Διδιάστατοι Πίνακες στη C

- Οι διδιάστατοι πίνακες μοιάζουν με τους γνωστούς μαθηματικούς πίνακες δύο διαστάσεων της άλγεβρας και αποτελούνται και αυτοί από γραμμές και στήλες

- Για να ορίσουμε έναν διδιάστατο πίνακα πρέπει να δηλώσουμε το `όνομα του πίνακα`, τον `τύπο δεδομένων` των στοιχείων του πίνακα, καθώς και το `πλήθος των γραμμών` και `των στηλών` του

- Η γενική περίπτωση ορισμού ενός διδιάστατου πίνακα είναι:

```
τύπος_δεδομένων  όνομα_πίνακα [πλήθος_γραμμών] [πλήθος_στηλών]
```

- Το πλήθος των στοιχείων ενός διδιάστατου πίνακα είναι ίσο με το γινόμενο του πλήθους των γραμμών του επί το πλήθος των στηλών του

Δήλωση διδιάστατου Πίνακα

- Π.χ. η εντολή `int array[10][5];` δηλώνει έναν διδιάστατο πίνακα με όνομα `array`, ο οποίος περιέχει `50` στοιχεία και καθένα από αυτά τα στοιχεία είναι ένας ακέραιος αριθμός (`int`)

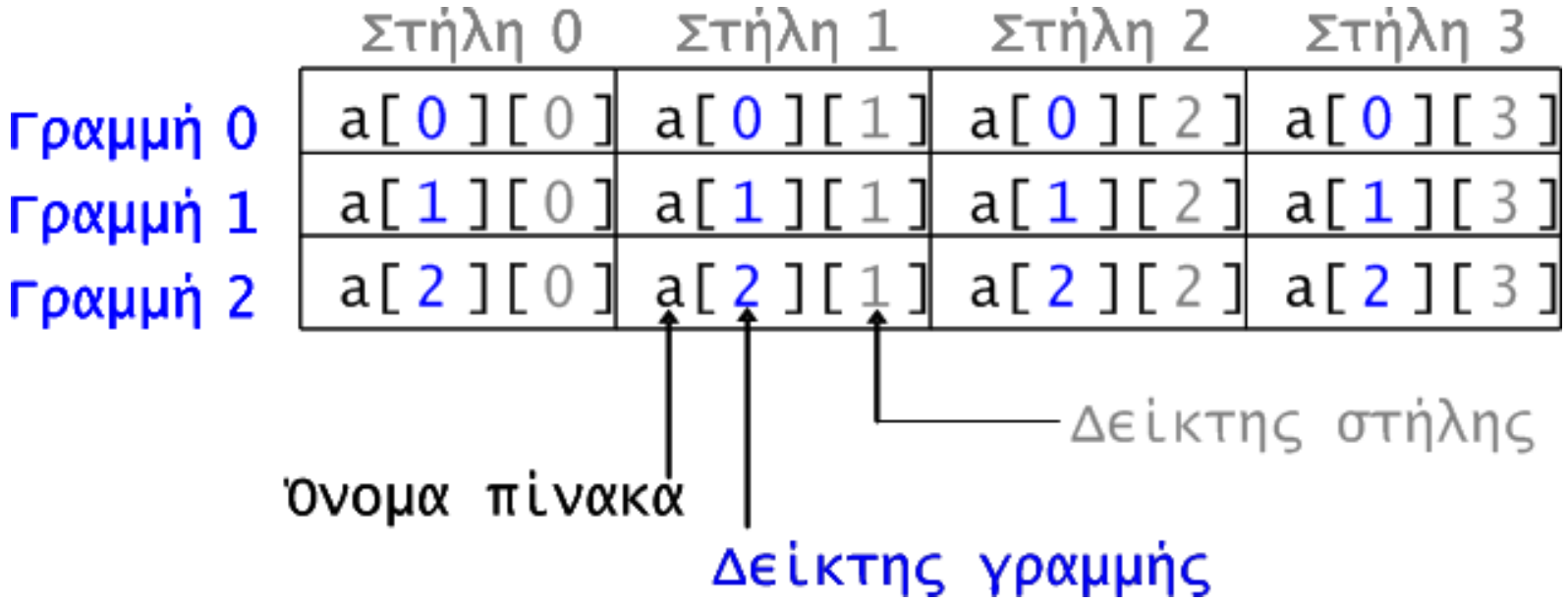
- Παρομοίως, η εντολή `char array[3][4];` δηλώνει έναν διδιάστατο πίνακα με όνομα `array`, ο οποίος περιέχει `12` στοιχεία και καθένα από αυτά τα στοιχεία είναι ένας χαρακτήρας (`char`)

- Για να αναφερθούμε σε κάποιο στοιχείο ενός διδιάστατου πίνακα γράφουμε το όνομα του πίνακα συνοδευόμενο από τον αριθμό γραμμής (ή αλλιώς τον δείκτη γραμμής) και τον αριθμό στήλης (ή αλλιώς τον δείκτη στήλης) του συγκεκριμένου στοιχείου του πίνακα

- Οι αριθμοί γραμμών και στηλών πρέπει να δηλώνονται μέσα σε αγκύλες `[][]`

Γραφική Αναπαράσταση

Π.χ. αν έχουμε δηλώσει τον πίνακα: `int a[3][4]`



Παρατηρήστε ότι – όπως και στους μονοδιάστατους πίνακες έτσι και στους διδιάστατους - η αρίθμηση για τις γραμμές και τις στήλες ξεκινάει από το 0

Διδιάστατοι Πίνακες και Μνήμη

- Κατά τη δήλωση του πίνακα, ο μεταγλωττιστής – όπως και στην περίπτωση των μονοδιάστατων πινάκων – δεσμεύει ένα τμήμα μνήμης από τη στοίβα (stack) για να αποθηκεύσει τα στοιχεία του

- Π.χ. με τη δήλωση `int array[10][5];` ο μεταγλωττιστής δεσμεύει 200 bytes για να αποθηκεύσει τα 50 στοιχεία του πίνακα (αφού κάθε στοιχείο του πίνακα – ως ακέραια μεταβλητή – απαιτεί 4 bytes).

- Τα στοιχεία του πίνακα αποθηκεύονται σε διαδοχικές θέσεις στη μνήμη ξεκινώντας από τα στοιχεία της 1ης γραμμής, συνεχίζοντας με τα στοιχεία της 2ης γραμμής, κ.ο.κ.

Διδιάστατοι Πίνακες και Μνήμη

- Συγκεκριμένα, η θέση μνήμης ενός τυχαίου στοιχείου `a[i][j]` του πίνακα `a[ROWS][COLS]` υπολογίζεται σύμφωνα με τον τύπο:

```
θέση_μνήμης = (i * COLS) + j + 1
```

- Π.χ. η θέση μνήμης του στοιχείου `a[2][1]` ενός πίνακα με 3 γραμμές και 4 στήλες είναι η:

```
θέση_μνήμης = (i * COLS) + j + 1 = 2*4 + 1 + 1 = 10
```

- Παρατηρήστε, ότι για να υπολογιστεί η θέση μνήμης του στοιχείου στη μνήμη δεν χρειάζεται να είναι γνωστή η πρώτη διάσταση του πίνακα (π.χ. `ROWS`), αλλά μόνο το πλήθος των στηλών του (π.χ. `COLS`)

Παράδειγμα

- Ο πιο απλός τρόπος αρχικοποίησης ενός διδιάστατου πίνακα είναι να αποδώσουμε αρχικές τιμές σε κάθε ένα στοιχείο του, αφού προηγουμένως έχουμε ορίσει τον πίνακα

```
int  i,j,arr[3][4];  /*  Δήλωση  διδιάστατου  πίνακα  με
στοιχεία 12 ακεραίους. */
i = j = 2;

arr[0][0] = 100; /* Η τιμή του 1ου στοιχείου του πίνακα
(1ο στοιχείο της 1ης γραμμής) γίνεται 100. */
arr[1][1] = 200; /* Η τιμή του 6ου στοιχείου (2ο στοιχείο
της 2ης γραμμής) του πίνακα γίνεται 200. */
arr[2][3]  =  array[0][0];  /*  Η  τιμή  του  τελευταίου
στοιχείου του πίνακα (4ο στοιχείο της 3ης γραμμής) γίνεται
ίση με την τιμή του 1ου στοιχείου. */
arr[i][j] = 300; /* Η τιμή του προτελευταίου στοιχείου του
πίνακα (3ο στοιχείο της 3ης γραμμής) γίνεται 300. */
arr[i-2][j-2]  =  300;  /*  Η  τιμή  του  1ου  στοιχείου  του
πίνακα (1ο στοιχείο της 1ης γραμμής) γίνεται 300. */
```

Δήλωση Πίνακα και απόδοση αρχικών τιμών (Ι)

- Όπως και στην περίπτωση των μονοδιάστατων πινάκων, η C υποστηρίζει παρόμοιους τρόπους απόδοσης αρχικών τιμών στα στοιχεία ενός διδιάστατου πίνακα, ταυτόχρονα με τη δήλωση του πίνακα

- Σε όλες τις παρακάτω περιπτώσεις οι αρχικές τιμές αποδίδονται στα στοιχεία του πίνακα ανά γραμμή, ξεκινώντας από τα στοιχεία της πρώτης γραμμής, συνεχίζοντας στα στοιχεία της δεύτερης γραμμής, κ.ο.κ.

Δήλωση Πίνακα και απόδοση αρχικών τιμών (ΙΙ)

1. Τη δήλωση του πίνακα την ακολουθεί ο τελεστής = και οι τιμές των στοιχείων κάθε γραμμής περικλείονται ανάμεσα σε εσωτερικά άγκιστρα { } διαχωριζόμενες μεταξύ τους με κόμμα (,)

```
int arr[3][3] = {{10,20,30},
                 {40,50,60},
                 {70,80,90}};
```

Σε αυτό το παράδειγμα:
η τιμή του `arr[0][0]` γίνεται `10`
η τιμή του `arr[0][1]` γίνεται `20`
η τιμή του `arr[0][2]` γίνεται `30` κ.ο.κ.

Εναλλακτικά, μπορούμε να παραλείψουμε τα εσωτερικά άγκιστρα και να γράψουμε:

```
int arr[3][3] = {10,20,30,40,50,60,70,80,90};
```

Δήλωση Πίνακα και απόδοση αρχικών τιμών (ΙΙΙ)

2. Τη δήλωση του πίνακα την ακολουθεί ο τελεστής = και μέσα στα εσωτερικά άγκιστρα { } δεν υπάρχουν τιμές για όλα τα στοιχεία της κάθε γραμμής του πίνακα
   Σε αυτή την περίπτωση ο μεταγλωττιστής αποδίδει την τιμή 0 στα υπόλοιπα στοιχεία της κάθε γραμμής του πίνακα

```
int arr[3][3] = {{10,20},
                 {40,50},
                 {70}};
```

Π.χ. στο παραπάνω παράδειγμα:
οι τιμές των `arr[0][2]` , `arr[1][2]` , `arr[2][1]` και `arr[2][2]` γίνονται 0

Δήλωση Πίνακα και απόδοση αρχικών τιμών (IV)

3. Αν έχουμε παραλείψει τιμές για όλα τα στοιχεία κάποιας γραμμής, τότε ο μεταγλωττιστής αποδίδει την τιμή `0` σε όλα τα στοιχεία της γραμμής

```
int arr[3][3] = {{10,20,30}};
```

Π.χ. στο παραπάνω παράδειγμα:
οι τιμές όλων των στοιχείων της 2ης και της 3ης γραμμήςγίνονται `0`

4. Αν τέλος έχουμε παραλείψει τα εσωτερικά άγκιστρα `{}` και δεν αποδίδουμε τιμές για όλα τα στοιχεία του πίνακα, τότε ο μεταγλωττιστής αποδίδει την τιμή `0` στα υπόλοιπα στοιχεία του πίνακα

```
int arr[3][3] = {10,20};
```

Π.χ. στο παραπάνω παράδειγμα:
οι τιμές των `arr[0][0]` και `arr[0][1]` γίνονται `10` και `20` αντίστοιχα, ενώ όλων των υπολοίπων στοιχείων γίνονται `0`

Δήλωση Πίνακα και απόδοση αρχικών τιμών (V)

5. Το πλήθος των γραμμών δεν είναι υποχρεωτικό να δηλωθεί. Πρέπει όμως υποχρεωτικά να δηλωθεί ο αριθμός των στηλών

```
int arr[][3] = {10,20,30,40,50,60};
```

- Π.χ. στο παραπάνω παράδειγμα:
- ο μεταγλωττιστής θα δημιουργήσει αυτόματα έναν διδιάστατο πίνακα ακεραίων με δύο γραμμές και τρεις στήλες, διότι οι αρχ τιμές που αποδίδονται στα στοιχεία του πίνακα (έξι τιμές συνο απαιτούν πίνακα με τουλάχιστον δύο γραμμές και τρεις στήλες
- Συγκεκριμένα, η τιμή:
- του `arr[0][0]` γίνεται `10`
- του `arr[0][1]` γίνεται `20`
- του `arr[0][2]` γίνεται `30` κ.ο.κ.

Δήλωση Πίνακα και απόδοση αρχικών τιμών (VI)

6. Οι τρόποι αρχικοποίησης ενός διδιάστατου πίνακα που παρουσιάστηκαν χρησιμοποιούνται όταν θέλουμε να δώσουμε συγκεκριμένες τιμές στα στοιχεία ενός πίνακα

   Για πίνακες μεγαλύτερου μεγέθους και όταν οι αρχικές τιμές του δεν απαιτείται να είναι συγκεκριμένες, συνήθως χρησιμοποιούνται διπλοί επαναληπτικοί βρόχοι

```c
#include <stdio.h>
int main()
{
	int i,j,arr[50][100];

	for(i = 0; i < 50; i++)
	{
		for(j = 0; j < 100; j++)
		{
			arr[i][j] = 300;
			printf("arr[%d][%d] = %d\n",i,j,arr[i][j]);
		}
	}
	return 0;
}
```

Μεθοδολογία (Ι)

ΒΑΣΙΚΗ ΕΠΕΞΕΡΓΑΣΙΑ (*int A[4][5];*)

ΕΙΣΟΔΟΣ κατά Γραμμές	**ΕΙΣΟΔΟΣ** κατά Στήλες

for (i=0; i<4; i++)
{
  for (j=0; j<5; j++)
  {
  printf("Give value:");
  scanf("%i", &A[i][j]);
  }
}
	for (j=0; j<5; j++)
	{
	  for (i=0; i<4; i++)
	  {
	  printf("Give value:");
	  scanf("%i", &A[i][j]);
	  }
	}

ΕΞΟΔΟΣ κατά Γραμμές	**ΕΞΟΔΟΣ** κατά Στήλες

for (i=0; i<4; i++)
{
  for (j=0; j<5; j++)
  {
  printf("%i ",A[i][j]);
  }
  printf("\n");
}
	for (j=0; j<5; j++)
	{
	  for (i=0; i<4; i++)
	  {
	  printf("%i ",A[i][j]);
	  }
	  printf("\n");
	}

Μεθοδολογία (ΙΙ)

ΒΑΣΙΚΗ ΕΠΕΞΕΡΓΑΣΙΑ (*int A[4][5];*)

ΑΡΧΙΚΟΠΟΙΗΣΗ σε σταθερά c (π.χ. c = 0)	**ΑΡΧΙΚΟΠΟΙΗΣΗ**

for (i=0; i<4; i++)
{
  for (j=0; j<5; j++)
  {
  A[i][j] = 0;
  }
}
	for (j=0; j<5; j++)
	{
	  for (i=0; i<4; i++)
	  {
	  if (i<j) A[i][j] = 0;
	  else if (i == j) A[i][j] = 1;
	  else A[i][j] = 2;
	  }
	}

Μεθοδολογία (ΙΙΙ)

ΒΑΣΙΚΗ ΕΠΕΞΕΡΓΑΣΙΑ (*int A[4][5];*)

ΑΘΡΟΙΣΜΑ , ΜΕΣΟΣ ΟΡΟΣ κατά γραμμές

```
for (i=0; i<4; i++)
{
    sr[i] = 0;
    for (j=0; j<5; j++)
    {
        sr[i] += A[i][j];
    }
    mr[i] = sr[i] / 5;
}
```

ΑΘΡΟΙΣΜΑ , ΜΕΣΟΣ ΟΡΟΣ κατά στήλες

```
for (j=0; j<5; j++)
{
    sc[j] = 0;
    for (i=0; i<4; i++)
    {
        sc[i] += A[i][j];
    }
    mc[i] = sc[i] / 4;
}
```

ΑΘΡΟΙΣΜΑ , ΜΕΣΟΣ ΟΡΟΣ συνολικά

```
s = 0;
for (i=0; i<4; i++)
{
    for (j=0; j<5; j++)
    {
        s += A[i][j];
    }
}
m = s/(4*5);
```

Μεθοδολογία (IV)

ΜΕΓΙΣΤΟ & Θέση κατά γραμμές

```
for (i=0; i<4; i++)
{
      maxr[i] = A[i][0];
      maxrp[i] = 0;
      for (j=0; j<5; j++)
      {
            if (A[i][j] > maxr[i])
            {
                  maxr[i] = A[i][j];
                  maxrp[i] = j;
            }
       }
}
```

ΜΕΓΙΣΤΟ & Θέση κατά στήλες

```
for (j=0; j<5; j++)
{
    maxc[j] = A[0][j];
    maxcp[j] = 0;
    for (i=0; i<4; i++)
    {
          if (A[i][j] > maxc[j])
          {
                maxc[j] = A[i][j];
                maxcp[j] = i;
          }
     }
}
```

ΜΕΓΙΣΤΟ & Θέση συνολικά

```
max = A[0][0];
maxpr = 0;
maxpc = 0;
for (i=0; i<4; i++)
{
      for (j=0; j<5; j++)
      {
            if (A[i][j] > max)
            {
                  max = A[i][j];
                  maxpr = i;
                  maxpc = j;
            }
       }
}
```

Μεθοδολογία (V)

ΕΛΑΧΙΣΤΟ & Θέση κατά γραμμές

```
for (i=0; i<4; i++)
{
      minr[i] = A[i][0];
      minrp[i] = 0;
      for (j=0; j<5; j++)
      {
            if (A[i][j] < minr[i])
            {
                  minr[i] = A[i][j];
                  minrp[i] = j;
            }
       }
}
```

ΕΛΑΧΙΣΤΟ & Θέση κατά στήλες

```
for (j=0; j<5; j++)
{
    minc[j] = A[0][j];
    mincp[j] = 0;
    for (i=0; i<4; i++)
    {
          if (A[i][j] < minc[j])
          {
                minc[j] = A[i][j];
                mincp[j] = i;
          }
     }
}
```

ΕΛΑΧΙΣΤΟ & Θέση συνολικά

```
min = A[0][0];
minpr = 0;
minpc = 0;
for (i=0; i<4; i++)
{
      for (j=0; j<5; j++)
      {
            if (A[i][j] < min)
            {
                  min = A[i][j];
                  minpr = i;
                  minpc = j;
            }
       }
}
```

Μεθοδολογία (VI)

ΤΑΞΙΝΟΜΗΣΗ κατά γραμμές	**ΤΑΞΙΝΟΜΗΣΗ** κατά στήλες

for (k=0; k<4; k++)
{
for (i=1;i<5;i++)
{
for (j=4; j>=i; j--)
{
if (A[k][j-1] > A[k][j])
{
temp = A[k][j-1];
A[k][j-1] = A[k][j];
A[k][j] = temp;
}
}
}
}
	for (k=0; k<5; k++)
	{
	for (i=1;i<4;i++)
	{
	for (j=3; j>=i; j--)
	{
	if (A[j-1][k] > A[j][k])
	{
	temp = A[j-1][k];
	A[j-1][k] = A[j][k];
	A[j][k] = temp;
	}
	}
	}
	}

Παραδείγματα (Ι)

- Γράψτε ένα πρόγραμμα το οποίο να αρχικοποιεί έναν διδιάστατο πίνακα 5×5 ως τον μοναδιαίο τετραγωνικό 5×5 πίνακα και να εμφανίζει τα στοιχεία του πίνακα στην οθόνη υπό τη μορφή πίνακα 5×5 της άλγεβρας

```
#include <stdio.h>

#define SIZE 5

int main()
{
	int i,j,arr[SIZE][SIZE] = {0}; /* Αρχικοποίηση των στοιχείων του πίνακα με την τιμή 0. */

	for(i = 0; i < SIZE; i++)
	{
		for(j = 0; j < SIZE; j++)
		{
			if(i == j) /* Για τα στοιχεία της κυρίας διαγωνίου ισχύει i=j */
				arr[i][j] = 1;

			printf("%3d",arr[i][j]);
		}
		printf("\n"); /* Χρησιμοποιείται για τη δημιουργία των διαφορετικών γραμμών του πίνακα. */
	}
	return 0;
}
```

Παραδείγματα (ΙΙ)

- Γράψτε ένα πρόγραμμα το οποίο να διαβάζει ακέραιους αριθμούς, να τους αποθηκεύει σε έναν 2×4 πίνακα και να εμφανίζει τη μικρότερη και τη μεγαλύτερη τιμή κάθε γραμμής του πίνακα

```c
#include <stdio.h>

#define ROWS 2
#define COLS 4

int main()
{
	int i,j,max,min,arr[ROWS][COLS];

	for(i = 0; i < ROWS; i++)
	{
		for(j = 0; j < COLS; j++)
		{
			printf("Enter the element arr[%d][%d]: ",i,j);
			scanf("%d",&arr[i][j]);
		}
	}
	for(i = 0; i < ROWS; i++)
	{
		/* Αρχικοποίηση των min και max με το πρώτο στοιχείο
της κάθε γραμμής. */
		min = max = arr[i][0];
		for(j = 0; j < COLS; j++)
		{
			if(arr[i][j] >= max)
				max = arr[i][j];

			if(arr[i][j] <= min)
				min = arr[i][j];
		}
		printf("Row_%d: Max = %d Min = %d\n",i+1,max,min);
	}
	return 0;
}
```

Ασκήσεις (Ι)

Δίνεται ο πίνακας Α (σχήμα 1) και το παρακάτω τμήμα προγράμματος. Αυτό το τμήμα προγράμματος χρησιμοποιεί τον πίνακα Α, με τις τιμές των στοιχείων του, όπως αυτές φαίνονται στο σχήμα 1.

- Να σχεδιάσετε στο τετράδιό σας τον πίνακα Α με τις τιμές που θα έχουν τα στοιχεία του, μετά την εκτέλεση του τμήματος προγράμματος. *Μονάδες 15*
- Ποια είναι η τιμή της μεταβλητής sum που εμφανίζεται στο τέλος; *Μονάδες 5*

Απολυτήριες εξετάσεις Δ' Τάξης του Εσπερινού Λυκείου 2003

```
sum = 0;
for(i=0; i<5; i++)
{
        for(i=0; i<5; i++)
        {
                if(i == j) sum += A[i][j];
                else A[i,j] = 0;
         }
}
printf("sum = %d\n", sum);
```

1	-1	7	1	1
6	2	0	8	-2
4	9	3	3	0
3	5	-4	2	1
0	1	2	0	1

Σχήμα 1: Πίνακας Α

Μελέτη Άσκησης

Μια εταιρεία έχει 5 πωλητές που διακινούν τα προϊόντα της και διατηρεί τις πωλήσεις κάθε πωλητή ανά δίμηνο για ένα έτος. Να γραφεί πρόγραμμα που να υπολογίζει και να εμφανίζει:

- τις συγκεντρωτικές πωλήσεις της εταιρείας για το έτος
- τον μέσο όρο διμηνιαίων πωλήσεων της εταιρείας
- τον καλύτερο πωλητή, και τη διαφορά πωλήσεων του καλύτερου και του χειρότερου πωλητή
- το δίμηνο που η εταιρεία είχε τις περισσότερες πωλήσεις