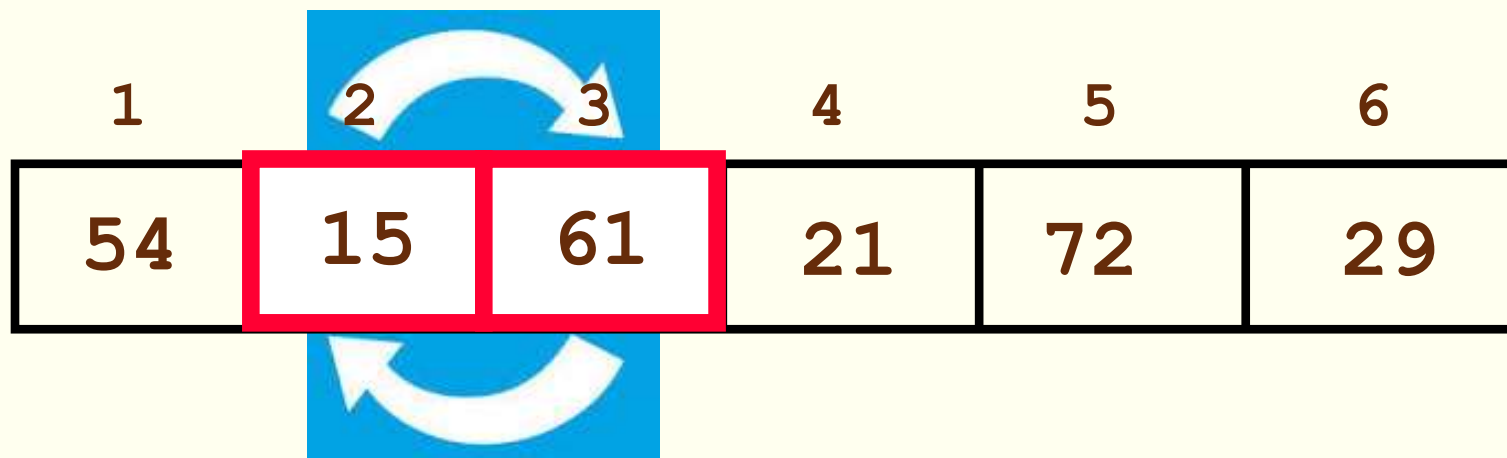




# Πρώτο πέρασμα

## Ανάδυση του μεγαλύτερου στοιχείου

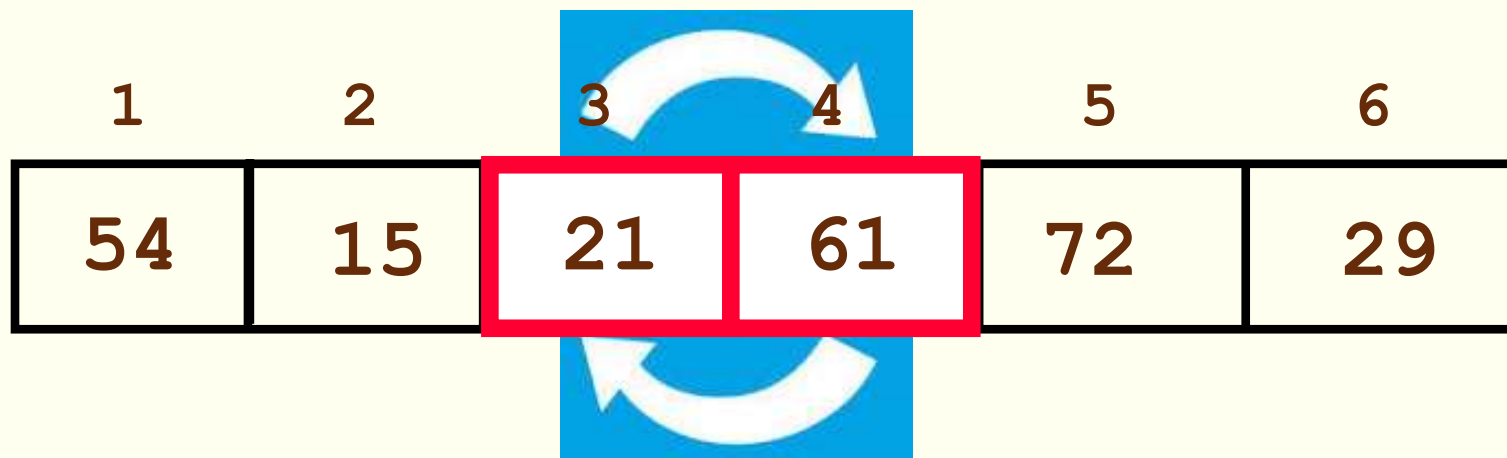
- Διάσχιση (πέρασμα) ενός συνόλου στοιχείων σειριακά από την αρχή μέχρι το τέλος τους
- Ανάδυση σαν φυσαλίδα της μεγαλύτερης τιμής στο τέλος χρησιμοποιώντας συγκρίσεις και ανταλλαγή διαδοχικών στοιχείων



# Πρώτο πέραςμα

## Ανάδυση του μεγαλύτερου στοιχείου

- Διάσχιση (πέραςμα) ενός συνόλου στοιχείων σειριακά από την αρχή μέχρι το τέλος τους
- Ανάδυση σαν φυσαλίδα της μεγαλύτερης τιμής στο τέλος χρησιμοποιώντας συγκρίσεις και ανταλλαγή διαδοχικών στοιχείων



# Πρώτο πέρασμα

## Ανάδυση του μεγαλύτερου στοιχείου

- Διάσχιση (πέρασμα) ενός συνόλου στοιχείων σειριακά από την αρχή μέχρι το τέλος τους
- Ανάδυση σαν φυσαλίδα της μεγαλύτερης τιμής στο τέλος χρησιμοποιώντας συγκρίσεις και ανταλλαγή διαδοχικών στοιχείων

1	2	3	4	5	6
54	15	21	61	72	29

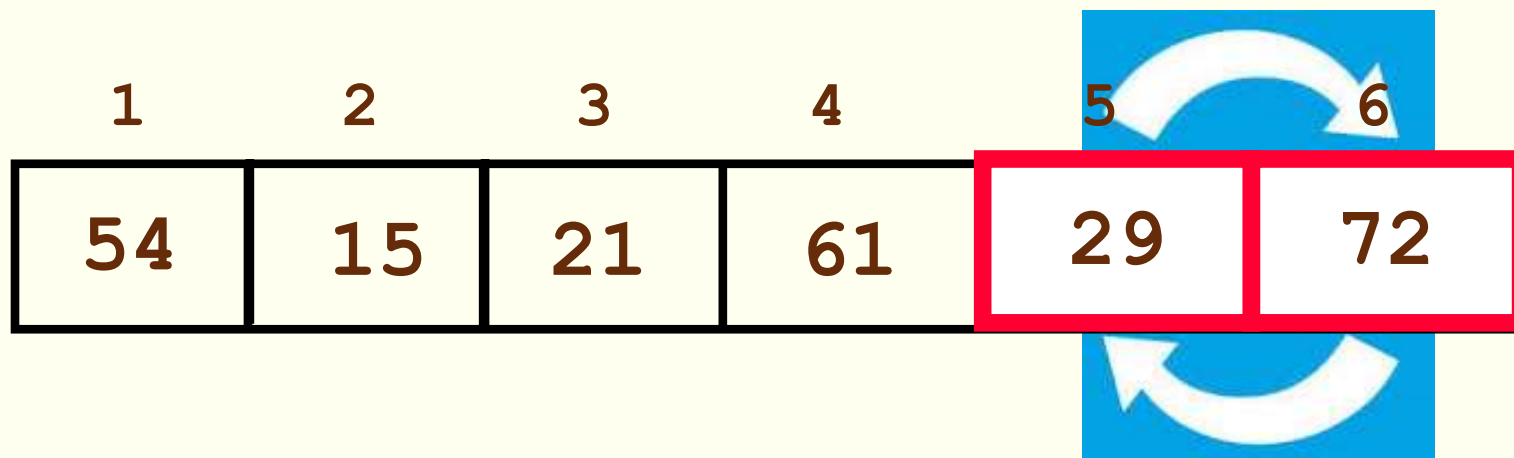
Δεν χρειάζεται  
ανταλλαγή



# Πρώτο πέραςμα

## Ανάδυση του μεγαλύτερου στοιχείου

- Διάσχιση (πέραςμα) ενός συνόλου στοιχείων σειριακά από την αρχή μέχρι το τέλος τους
- Ανάδυση σαν φυσαλίδα της μεγαλύτερης τιμής στο τέλος χρησιμοποιώντας συγκρίσεις και ανταλλαγή διαδοχικών στοιχείων



# Πρώτο πέρασμα

## Ανάδυση του μεγαλύτερου στοιχείου

- Διάσχιση (πέρασμα) ενός συνόλου στοιχείων σειριακά από την αρχή μέχρι το τέλος τους
- Ανάδυση σαν φυσαλίδα της μεγαλύτερης τιμής στο τέλος χρησιμοποιώντας συγκρίσεις και ανταλλαγή διαδοχικών στοιχείων

1	2	3	4	5	6
54	15	21	61	29	72

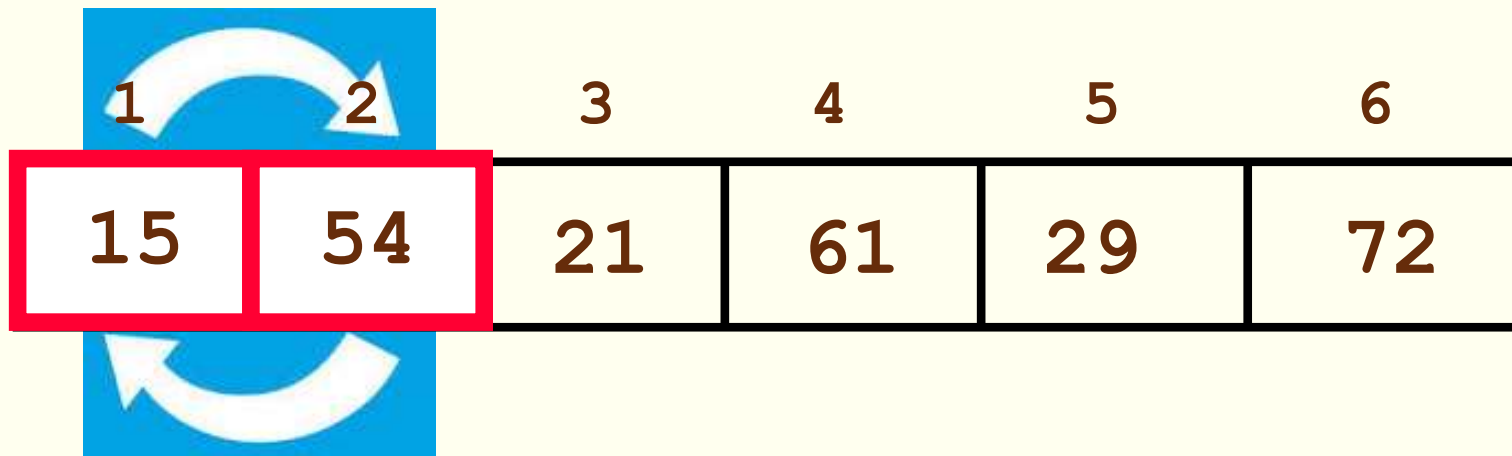
Σωστή θέση  
μεγαλύτερης τιμής

Τέλος 1<sup>ου</sup> περάσματος



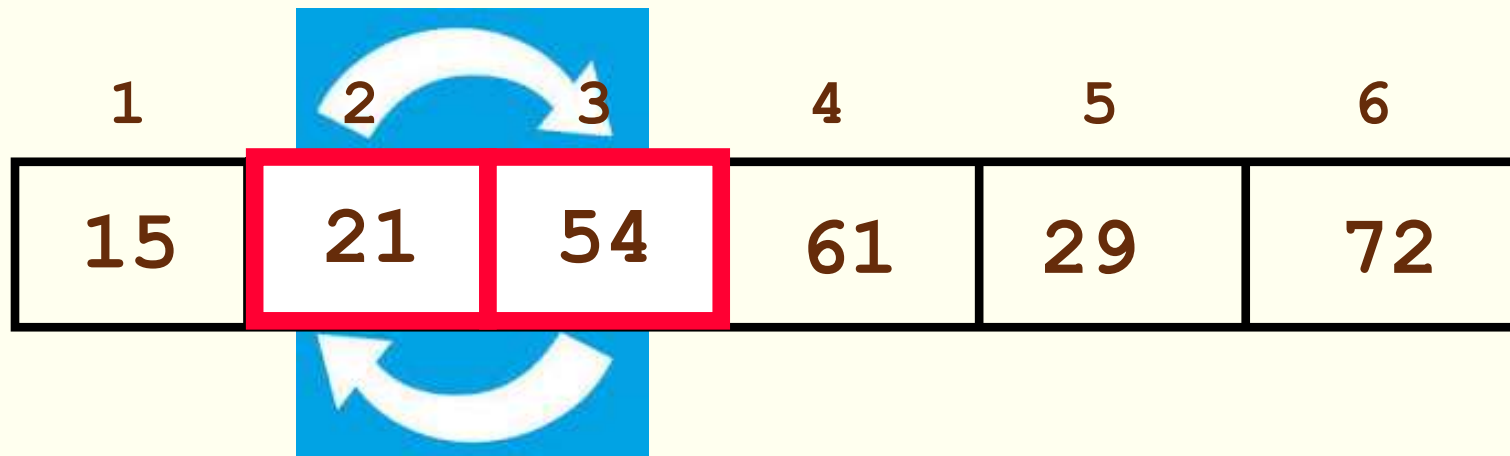
# Δεύτερο πέραςμα

## Ανάδυση του 2<sup>ου</sup> μεγαλύτερου στοιχείου



# 2Δεύτερο πέραςμα

## Ανάδυση του 2<sup>ου</sup> μεγαλύτερου στοιχείου



# Δεύτερο πέραςμα

## Ανάδυση του 2<sup>ου</sup> μεγαλύτερου στοιχείου

1	2	3	4	5	6
15	21	54	61	29	72

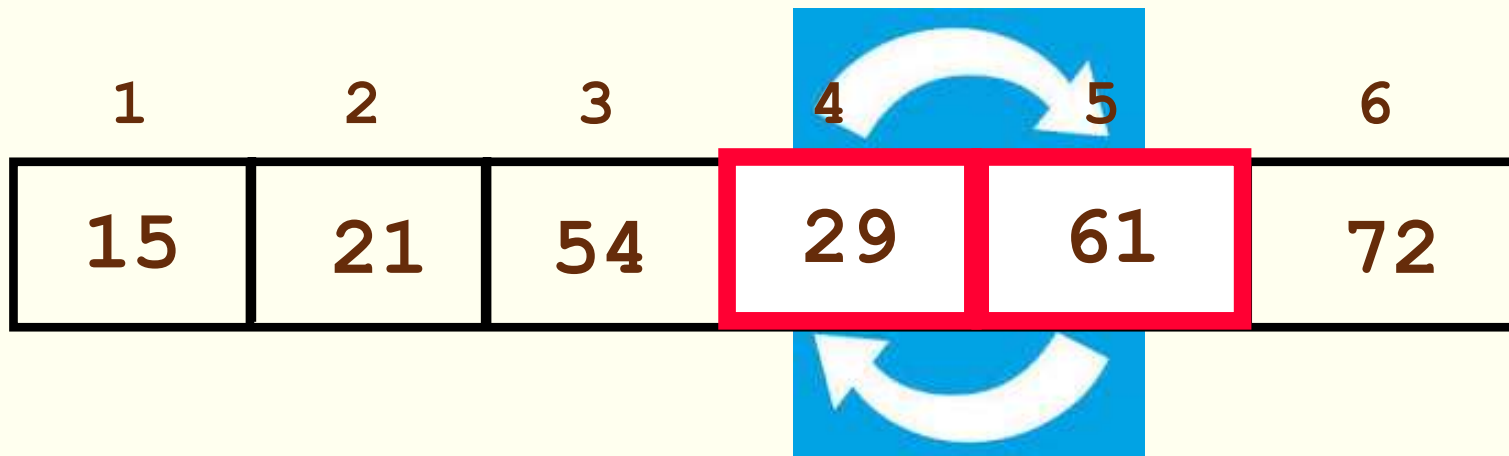
Δεν χρειάζεται  
ανταλλαγή





# Δεύτερο πέραςμα

## Ανάδυση του 2<sup>ου</sup> μεγαλύτερου στοιχείου



# Δεύτερο πέραςμα

## Ανάδυση του 2<sup>ου</sup> μεγαλύτερου στοιχείου

1	2	3	4	5	6
15	21	54	29	61	72

Σωστή θέση 2<sup>ης</sup>  
μεγαλύτερης τιμής

Τέλος 2<sup>ου</sup> περάσματος



# Τρίτο πέραςμα

## Ανάδυση του 3<sup>ου</sup> μεγαλύτερου στοιχείου

1	2	3	4	5	6
15	21	54	29	61	72

Δεν χρειάζεται  
ανταλλαγή



# Τρίτο πέραςμα

## Ανάδυση του 3<sup>ου</sup> μεγαλύτερου στοιχείου

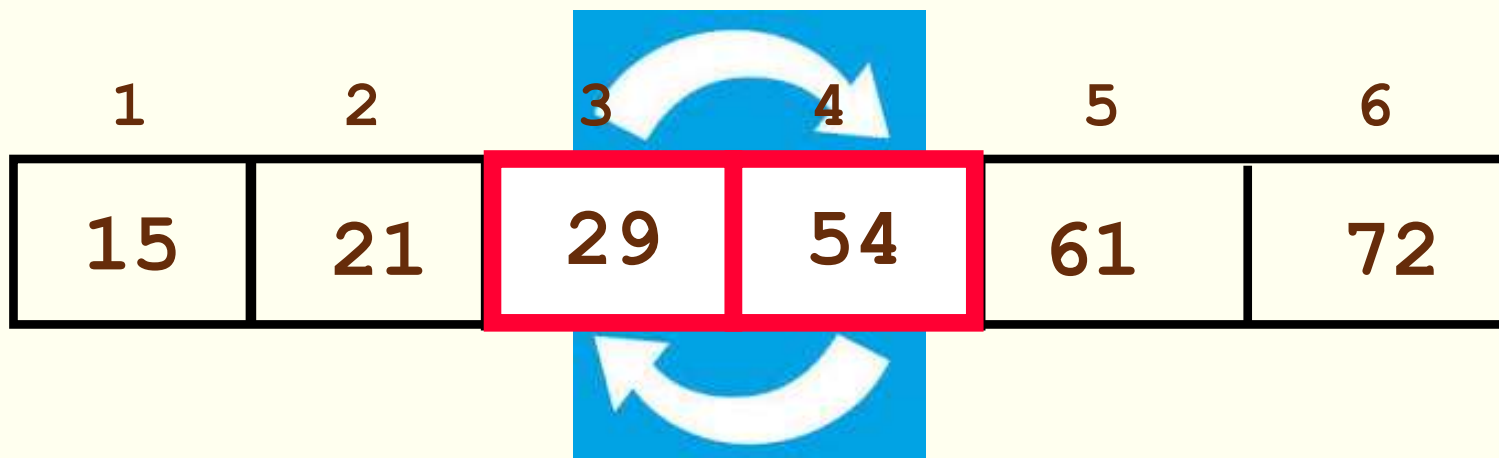
1	2	3	4	5	6
15	21	54	29	61	72

Δεν χρειάζεται  
ανταλλαγή



# Τρίτο πέραςμα

## Ανάδυση του 3<sup>ου</sup> μεγαλύτερου στοιχείου



# Τρίτο πέραςμα

## Ανάδυση του 3<sup>ου</sup> μεγαλύτερου στοιχείου

- Ταξινομημένος πίνακας
- Δεν χρειάζεται άλλο πέραςμα

1	2	3	4	5	6
15	21	29	54	61	72

Σωστή θέση 3<sup>ης</sup>  
μεγαλύτερης  
τιμής

Τέλος 3<sup>ου</sup> περάσματος



# Παράδειγμα: Bubble Sort

- Δίνονται οι αριθμοί:            7            3            8            6            4
- Αρχή 1ου            3            7            8            6            4 (Ανταλλαγή 3, 7)
- 3            7            8            6            4
  - 3            7            6            8            4 (Ανταλλαγή 6, 8)
  - 3            7            6            4            8 (Ανταλλαγή 4, 8)
  - Τέλος 1ου περάσματος. Το 8 είναι στη σωστή θέση.
- Αρχή 2ου            3            7            6            4            8
- 3            6            7            4            8 (Ανταλλαγή 6, 7)
  - 3            6            4            7            8 (Ανταλλαγή 4, 7)
  - Δεν είναι απαραίτητος ο έλεγχος του τελευταίου ζεύγους αριθμών (7, 8)
  - Τέλος 2ου. Το 7 είναι στη σωστή θέση.
- Αρχή 3ου            3            6            4            7            8
- 3            4            6            7            8 (Ανταλλαγή 4, 6)
  - Δεν είναι απαραίτητος ο έλεγχος των 2 τελευταίων ζευγών (6, 7) και (7, 8)
  - Τέλος 3ου. Και το 6 είναι στη σωστή θέση
- Δεν χρειάζεται 4ο πέρασμα γιατί οι αριθμοί είναι ήδη ταξινομημένοι.



# Μέθοδος: Bubble Sort

➤ Ταξινόμηση πίνακα:

```
public static void bubble (int[] num) {  
    int temp;  
  
    ...  
  
    if (numbers[0] > numbers[1]) {  
        temp=numbers[0];  
        numbers[0]= numbers[1];  
        numbers[1]= temp;  
    }  
  
}  
  
} // method sort
```





# Μέθοδος: Bubble Sort

- Ταξινόμηση πίνακα ακεραίων:

```
public static void bubble (int[] num) {  
    int temp, j;  
  
    ...  
    // Ένα πέρασμα του πίνακα  
    for (j=0; j < numbers.length-1; j++)  
        if (numbers[j] > numbers[j+1]){  
            temp=numbers[j];  
            numbers[j]= numbers[j+1];  
            numbers[j+1]= temp;  
  
        }  
  
    }  
}  
// method sort
```



# Μέθοδος: Bubble Sort

➤ Ταξινόμηση πίνακα ακεραίων:

```
public static void bubble (int[] num) {  
    int i, j, temp;  
    // N-1 περάσματα του πίνακα  
    for (i=1; i < numbers.length; i++) {  
        ...  
        for (j=0; j < numbers.length-i; j++)  
            if (numbers[j] > numbers[j+1]){  
                temp=numbers[j];  
                numbers[j]= numbers[j+1];  
                numbers[j+1]= temp;  
                ...  
            }  
        }  
    }  
} // method sort
```



# Μέθοδος: Bubble Sort

➤ Ταξινόμηση πίνακα ακεραίων:

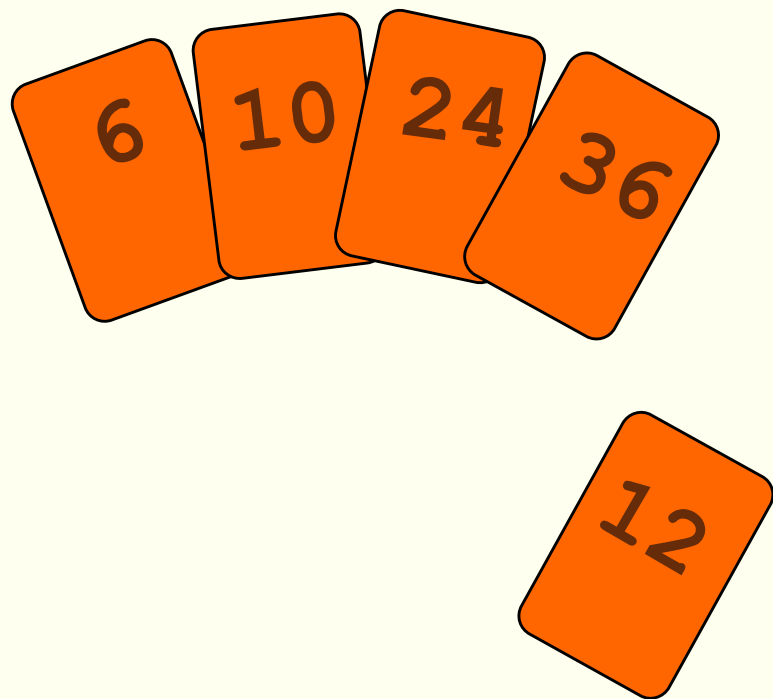
```
public static void bubble (int[] num) {  
    int i, j, temp;  
    boolean flag;  
    for (i=1; i < numbers.length; i++) {  
        flag=true;  
        for (j=0; j < numbers.length-i; j++)  
            if (numbers[j] > numbers[j+1]){  
                temp=numbers[j];  
                numbers[j]= numbers[j+1];  
                numbers[j+1]= temp;  
                flag=false;  
            }  
        if (flag) return;  
    }  
} // method sort
```



# Ερωτήσεις;



# Insertion Sort

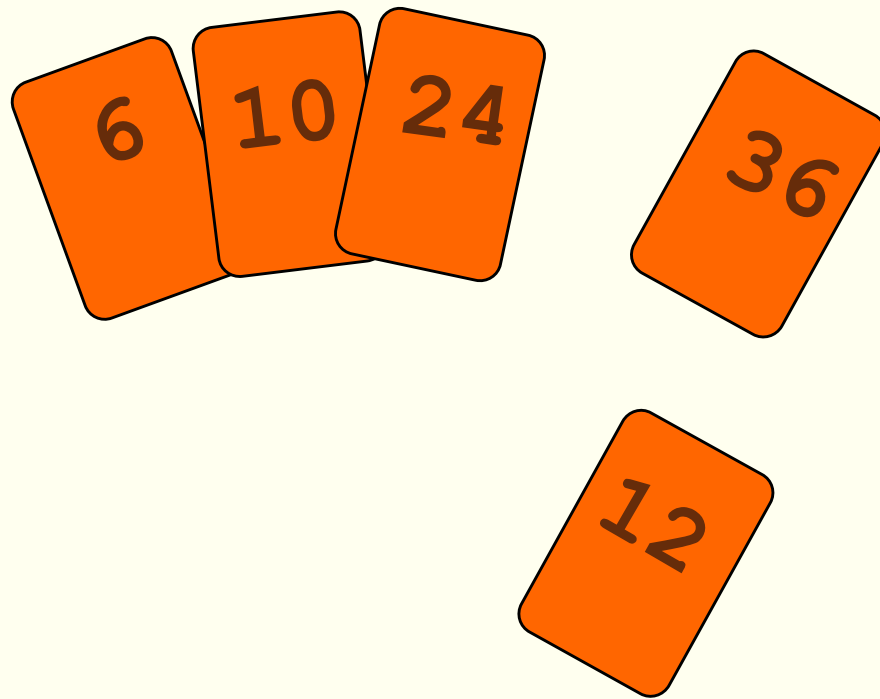


Για να εισάγουμε το 12, πρέπει να κάνουμε χώρο για αυτό μετακινώντας πρώτα το 36 και μετά το 24.



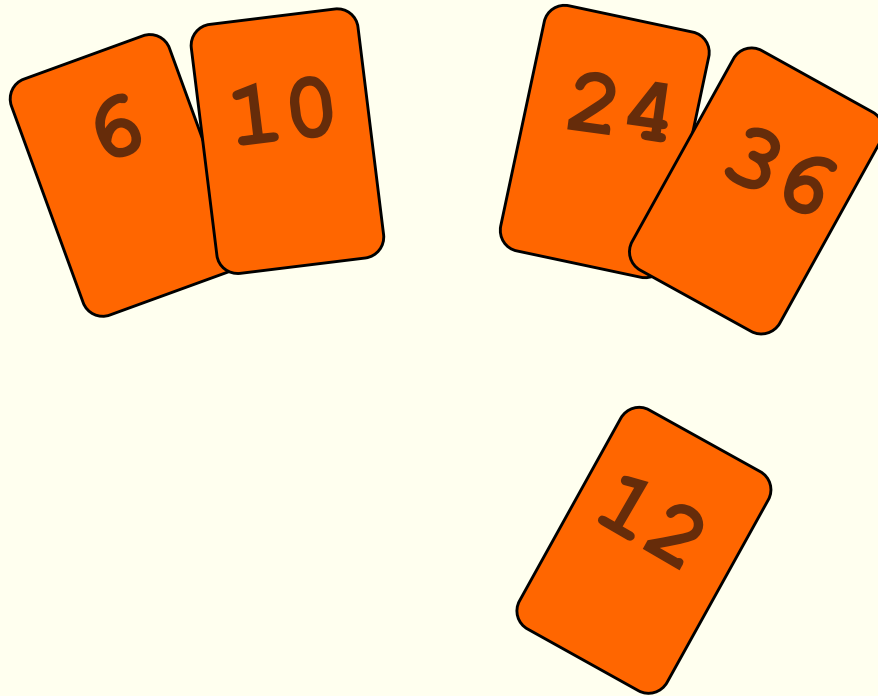
# Insertion Sort

---



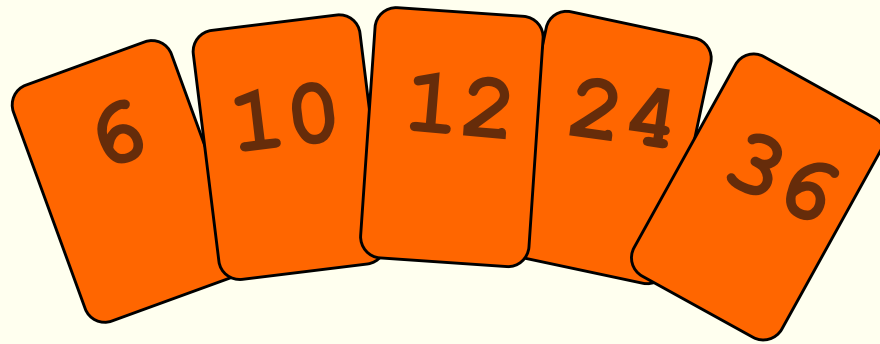
# Insertion Sort

---



# Insertion Sort

---





# Insertion Sort

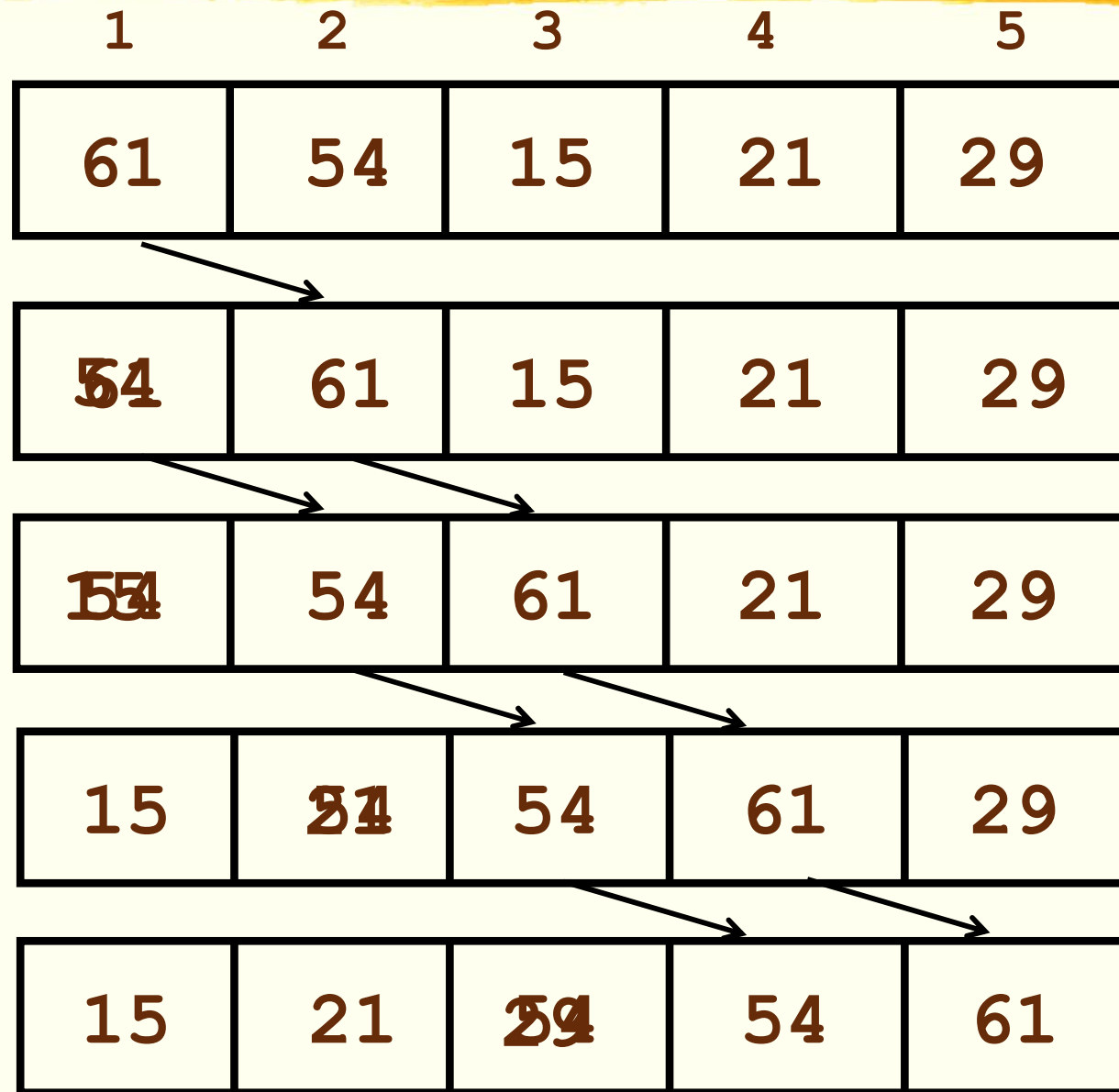
- Βασική ιδέα: ένα-ένα τα στοιχεία εισάγονται στη σωστή θέση στον προηγούμενο ταξινομημένο υπο-πίνακα.
- Λειτουργία (για αύξουσα ταξινόμηση):
  - ♦ Ένα στοιχείο μόνο του, είναι ταξινομημένο.
  - ♦ Αρχή με το δεύτερο στοιχείο. Αν είναι μικρότερο από το πρώτο στοιχείο αλλάζουν θέση, αλλιώς δεν υπάρχει μεταβολή. Τα δύο πρώτα στοιχεία είναι ταξινομημένα.
  - ♦ Το τρίτο στοιχείο εισάγεται στη σωστή θέση: Εάν τα προηγούμενα στοιχεία είναι μεγαλύτερα, ολισθαίνουν προς τα δεξιά. Τα τρία πρώτα στοιχεία είναι ταξινομημένα.
  - ♦ Το τέταρτο στοιχείο εισάγεται στη σωστή θέση: Εάν τα προηγούμενα στοιχεία είναι μεγαλύτερα, ολισθαίνουν προς τα δεξιά. Τα τέσσερα πρώτα στοιχεία είναι ταξινομημένα.
  - ♦ Η διαδικασία συνεχίζεται με την εισαγωγή όλων των στοιχείων.
  - ♦ Για να ταξινομηθούν  $n$  στοιχεία χρειάζονται το πολύ  $n-1$  περάσματα του πίνακα.



# Insertion Sort

## Ταξινόμηση πίνακα 5 στοιχείων

29



# Παράδειγμα: Insertion Sort

➤ Δίνονται οι αριθμοί:            7            3            8            6            4

➤ 1ο: Το **3** ταξινομείται στη σωστή θέση (πρώτη) στον προηγούμενο ταξινομημένο υπο-πίνακα. Υπόλοιπα στοιχεία ολισθαίνουν προς τα δεξιά.

•    3            7            8            6            4

➤ 2ο: Το **8** ταξινομείται στη σωστή θέση (τρίτη-όπου βρίσκεται) στον προηγούμενο ταξινομημένο υπο-πίνακα.

•    3            7            8            6            4

➤ 3ο: Το **6** ταξινομείται στη σωστή θέση (δεύτερη) στον προηγούμενο ταξινομημένο υπο-πίνακα.

•    3            6            7            8            4

➤ 4ο: Το **4** ταξινομείται στη σωστή θέση (δεύτερη) στον προηγούμενο ταξινομημένο υπο-πίνακα.

•    3            4            6            7            8



# Μέθοδος: Insertion Sort

➤ Ταξινόμηση πίνακα ακεραίων:

```
public static void sort(int[] num) {  
    for (int i=1; i < num.length; i++) {  
        int current = num[i];  
        int position=i;  
        // shift larger values to the right  
        while (position>0 && num[position-1]>current) {  
            num[position] = num[position-1];  
            position--;  
        }  
        num[position] = current;  
    }  
} // method sort
```



# Ερωτήσεις;



# Selection Sort

- Βασική ιδέα: εύρεση μικρότερου (ή μεγαλύτερου ανάλογα με το είδος της ταξινόμησης) στοιχείου και ανταλλαγή με το στοιχείο που βρίσκεται στην κατάλληλη θέση.
- Λειτουργία:
  - ♦ Αρχή με την εύρεση του μικρότερου όλων των στοιχείων και ανταλλαγή με το στοιχείο που βρίσκεται στην πρώτη θέση.
  - ♦ Εύρεση του μικρότερου όλων των στοιχείων, εκτός από το στοιχείο της πρώτης θέσης και ανταλλαγή με το στοιχείο που βρίσκεται στη δεύτερη θέση.
  - ♦ Εύρεση του μικρότερου όλων των στοιχείων, εκτός από τα στοιχεία των δύο πρώτων θέσεων και ανταλλαγή με το στοιχείο που βρίσκεται στην τρίτη θέση.
  - ♦ Η διαδικασία συνεχίζεται μέχρι να μείνει ένα μόνο στοιχείο.
  - ♦ Για να ταξινομηθούν  $n$  στοιχεία χρειάζονται το πολύ  $n-1$  περάσματα του πίνακα.



# Παράδειγμα: Selection Sort

- Δίνονται οι αριθμοί:            7            3            8            6            4
- 1ο: 3 μικρότερο όλων. Ανταλλαγή με 7 που βρίσκεται στην πρώτη θέση
  - 3            7            8            6            4 (Ανταλλαγή 3, 7)
- 2ο: 4 μικρότερο υπολοίπων. Ανταλλαγή με 7 που βρίσκεται στην δεύτερη θέση
  - 3            4            8            6            7 (Ανταλλαγή 4, 7)
- 3ο: 6 μικρότερο υπολοίπων. Ανταλλαγή με 8 που βρίσκεται στην τρίτη θέση
  - 3            4            6            8            7 (Ανταλλαγή 6, 8)
- 4ο: 7 μικρότερο υπολοίπων. Ανταλλαγή με 8 που βρίσκεται στην τέταρτη θέση
  - 3            4            6            7            8 (Ανταλλαγή 7, 8)



# Μέθοδος: Selection Sort

➤ Ταξινόμηση πίνακα ακεραίων:

```
public static void sort (int[] num) {  
    int min, temp;  
    for (int i = 0; i < num.length-1; i++) {  
        // find position of minimum value  
        min=i;  
        for (int k=i+1; k<num.length; k++)  
            if (num[k] < num[min])  
                min=k;  
        //swap the values in positions "min" and "i"  
        temp = num[min];  
        num[min] = num[i];  
        num[i] = temp;  
    }  
} // method sort
```





# Χρόνοι εκτέλεσης (ms)

(Intel Core i7-4700MQ, 2.4GHz και 8 GB RAM)

Μέσος όρος από 1000 επαναλήψεις εκτέλεσης του κάθε αλγόριθμου σε τυχαία/διαφορετικά σύνολα δεδομένων για διαφορετικό πλήθος στοιχείων.

Πλήθος στοιχείων	Bubble	Selection	Insertion
1.000	1.187	0.469	0.188
2.000	4.687	1.687	0.719
5.000	32.17	9.79	4.05
10.000	151.65	38.18	16.38
20.000	620.32	98.08	64.17
100.000	3003.56	661.22	284.25



# Ασκήσεις

- Χρησιμοποιείτε το παράδειγμα που χρησιμοποιήθηκε στην μέθοδο bubble sort, για να εξηγήσετε την λειτουργία των μεθόδων ταξινόμησης με επιλογή (Sorting by selection) και με εισαγωγή (Sorting by insertion)
- Γράψτε ένα πρόγραμμα το οποίο καταχωρεί τον παρακάτω πίνακα με τις τιμές ενοικίασης κάποιων αυτοκινήτων

Τύπος αυτοκινήτου	Ημερήσια χρέωση	Χρέωση ανά χιλ.
BMW 316	65	0.35
Toyota Yaris	45	0.32
Fiat Punto	40	0.28
Hyundai	35	0.30
Mercedes	80	0.40

Ο πίνακας είναι ταξινομημένος σύμφωνα με τον τύπο του αυτοκινήτου (String).

Το πρόγραμμά σας θα ταξινομεί τον πίνακα σύμφωνα με την ημερήσια χρέωση και θα τον εμφανίζει πριν και μετά την ταξινόμηση.



# Ερωτήσεις;

