Πίνακες

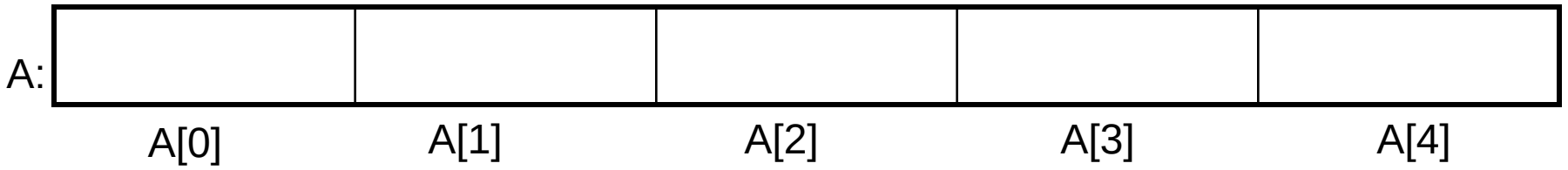
- Χρησιμοποιούνται για αποθήκευση συνόλου δεδομένων του ίδιου τύπου.
- Γραμμική Διάταξη
- Δήλωση

Τύπος Δεδομένων ΌνομαΠίνακα[length]

Αριθμητική τιμή που εκφράζει το μέγεθος του πίνακα

Πχ: int A[5];

Πίνακες - Προσπέλαση



- Κάθε θέση του πίνακα μπορεί να προσπελαστεί με χρήση Index. Για πίνακα μεγέθους N το index ανήκει στο διάστημα [0, N-1]
- Για να αναφερθούμε στο i στοιχείο του πίνακα αρκεί να γράψουμε A[i-1]

Παράδειγμα – Αρχικοποίηση στοιχείων

```
main()
{
    int k[20],i;
//αρχικοποίηση όλων των στοιχείων του πίνακα με την τιμή 5
    for (i=0; i<20;i++)
          k[i] = 5;

}
```

Δήλωση και Αρχικοποίηση ταυτόχρονα

- int A[20] = {3,1,2,56,78};
- Τα 5 πρώτα στοιχεία του Α θα έχουν τις τιμές που δηλώνονται εντός των αγκυλών. Ο πίνακας είναι 20 θέσεων

- Αν απουσιάζει το μέγεθος του πίνακα, ο πίνακας έχει τόσα στοιχεία όσα δηλώνονται στις αγκύλες
- Πχ int A = {3,12,6} είναι πίνακας 3 θέσεων

Πίνακες

- Κάθε στοιχείο του πίνακα μπορεί να χρησιμοποιηθεί σαν μεταβλητή του τύπου δεδομένων του πίνακα.
- Πχ int A[20];
- **scanf(“%d”,&A[0]);**

αποθηκεύει στο A[0] τον ακέραιο που θα διαβαστεί από το stream εισόδου

- printf(“Value of A[0] is:%d”,A[0]);

Εκτυπώνει την τιμή του A[0]

Παράδειγμα

- Φτιάξτε ένα πρόγραμμα που να αποθηκεύει σε έναν πίνακα 10 χαρακτήρες από το stream εισόδου. Τέλος εκτυπώστε τον πίνακα.

Πρόγραμμα

```
main(){
char A[10];
int I;
        printf(“Input 10 characters\n”);
        for (i=0; i<10; i++)
                A[i] = getchar();
        printf(“Characters are:\n”);
        for (i=0; i<10; i++)
                putchar(A[i]);


}
```

Δείκτες και Πίνακες

- Το όνομα ενός πίνακα είναι δείκτης στο πρώτο του στοιχείο

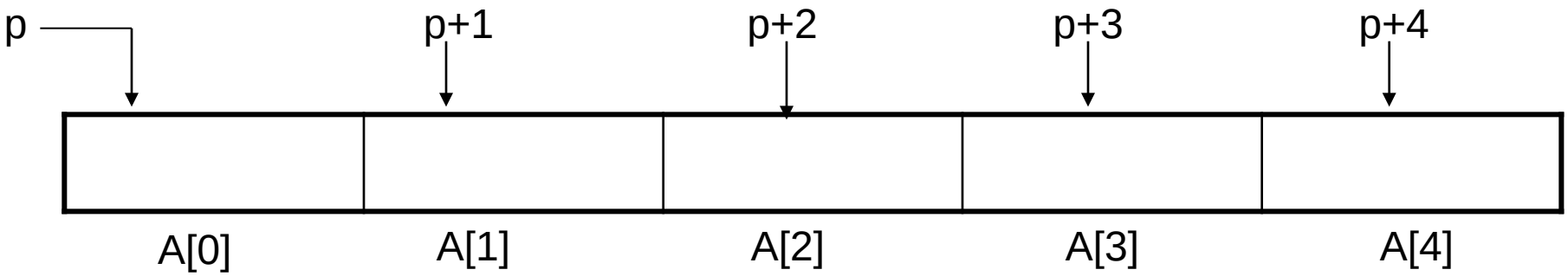
int A[20], *p;

p = A;

Είναι ισοδύναμη με την εντολή

p = &A[0];

Αριθμητική δεικτών (έχει νόημα μόνο σε πίνακες)



- **«Προσθέτω 1 σε δείκτη» ισοδυναμεί με την έκφραση «Δείκτης στην επόμενη θέση»**
- **A + i είναι δείκτης στο στοιχείο A[i]**

Για προσπέλαση του A μπορούμε να γράψουμε

```
int i;
        for (i = 0; i<N; i++)
            printf("A[%d]:%d\n",i, *(A+i) );
```

Πίνακες ως παράμετροι συναρτήσεων

- Δήλωση

double athroisma(**int vathmos[ ]**, int megethos);

Ισοδύναμη με τη δήλωση

double athroisma(**int *vathmos**, int megethos);

- Κλήση:

athroisma(vathmos,300);

Πίνακες

- Το πέρασμα των πινάκων σε συναρτήσεις γίνεται με αναφορά (by reference), περνάμε τη διεύθυνση του πρώτου στοιχείου του πίνακα.
- Αλλαγές σε στοιχεία του πίνακα που περνιέται σαν όρισμα στη συνάρτηση, είναι μόνιμα
- **Δεν υποστηρίζεται επιστροφή πίνακα από συνάρτηση**

Άσκηση

- Φτιάξτε ένα πρόγραμμα που να διαβάζει 10 ακεραίους από το stream εισόδου και να τους αποθηκεύει σε έναν πίνακα. Στη συνέχεια μια συνάρτηση που να επιστρέφει το πλήθος εμφάνισης συγκεκριμένου ακεραίου στον πίνακα.
- Υπόδειξη: Η επικεφαλίδα της συνάρτησης int IntCount(int A[],int k); Θέλουμε να επιστρέφεται το πλήθος εμφάνισης του k στον πίνακα A[].

Πρόγραμμα

```
#include <stdio.h>
#include <stdlib.h>

int IntCount(int A[ ], int k){
int i,count =0;
    for (i =0; i<10; i++)
        if (A[i] == k)
            count++;
    return count;
}
```

Πρόγραμμα

```
main(){
int Input[20], a, i;
    printf("Input 10 int\n");
    for (i =0; i<10; i++)
        scanf("%d",&Input[i]);
    printf("Input integer to be searched\n");
    scanf("%d",&a);
    printf("%d sequences\n",IntCount(Input,a));
    system("pause");
    return 0;
}
```

Άσκηση

- Τροποποιήστε το προηγούμενο πρόγραμμα ώστε η προσπέλαση των στοιχειών του πίνακα να γίνεται με χρήση δείκτη.

Πρόγραμμα

```
#include <stdio.h>
#include <stdlib.h>

int IntCount(int A[ ], int k){
int i,count =0;
    for (i =0; i<10; i++)
        if ( *(A + i) == k)
            count++;
    return count;
}
```

Πρόγραμμα

```
main(){
int Input[20], a, i;


      printf("Input 10 int\n");
      for (i =0; i<10; i++)
          scanf("%d", Input+i );
      printf("Input integer to be searched\n");
      scanf("%d",&a);
      printf("%d sequences \n",IntCount(Input,a),a);
      system("pause");
      return 0;
}
```

Πίνακες μεγαλύτερης διάστασης

- Η C υποστηρίζει πίνακες μέχρι 12 διαστάσεων
- Δήλωση πινάκων 2 διαστάσεων

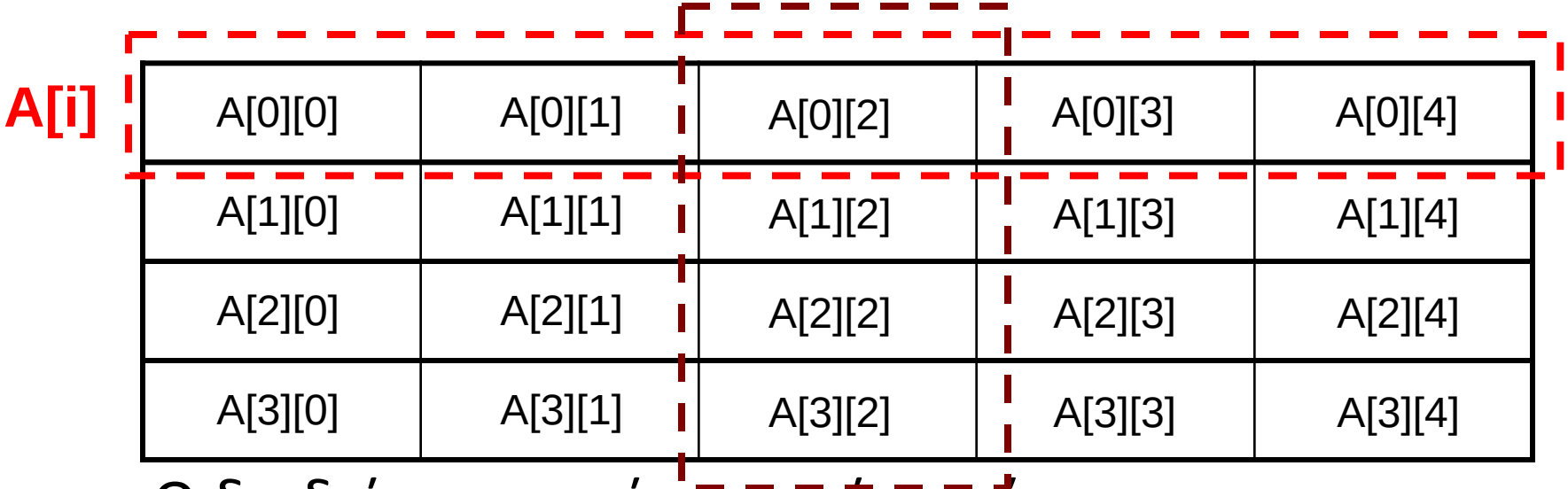
Τύπος Δεδομένων ΌνομαΠίνακα**[**Γραμμές**][**Στήλες**]**

Πχ με τη δήλωση

int A[4][5] ορίζεται ο δισδιάστατος πίνακας ακεραίων Α, 4x5

Πίνακες μεγαλύτερης διάστασης

- Δείκτες Γραμμών και στηλών ξεκινούν από το 0.



- Ο δισδιάστατος πίνακας είναι πίνακας μονοδιάστατων πινάκων

Πίνακες μεγαλύτερης διάστασης Προσπέλαση στοιχείων

- Σάρωση των τιμών του δισδιάστατου πίνακα Α 4 γραμμών και 5 στηλών:

```
int i,j;
   for (i=0;i<4;i++) //σάρωση κάθε γραμμής
        for (j=0;j<5;j++) //σάρωση κάθε στήλης της γραμμής i
             A[i][j] = 0; //ανάθεση τιμής στη γραμμή i και στήλη j
```

Πολυδιάστατοι πίνακες ως παράμετροι συναρτήσεων

- Δήλωση

double athroisma(**int vathmos[ ][sthles]**, int grammes);

- Κλήση:

int vathmos[grammes][sthles]

athroisma(vathmos,grammes);

Ταξινόμηση Πινάκων – Μέθοδος Bubble Sort

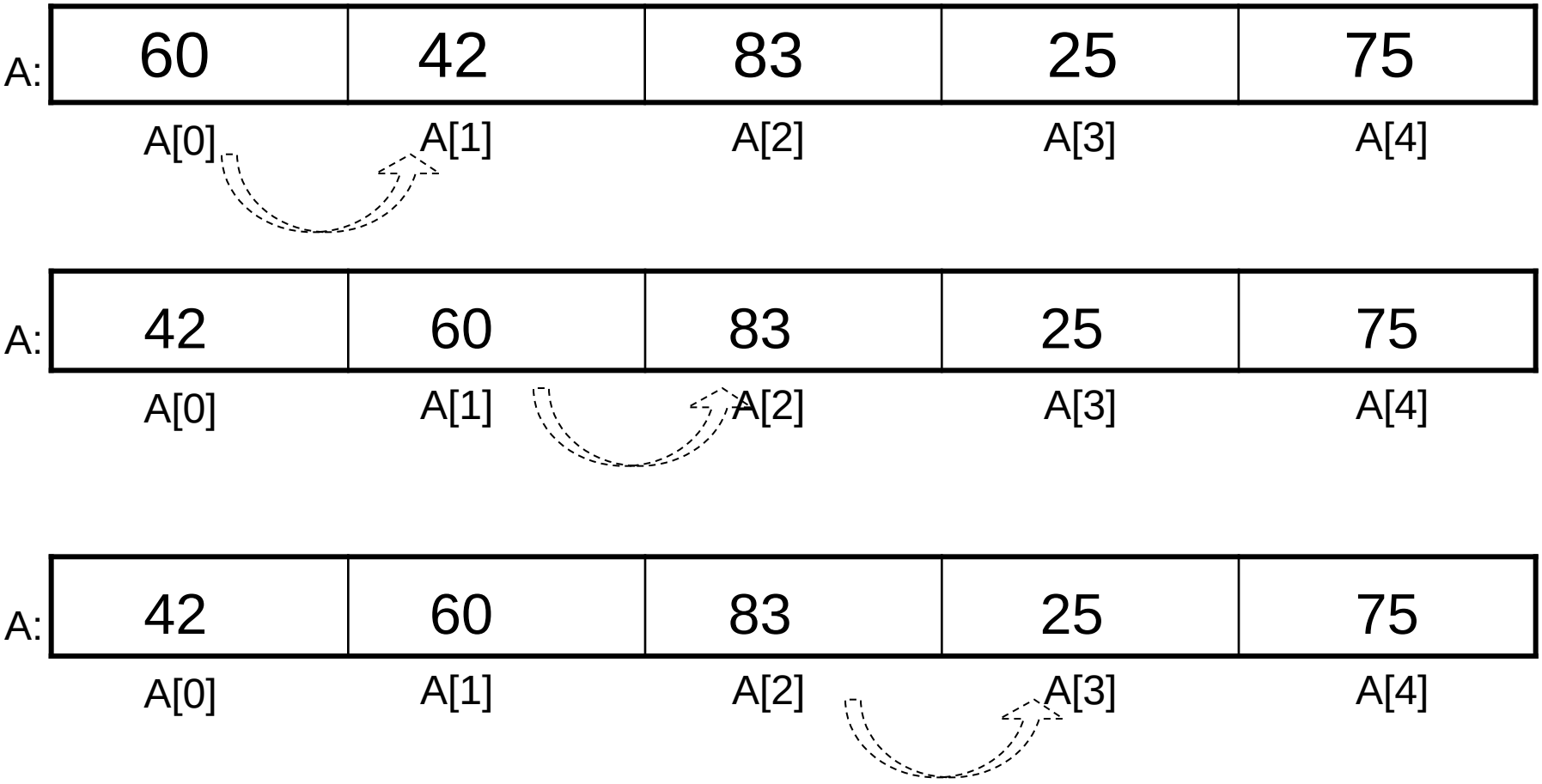
Για την ταξινόμηση πίνακα Ν στοιχείων

- Σύγκριση των στοιχείων ανά 2
- Ανταλλαγή τιμών αν το 1ο στοιχείο είναι μεγαλύτερο απ’ το 2ο

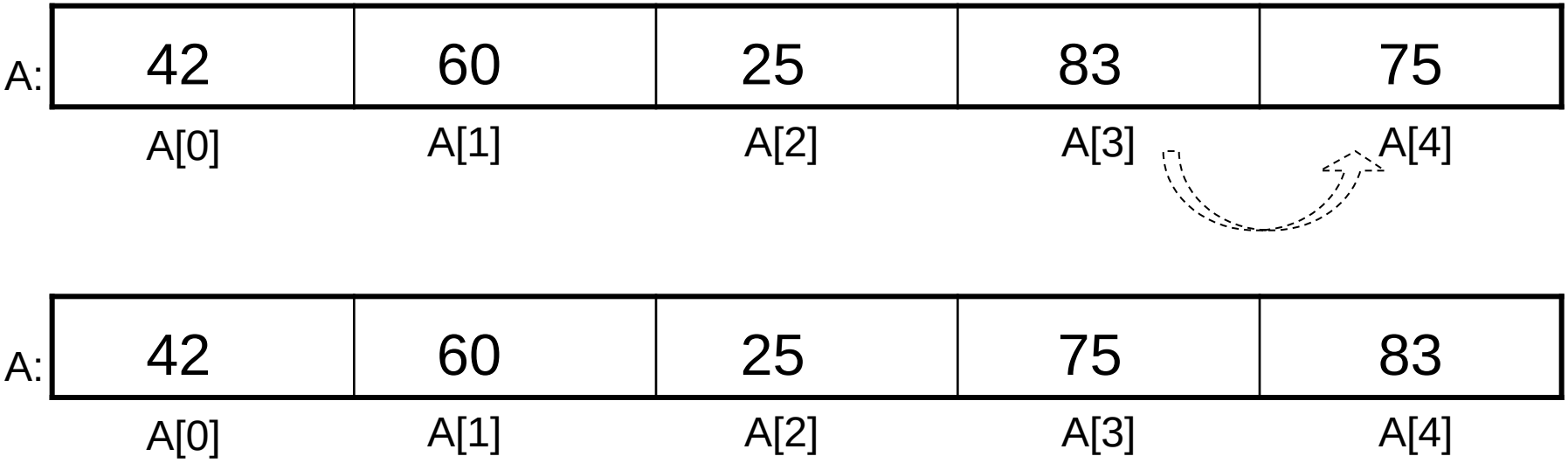
Το μεγαλύτερο στοιχείο του πίνακα βρίσκεται στην τελευταία θέση

Τα βήματα επαναλαμβάνονται Ν-1 φορές

Ταξινόμηση Πινάκων – Μέθοδος Bubble Sort

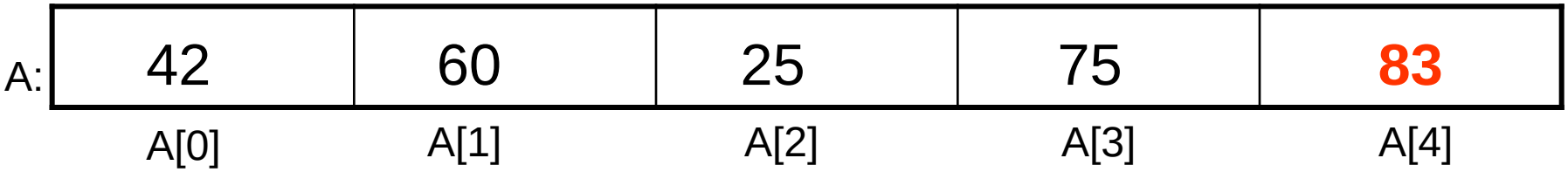


Ταξινόμηση Πινάκων – Μέθοδος Bubble Sort

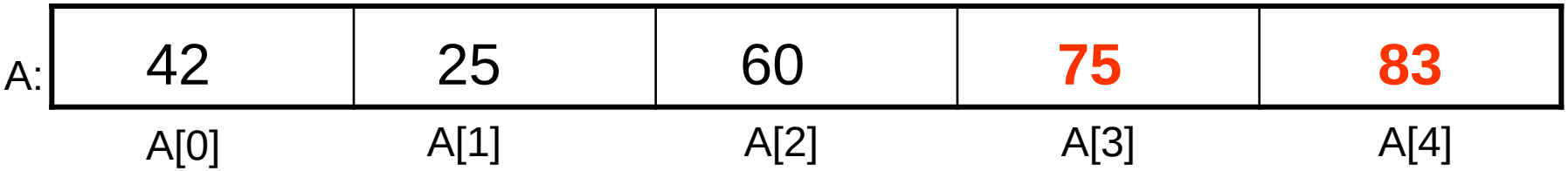


Ταξινόμηση Πινάκων – Μέθοδος Bubble Sort

- Τέλος 1ης φάσης



- Τέλος 2ης φάσης



Ταξινόμηση Πινάκων – Μέθοδος Bubble Sort

- Τέλος 3ης φάσης

- Τέλος 4ης φάσης

Ταξινόμηση Πινάκων – Μέθοδος Bubble Sort

```
#include <stdio.h>
#include <stdlib.h>

main()
{ int m[5] = {23,20,12,52,10}, i, j, temp;
    for (i = 4; i>0; i--) // Επανάληψη βημάτων 4 φορές
    {
                    for (j = 0; j<i; j++) // i συγκρίσεις σε κάθε φάση
                   if (m[j]>m[j+1]) //Ανταλλαγή στοιχείων m[j], m[j+1]
                    {
                                temp = m[j+1];
                                m[j+1] = m[j];
                                m[j] = temp;
                    }
    }
```

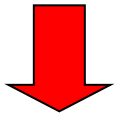
Ταξινόμηση Πινάκων – Μέθοδος Bubble Sort

```
//Εκτύπωση στοιχείων ταξινομημένου πίνακα
    for (i = 0; i<5; i++)
        printf("%d\n",m[i]);

    system("pause");
    return 0;
}
```

Μέθοδος Επιλογής – Selection Sort

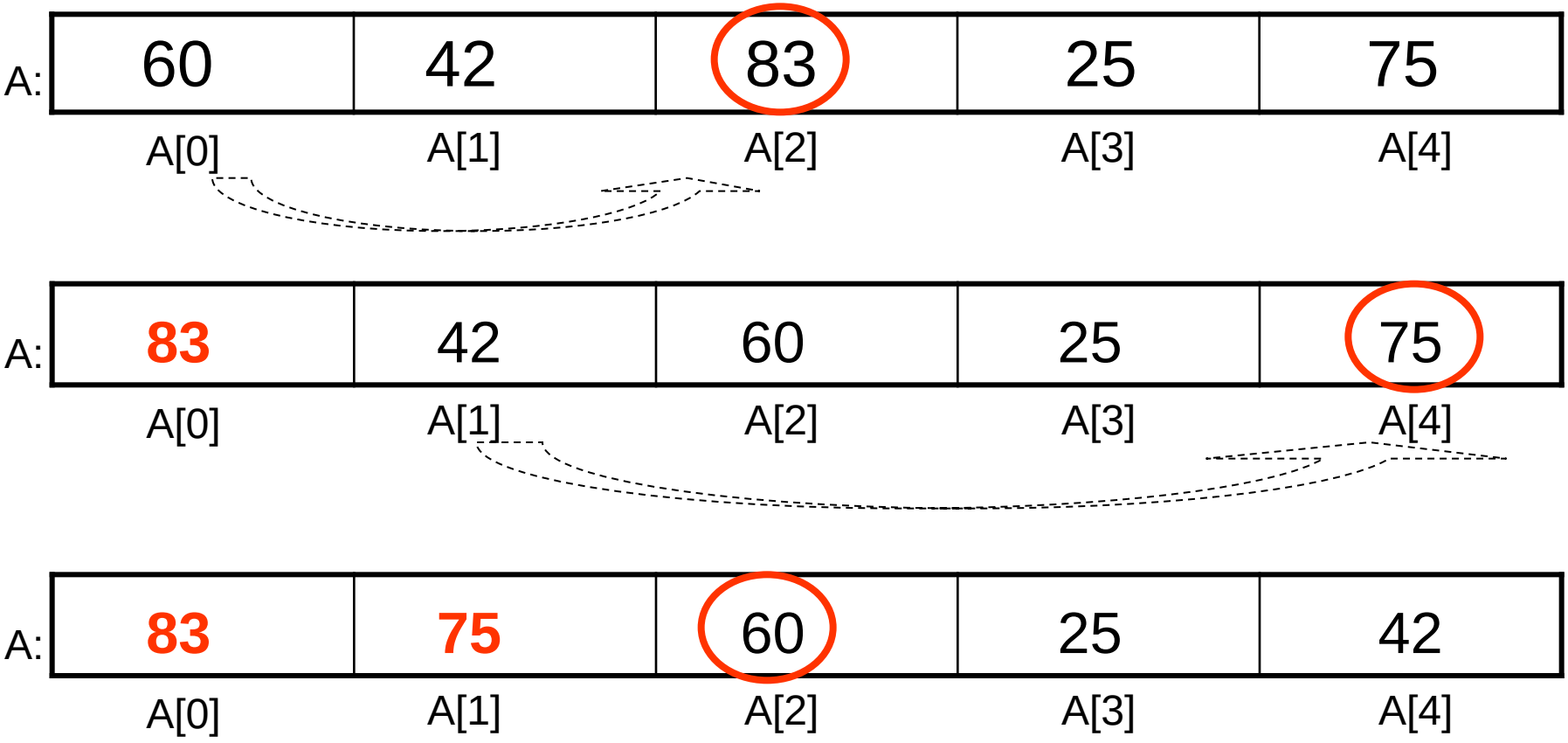
- Εντοπισμός εκάστοτε μεγίστου
- Ανταλλαγή μεγίστου με το στοιχείο στην πρώτη μη ταξινομημένη θέση του πίνακα



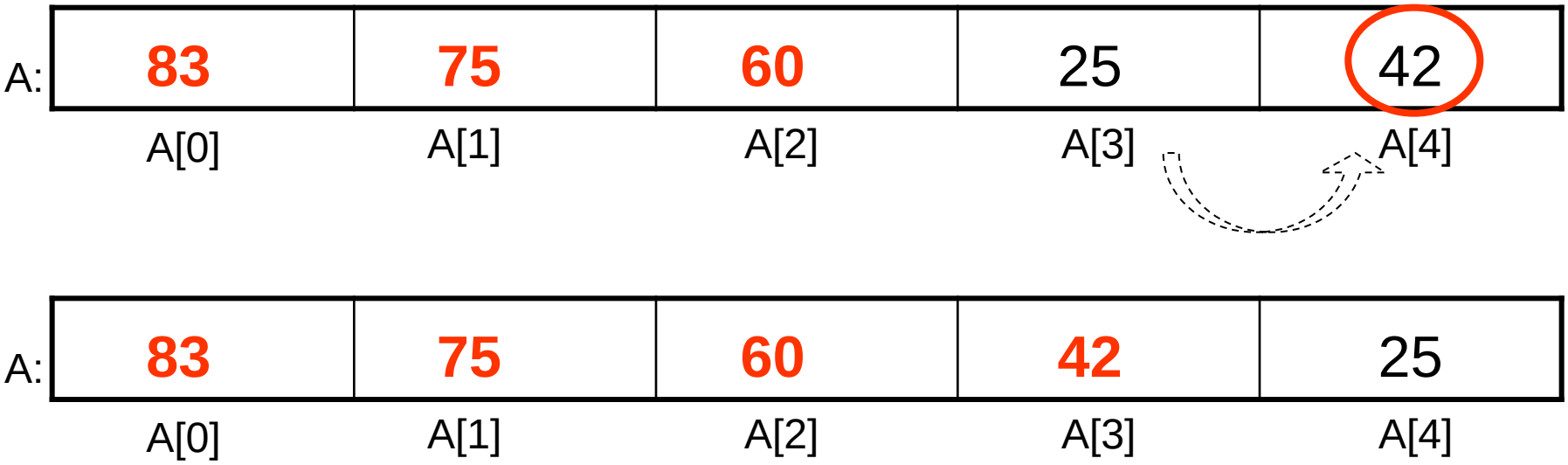
Το μεγαλύτερο στοιχείο του πίνακα βρίσκεται στην πρώτη θέση

Τα βήματα επαναλαμβάνονται Ν-1 φορές

Μέθοδος Επιλογής – Selection Sort



Μέθοδος Επιλογής – Selection Sort



Μέθοδος Επιλογής – Selection Sort

```
#include <stdio.h>
#include <stdlib.h>

main()
{ int m[5] = {23,20,12,52,122}, i, j, meg, temp;

     for (i = 0; i<4; i++){ // Επανάληψη βημάτων 4 φορές
        meg = i;

        for (j = i; j<5; j++) //Εύρεση μεγίστου
           if (m[meg]<m[j])
              meg = j;
//Ανταλλαγή τιμών μεγίστου με το 1ο μη ταξινομημένο
        temp = m[i];
        m[i] = m[meg];
        m[meg] = temp;
     }
```

Δείκτης μεγίστου

Μέθοδος Επιλογής – Selection Sort

```
//Εκτύπωση στοιχείων ταξινομημένου πίνακα
    for (i = 0; i<5; i++)
        printf("%d\n",m[i]);

    system("pause");
    return 0;
}
```