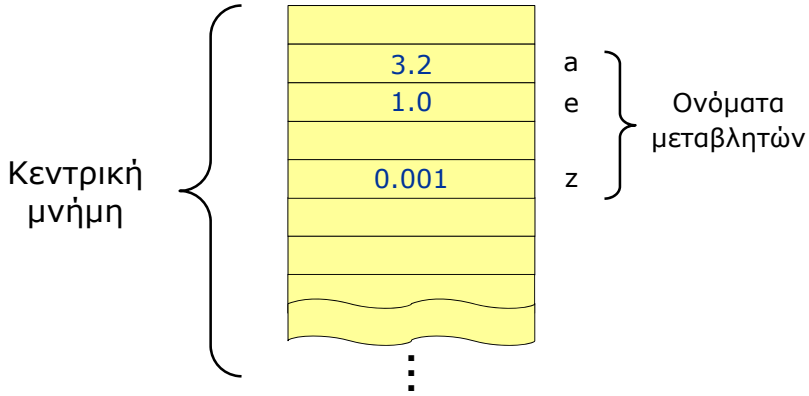
ΥΠΟΛΟΓΙΣΤΕΣ ΙΙ

Μονοδιάστατοι πίνακες

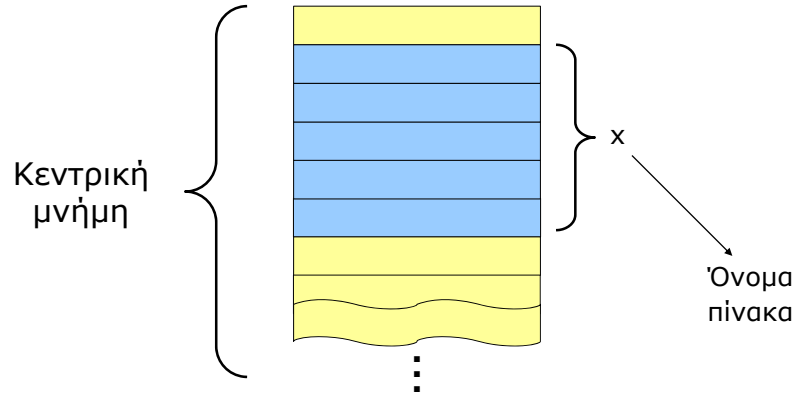
Τι είναι οι πίνακες;

Απλές μεταβλητές:



Τι είναι οι πίνακες;

Πίνακες:



Τι είναι οι πίνακες;

Μια ομάδα διαδοχικών θέσεων αποθήκευσης στην κεντρική μνήμη, στην οποία αναφερόμαστε με ένα κοινό όνομα.

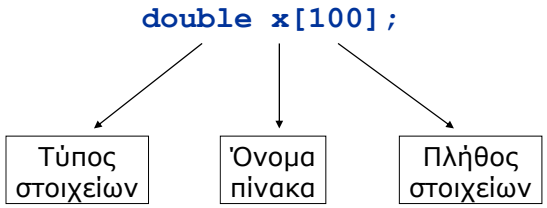
Κάθε στοιχείο του πίνακα ξεχωρίζει από τα υπόλοιπα με τη χρήση ενός δείκτη. Έτσι για ένα πίνακα που ονομάζεται x:

Το πρώτο στοιχείο είναι το x[0]
Το δεύτερο στοιχείο είναι το x[1]

Σημείωση:
Όλα τα στοιχεία του πίνακα είναι του ιδίου τύπου.

Δήλωση

Πριν χρησιμοποιήσουμε ένα πίνακα πρέπει να τον δηλώσουμε. Για να δηλώσουμε ένα πίνακα που ονομάζεται x και αποτελείται από 100 στοιχεία διπλής ακρίβειας γράφουμε:



Ορισμένες παρατηρήσεις

- Το 100 είναι το πλήθος των στοιχείων του πίνακα και όχι ο δείκτης του τελευταίου στοιχείου.
- Τα στοιχεία του πίνακα είναι τα:
  x[0], x[1], x[2], … x[99].
- Το κάθε στοιχείο του πίνακα μπορεί να χρησιμοποιηθεί όπως και μια μεταβλητή του αντίστοιχου τύπου. Πχ.

```
x[0] = 4;
a = 3*x[0]+2*x[1];
cin >> x[4];
```

- Από τα 100 στοιχεία μπορούμε να χρησιμοποιήσουμε στο πρόγραμμα λιγότερα.

Δήλωση

Το πλήθος των στοιχείων μπορεί να δηλωθεί με τη χρήση συμβολικών σταθερών:

```
#define NMAX 100
double x[NMAX];
```

Η γραμμή **#define** απευθύνεται στον προεπεξεργαστή και ορίζει μια συμβολική σταθερά - όχι μεταβλητή -

Προσοχή: όχι ερωτηματικό στο τέλος του **#define**

Ανάθεση τιμών στα στοιχεία πίνακα

Ανάθεση τιμών κατά τη δήλωση:

```
int a[10]={4,7,1,-3,0};
```

Η ανωτέρω δήλωση αναθέτει αρχικές τιμές στα 5 πρώτα από τα 10 στοιχεία του πίνακα a

Ανάθεση τιμών στα στοιχεία πίνακα

Απευθείας ανάθεση:

```
x[0] = 1.2;
x[1] = 2.5;
x[2] = z+2*x[0];

for ( k=0; k<n; ++k )
    x[k] = k*k/2;
```


Ανάθεση τιμών στα στοιχεία πίνακα

Εισαγωγή από το πληκτρολόγιο:

Εισάγουμε ένα-προς-ένα τα στοιχεία:

```
#define NMAX 100
double x[NMAX];

cout << "Πόσα στοιχεία ? ";
cin >> n;
if ( n>NMAX || n<1 ) {
    cout << "Λανθασμένος αριθμός \n";
    return 1;
}
for ( k=0; k<n; ++k ) {
    cout << "Εισάγετε το στοιχείο " << k << endl;
    cin >> x[k];
}
```


Εμφάνιση των στοιχείων πίνακα στην οθόνη

Εμφανίζουμε ένα-προς-ένα τα στοιχεία:

```
#define NMAX 100
double x[NMAX];

for ( k=0; k<n; ++k )
    cout << x[k] << endl;
```


Παράδειγμα #1

Βρείτε το μέσο όρο μιας σειράς αριθμών που εισάγονται από το πληκτρολόγιο.

Παράδειγμα #1

```
#include <iostream>
using namespace std;

#define NMAX 100
int main ( )
{
    double x[NMAX];
    int n, k;
    double s, avg;

    cout << "Πόσα στοιχεία ? ";
    cin >> n;
    if ( n > NMAX || n < 1 ) {
        cout << "Λάθος \n";
        return 1;
    }
    for ( k=0; k<n; ++k ) {
        cout << "Εισάγετε το στοιχείο " << k << endl;
        cin >> x[k];
    }
```

Συνεχίζεται...

Παράδειγμα #1

...Συνέχεια

```
    s = 0;
    for ( k=0; k<n; ++k )
        s += x[k];
    avg = s/n;

    cout << "Ο μέσος όρος είναι: " << avg << endl;

}
```


Παράδειγμα #2

Συμπληρώστε το προηγούμενο παράδειγμα έτσι ώστε να εμφανίζει στην οθόνη πόσο απέχει κάθε αριθμός που εισάγαμε από το μέσο όρο.

Παράδειγμα #2

```
    for ( k=0; k<n; ++k )
        cout << k << "    " << x[k] << "    "
             << abs(avg-x[k]) << endl;
```

Τυπικό αποτέλεσμα στην οθόνη για 4 στοιχεία:

```
0    1.7    0.8805
1    0.85    1.7305
2    3.6    1.0195
3    4.172    1.5915
```

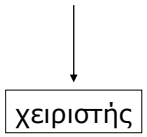
Παρατηρείστε ότι οι τρεις στήλες των αποτελεσμάτων δεν είναι στοιχισμένες.

Μορφοποίηση της εξόδου

Για αν μορφοποιήσουμε κατά βούληση την εμφάνιση των αποτελεσμάτων μας στην οθόνη χρησιμοποιούμε τους χειριστές στην εντολή cout.

Πχ. για να εμφανίσουμε έναν ακέραιο ώστε να καταλαμβάνει 5 θέσεις στην οθόνη:

```
cout << setw(5) << k << endl;
```



Οι βασικότεροι χειριστές

- **`setw(n)`**
  Η επόμενη έξοδος θα καταλαμβάνει n θέσεις στην οθόνη.
- **`setprecision(n)`**
  Όλες οι επόμενες έξοδοι θα γίνουν με n δεκαδικά ψηφία.
- **`fixed`**
  Όλες οι επόμενες έξοδοι θα γίνουν με την κλασσική αναπαράσταση.
- **`scientific`**
  Όλες οι επόμενες έξοδοι θα γίνουν με την επιστημονική αναπαράσταση.

Σημείωση: Για να χρησιμοποιήσουμε τους χειριστές μορφοποίησης πρέπει να δηλώσουμε:

```
#include <iomanip>
```


Παράδειγμα #2 (με μορφοποίηση)

```
cout << fixed << setprecision(7);
for ( k=0; k<n; ++k )
    cout << setw(3)  << k << "    "
         << setw(10) << x[k] << "   "
         << setw(10) << abs(avg-x[k]) << endl;
```

Τυπικό αποτέλεσμα στην οθόνη για 4 στοιχεία:

```
  0     1.7000000     0.8805000
  1     0.8500000     1.7305000
  2     3.6000000     1.0195000
  3     4.1720000     1.5915000
```


Παράδειγμα #3

Βρείτε το μικρότερο στοιχείο ενός πίνακα με n στοιχεία.

Παράδειγμα #3

```
min = x[0];
for ( k=1; k<n; ++k )
    if ( x[k] < min )
        min = x[k];
```


Παράδειγμα #4

Βρείτε το μικρότερο στοιχείο ενός πίνακα με n στοιχεία και τη θέση στην οποία βρίσκεται αυτό.

Παράδειγμα #4

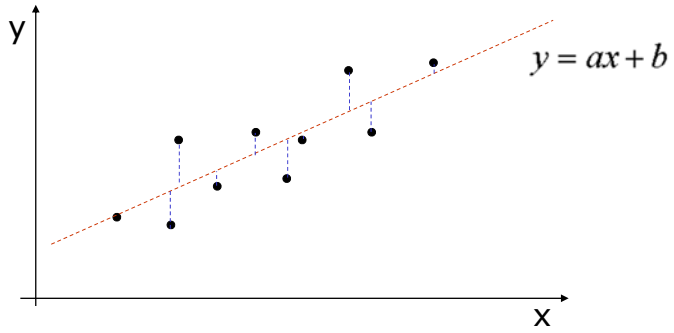
```
min = x[0];
kmin = 0;
for ( k=1; k<n; ++k )
    if ( x[k] < min ) {
        min = x[k];
        kmin = k;
    }
```


Παράδειγμα #5

Προσαρμογή ελαχίστων τετραγώνων

Δεδομένων Ν σημείων (x_i,y_i) στο επίπεδο προσδιορίστε τη βέλτιστη ευθεία που ελαχιστοποιεί τις αποστάσεις από τα σημεία.

Παράδειγμα #5



Παράδειγμα #5

Η βέλτιστη ευθεία έχει εξίσωση: $y = ax + b$

Οι συντελεστές *a*, *b* δίνονται από:

$$a = \frac{Ns_{xy} - s_x s_y}{Ns_{xx} - s_x s_x} \qquad b = \frac{s_{xx} s_y - s_x s_{xy}}{Ns_{xx} - s_x s_x}$$

$$s_x = \sum_{i=1}^{N} x_i \qquad s_y = \sum_{i=1}^{N} y_i \qquad s_{xx} = \sum_{i=1}^{N} x_i^2 \qquad s_{xy} = \sum_{i=1}^{N} x_i y_i$$

Παράδειγμα #5

```
#include <iostream>
using namespace std;
#define NMAX 100

int main ( )
{
    double x[NMAX], y[NMAX];
    int n, k;
    double sx, sy, sxx, sxy;
    double a, b;

    cout << "Πόσα σημεία έχετε ";
    cin >> n;
    if ( n < 2 || n > NMAX ) {
        cout << "Λάθος \n";
        return 1;
    }
    for ( k=0; k<n; ++k ) {
        cout << "Δώστε το σημείο " << k << endl;
        cin >> x[k] >> y[k];
    }
```

Συνεχίζεται...

Παράδειγμα #5

...Συνέχεια

```
    sx = sy = sxx = sxy = 0;
    for ( k=0; k<n; ++k ) {
        sx += x[k];
        sy += y[k];
        sxx += x[k]*x[k];
        sxy += x[k]*y[k];
    }

    a = (n*sxy-sx*sy)/(n*sxx-sx*sx);
    b = (sxx*sy-sx*sxy)/(n*sxx-sx*sx);

    cout << "Οι συντελεστές είναι: \n";
    cout << a << endl;
    cout << b << endl;
}
```

Παράδειγμα #6

Σύστημα φορτισμένων σωματιδίων

Θεωρήστε ένα σύστημα που αποτελείται από N φορτισμένα σωματίδια, το καθένα με φορτίο q_i, και μάζα m_i. Η θέση του κάθε σωματιδίου προσδιορίζεται από τις συντεταγμένες x_i, y_i, z_i.

Βρείτε:

α) το ολικό φορτίο του συστήματος

β) το κέντρο μάζας του συστήματος

γ) την ολική ηλεκτροστατική ενέργεια

Παράδειγμα #6

Το ολικό φορτίο είναι: $q_{tot} = \sum q_i$

Το κέντρο μάζας είναι:

$$x_m = \frac{\sum x_i m_i}{\sum m_i} \qquad y_m = \frac{\sum y_i m_i}{\sum m_i} \qquad z_m = \frac{\sum z_i m_i}{\sum m_i}$$

Η ηλεκτροστατική ενέργεια είναι:

$$E = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{q_i q_j}{|r_i - r_j|}$$

Παράδειγμα #6 (Δηλώσεις)

```
#include <iostream>
#include <cmath>
using namespace std;

#define NMAX 100              // Μέγιστο πλήθος σωματιδίων

int main ( )
{
    int n;                            // Πλήθος σωματιδίων
    double x[NMAX], y[NMAX], z[NMAX];// Συντεταγμένες
    double m[NMAX], q[NMAX];          // Μάζες και φορτία
    int k, i, j;
    double sx, sy, sz, sm;
    double xm, ym, zm;                // Κέντρο μάζας
    double qtot;                      // Ολικό φορτίο
    double e;                         // Ηλεκτρ. ενέργεια.
    double d;
```

Συνεχίζεται...

Παράδειγμα #6 (Εισαγωγή)

...Συνέχεια

```
cout << "Πόσα σωματίδια? \n";
cin >> n;
if ( n < 1 || n > NMAX ) {
    cout << "Λανθασμένος αριθμός\n";
    return 1;
}
for ( k=0; k<n; ++k ) {
    cout << "Δώστε συντεταγμένες, μάζα και φορτίο "
         << "για το σωματίδιο " << k << endl;
    cin >> x[k] >> y[k] >> z[k] >> m[k] >> q[k];
}
```

Συνεχίζεται...

Παράδειγμα #6 (Ολικό φορτίο)

...Συνέχεια

```
    qtot = 0;
    for ( k=0; k<n; ++k )
        qtot += q[k];
    cout << "Το ολικό φορτίο είναι " << qtot << endl;
```

Συνεχίζεται...

Παράδειγμα #6 (Κέντρο μάζας)

...Συνέχεια

```
    sx = sy = sz = sm = 0;
    for ( k=0; k<n; ++k ) {
        sx += m[k]*x[k];
        sy += m[k]*y[k];
        sz += m[k]*z[k];
        sm += m[k];
    }
    xm = sx/sm;
    ym = sy/sm;
    zm = sz/sm;
    cout << "Το κέντρο μάζας είναι "
         << xm << " " << ym << " " << zm << endl;
```

Συνεχίζεται...

Παράδειγμα #6 (Ηλ. ενέργεια)

...Συνέχεια

```
    e = 0;
    for ( i=0; i<n-1; ++i )
        for ( j=i+1; j<n; ++j ) {
            d = sqrt( pow(x[i]-x[j],2)+pow(y[i]-y[j],2)+
                      pow(z[i]-z[j],2) );
            e += q[i]*q[j]/d;
        }
    cout << "Η ηλεκτροστατική ενέργεια είναι "
         << e << endl;
}
```


ΥΠΟΛΟΓΙΣΤΕΣ ΙΙ

Πίνακες χαρακτήρων

Μεταβλητές τύπου char

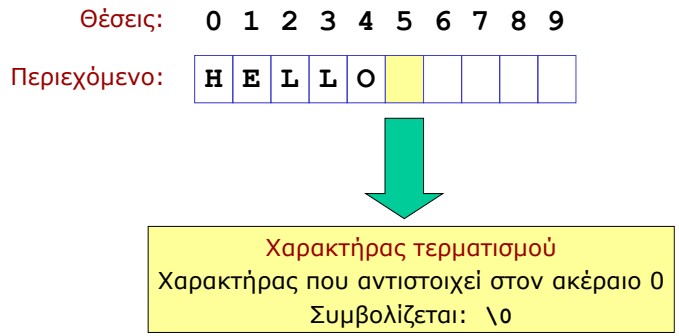
- Που χρησιμεύουν:
  Για την αποθήκευση μεμονωμένων χαρακτήρων.
- Δήλωση:
```
char c;
```
- Εισαγωγή από το πληκτρολόγιο:
```
cin >> c;
```
- Εμφάνιση στην οθόνη:
```
cout << c;
```
- Ανάθεση τιμής:
```
c = 'A';        // Ο χαρακτήρας A
c = '4';        // Ο χαρακτήρας 4
c = '\n';       // Ο χαρακτήρας αλλαγής γραμμής
c = '\0';       // Ο χαρακτήρας με αριθμό 0
```


Πίνακες χαρακτήρων (συμβολοσειρές)

- Που χρησιμεύουν:
  Για την αποθήκευση σειρών χαρακτήρων (λέξεων, φράσεων, κειμένου).
- Δήλωση:
```
#define MAX 100
char s[MAX];
```
- Εισαγωγή από το πληκτρολόγιο (μόνο μια λέξη):
```
cin >> s;
```
- Εμφάνιση στην οθόνη:
```
cout << s;
```


Χαρακτήρας τερματισμού

Από το μέγιστο δηλωμένο μέγεθος ενός πίνακα χαρακτήρων μπορεί να χρησιμοποιήσουμε λιγότερες θέσεις. Μετά το τέλος των χρησιμοποιημένων θέσεων τοποθετείται ο χαρακτήρας τερματισμού. Πχ για ένα πίνακα 10 χαρακτήρων:



Ποιος τοποθετεί το χαρακτήρα τερματισμού ;

Εξαρτάται πως δίνω τιμή στη συμβολοσειρά:

- `cin >> s;`
  Ο χαρακτήρας τερματισμού τοποθετείται αυτόματα από την cin.
- Μπορώ να δώσω τιμές με απευθείας ανάθεση (δεν συνιστάται):
```
s[0] = 'H';
s[1] = 'E';
s[2] = 'L';
s[3] = 'L';
s[4] = 'O';
s[5] = '\0';
```

Παράδειγμα #7

Κατασκευάστε πρόγραμμα που θα δέχεται ως είσοδο μια συμβολοσειρά και θα βρίσκει το μήκος της.

Παρατήρηση:
Θα πρέπει να βρούμε σε ποια θέση του πίνακα βρίσκεται ο χαρακτήρας τερματισμού.

Παράδειγμα #7

```
#include <iostream>
using namespace std;

#define MAX 100

int main ( )
{
    char s[MAX];
    int k;

    cout << "Δώστε μια λέξη ";
    cin >> s;

    for ( k=0; s[k] != '\0'; ++k )
        ;

    cout << "Το μήκος είναι " << k << endl;
}
```


Συναρτήσεις για συμβολοσειρές

- **strlen (s)**
  Επιστρέφει το πλήθος των χαρακτήρων της συμβολοσειράς s (Δεν συμπεριλαμβάνεται ο χαρακτήρας τερματισμού).
- **strcpy (s1, s2)**
  Αντιγράφει τη συμβολοσειρά s2 στη s1.
- **strcmp (s1, s2)**
  Συγκρίνει τις συμβολοσειρές s1 και s2 και επιστρέφει:
  -1 αν η s1 είναι λεξικογραφικά μικρότερη από την s2.
  0 αν η s1 είναι λεξικογραφικά ίση με την s2.
  1 αν η s1 είναι λεξικογραφικά μεγαλύτερη από την s2.

Για να χρησιμοποιήσουμε τις ανωτέρω συναρτήσεις για συμβολοσειρές πρέπει να βάλουμε:
#include <cstring>

Συναρτήσεις για συμβολοσειρές

- **gets (s)**
  Εισαγωγή από το πληκτρολόγιο της συμβολοσειράς s (μια ολόκληρη φράση μέχρι να πατήσουμε Enter). Τοποθετεί αυτόματα και το χαρακτήρα τερματισμού.

Για να χρησιμοποιήσουμε την ανωτέρω συνάρτηση πρέπει να βάλουμε:
#include <cstdio>

Παράδειγμα #8

Κατασκευάστε πρόγραμμα που θα δέχεται ως είσοδο μια συμβολοσειρά και θα μετατρέπει τυχόν κεφαλαίους χαρακτήρες σε μικρούς.

Παράδειγμα #8

Πως μετατρέπουμε ένα χαρακτήρα από κεφαλαίο σε μικρό;

Βασιζόμαστε στο ότι:

- Υπάρχει αντιστοιχία χαρακτήρων αριθμών.
- Στον πίνακα αντιστοίχισης υπάρχουν τρεις ομάδες χαρακτήρων: κεφαλαία, μικρά, ψηφία.
- Οι χαρακτήρες κάθε ομάδας βρίσκονται σε διαδοχικές θέσεις του πίνακα.

Ακέραιος	Χαρακτήρας
⋮	⋮
48	0
49	1
50	2
51	3
52	4
⋮	⋮
65	A
66	B
67	C
68	D
69	E
⋮	⋮
97	a
98	b
99	c
100	d
101	e
⋮	⋮

Παράδειγμα #8

Για να μετατρέψουμε ένα χαρακτήρα (`char c`) από κεφαλαίο σε μικρό:

- Βρίσκουμε πόσο απέχει από την αρχή των κεφαλαίων (δηλαδή από το A):
  ```
  d = c-'A';
  ```
- Προσθέτουμε την απόσταση αυτή στην αρχή των μικρών (δηλαδή στο a):
  ```
  new = 'a'+d;
  ```
- Ή συνολικά:
  ```
  new = 'a'-'A'+c;
  ```

Στις αριθμητικές πράξεις χρησιμοποιούμε χαρακτήρες. Εκεί εννοείται ο αντίστοιχος ακέραιος.

Παράδειγμα #8

Πως αποφασίζουμε αν ένας χαρακτήρας (`char c`) είναι κεφαλαίος ;

Πρέπει να βρίσκεται μεταξύ των χαρακτήρων A και Z

```
if ( c >= 'A' && c <= 'Z' )
```

Παράδειγμα #8

```
#include <iostream>
#include <cstdio>
#include <cstring>
using namespace std;

#define MAX 100

main ( )
{
    char s[MAX] ;
    char snew[MAX];
    int len, k;

    cout << "Εισάγετε μια συμβολοσειρά ";
    gets(s);
```

Συνεχίζεται...

Παράδειγμα #8

...Συνέχεια

```
    len = strlen(s);
    for (k=0; k<len; ++k)
        if (s[k]>='A' && s[k]<='Z')
            snew[k] = s[k]-'A'+'a';
        else
            snew[k] = s[k];

    snew[len] = '\0';
    cout << snew << endl;
}
```


Παράδειγμα #9

Κατασκευάστε πρόγραμμα που θα δέχεται ως είσοδο μια συμβολοσειρά και θα μετράει πόσοι:

1) Κεφαλαίοι χαρακτήρες
2) Μικροί χαρακτήρες
3) Αριθμητικά ψηφία
4) Σύμβολα

υπάρχουν στη συμβολοσειρά.

Παράδειγμα #9

```
#include <iostream>
#include <cstdio>
#include <cstring>
using namespace std;

#define MAX 100

int main ()
{
    char str[MAX];
    int len;                    // Μήκος αλφαριθμητικού
    int n1;                     // Πλήθος κεφαλαίων
    int n2;                     // Πλήθος μικρών
    int n3;                     // Πλήθος ψηφίων
    int n4;                     // Πλήθος σύμβόλων
    int k;

    cout << "Εισάγετε μια συμβολοσειρά ";
    gets(str);
```

Συνεχίζεται...

Παράδειγμα #9

...Συνέχεια

```
    n1 = n2 = n3 = n4 = 0;
    len = strlen(str);
    for (k=0; k<len; ++k)
        if (str[k]>='A' && str[k]<='Z')
            ++n1;
        else if (str[k]>='a' && str[k]<='z')
            ++n2;
        else if (str[k]>='0' && str[k]<='9')
            ++n3;
        else
            ++n4;

      cout << "Δώσατε: " << str << endl;
      cout << n1 << " Κεφαλαία \n";
      cout << n2 << " Μικρά \n";
      cout << n3 << " Ψηφία \n";
      cout << n4 << " Σύμβολα \n";
}
```


Παράδειγμα #10

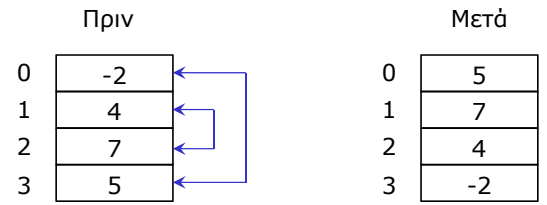
Μετρήστε πόσα θετικά, πόσα αρνητικά και πόσα μηδενικά στοιχεία έχει ένας πίνακας x με n στοιχεία.

Παράδειγμα #10

```
    nt = 0;                    // Πλήθος θετικών
    na = 0;                    // Πλήθος αρνητικών
    nm = 0;                    // Πλήθος μηδενικών
    for (k=0; k<n; ++k)
        if ( x[k] > 0 )
            ++nt;
        else if ( x[k] < 0 )
            ++na;
        else
            ++nm;
    cout << "Θετικά:    " << nt << endl;
    cout << "Αρνητικά: " << na << endl;
    cout << "Μηδενικά: " << nm << endl;
```


Παράδειγμα #11

Γράψτε τμήμα πρόγραμματος το οποίο θα αντιστρέφει τη σειρά αποθήκευσης των στοιχείων ενός πίνακα.



Εναλλάσσουμε τις τιμές των στοιχείων: 0 και 3
1 και 2

Παράδειγμα #11

Πως εναλλάσσω τις τιμές δύο απλών μεταβλητών ;

```
// Εναλλάσσω τις τιμές των μεταβλητών a, b
// χρησιμοποιώντας τη βοηθητική μεταβλητή t
    t = a;
    a = b;
    b = t;
```


Παράδειγμα #11

```
for (k=0; k<n/2; ++k) {
    t = x[k];          // Εναλλαγή των στοιχείων
    x[k] = x[n-k-1];   // x[k] και x[n-k-1]
    x[n-k-1] = t;
}
```

Θα λειτουργήσει σωστά για περιττό αριθμό στοιχείων ;

Παράδειγμα #12

Ένας πίνακας x έχει n στοιχεία. Δημιουργήστε ένα δεύτερο πίνακα z ο οποίος θα περιέχει μόνο τα θετικά στοιχεία του x.

Παράδειγμα #12

```
nz = 0;                 // Πλήθος στοιχείων του z
for (k=0; k<n; ++k)
    if (x[k]>0) {       // Εξετάζω το στοιχείο x[k]
        z[nz] = x[k];   // Τοποθετώ στον πίνακα z
        ++nz;           // Αυξάνω το πλήθος των
    }                   // στοιχείων του z
```

Παράδειγμα #13

Ένας πίνακας x έχει n στοιχεία. Γράψτε τμήμα προγράμματος που θα γεμίζει τα στοιχεία του πίνακα με την ακολουθία:

$$x_k = 0.7x_{k-1} + 0.2x_{k-2}$$

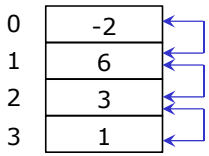
με αρχικές τιμές: $x_0 = 0$ και $x_1 = 1$

Παράδειγμα #13

```
x[0] = 0;
x[1] = 1;
for (k=2; k<n; ++k)
    x[k] = 0.7*x[k-1] + 0.2*x[k-2];
```


Παράδειγμα #14

Ένας πίνακας x έχει n στοιχεία. Γράψτε τμήμα προγράμματος που θα ελέγχει αν τα στοιχεία του είναι ταξινομημένα σε αύξουσα σειρά.



Ελέγχουμε ανά δύο τα στοιχεία:

0 και 1

1 και 2

2 και 3

.....

n-2 και n-1

Παράδειγμα #14

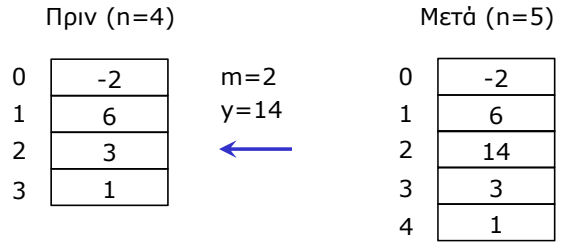
```
    isord = 1;              //  Δείχνει αν είναι σε σειρά
    for (k=0; k<n-1; ++k)
        if (x[k]>x[k+1]) {
            isord = 0;      //  Βρήκαμε ένα ζευγάρι που
            break;          //  δεν είναι σε αύξουσα σειρά
        }

//  Εμφάνιση αποτελέσματος στην οθόνη.
    if (isord==1)
        cout << "Σε αύξουσα σειρά" << endl;
    else
        cout << "Όχι σε αύξουσα σειρά" << endl;
```

Παράδειγμα #14 (2ος τρόπος)

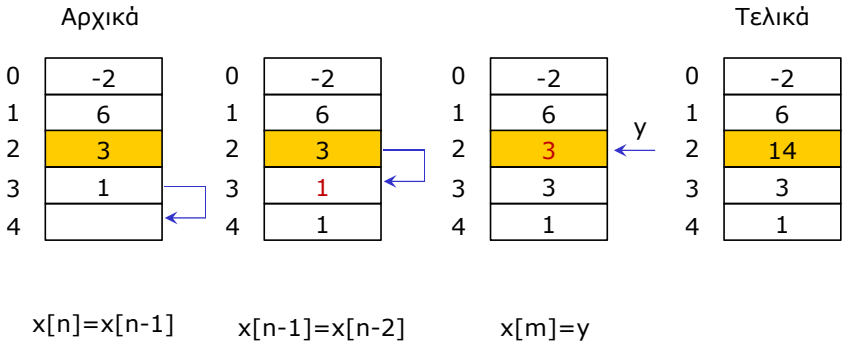
```
    isord = 1;                  // Δείχνει αν είναι σε σειρά
    for (k=0; k<n-1 && isord==1; ++k)
        if (x[k]>x[k+1])        // Βρήκαμε ένα ζευγάρι που
            isord = 0;          // δεν είναι σε αύξουσα σειρά

//  Εμφάνιση αποτελέσματος στην οθόνη.
    if (isord==1)
        cout << "Σε αύξουσα σειρά" << endl;
    else
        cout << "Όχι σε αύξουσα σειρά" << endl;
```


Παράδειγμα #15

Ένας πίνακας x έχει n στοιχεία. Γράψτε τμήμα προγράμματος που θα εισάγει στη θέση με δείκτη m ένα αριθμό y αυξάνοντας κατά ένα το πλήθος των στοιχείων του πίνακα.



Παράδειγμα #15

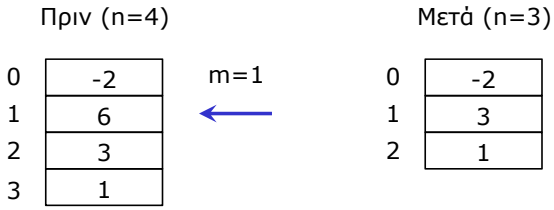


Παράδειγμα #15

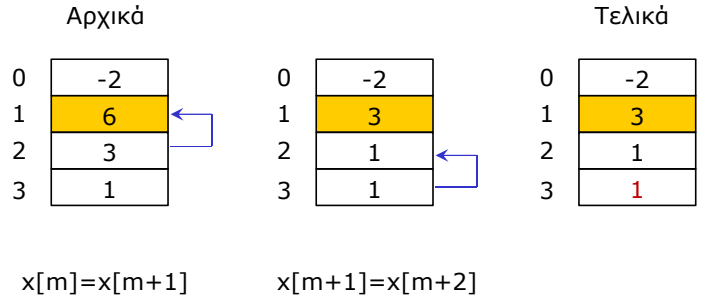
```
    for (k=n-1; k>=m; --k)  // Μεταφορά κατά μία
        x[k+1] = x[k];      // θέση προς τα κάτω

    x[m] = y;               // Εισαγωγή νέου στοιχείου
    ++n;                    // Νέο πλήθος στοιχείων
```

Παράδειγμα #16

Ένας πίνακας x έχει n στοιχεία. Γράψτε τμήμα προγράμματος που θα διαγράφει το περιεχόμενο της θέσης m μειώνοντας κατά ένα το πλήθος των στοιχείων του πίνακα.



Παράδειγμα #16



Παράδειγμα #16

```
for (k=m; k<n-1; ++k)    // Μεταφορά κατά μία
    x[k] = x[k+1];       // θέση προς τα πάνω

--n;                     // Νέο πλήθος στοιχείων
```