ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν Πανεπιστήμιον Αθηνών

Σχολή Θετικών Επιστημών
ΤΜΗΜΑ ΑΕΡΟΔΙΑΣΤΗΜΙΚΗΣ ΕΠΙΣΤΗΜΗΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ

104
Αλγόριθμοι, Προγραμματισμός και Δομές Δεδομένων

Εργαστήριο – Κεφάλαιο 7β & 8α
Δυσδιάστατοι πίνακες & Δείκτες

Διδάσκουσα:

Σοφία Αλμπάνη

Άσκηση 7.4

7.4) Να γραφεί πρόγραμμα σε C που να:
α) δημιουργεί έναν **δισδιάστατο πίνακα** ακεραίων διαστάσεων **6×8**, να τον γεμίζει με **τυχαίους αριθμούς στο διάστημα [0–20]** και να τον εμφανίζει, και

β) να επιτρέπει την **ανταλλαγή δύο γραμμών ή δύο στηλών** του πίνακα.

- Ο χρήστης επιλέγει αν θέλει να γίνει ανταλλαγή γραμμών ή στηλών πληκτρολογώντας: Enter swap type (1:rows):
  - Αν δώσει 1, γίνεται ανταλλαγή γραμμών.
  - Αν δώσει άλλη τιμή, γίνεται ανταλλαγή στηλών.
- Ο χρήστης εισάγει τους αριθμούς των δύο γραμμών ή στηλών που θέλει να ανταλλάξει (αριθμημένες από 1 έως ROWS ή COLS αντίστοιχα).

Στόχος:
Εξάσκηση στη χρήση **δισδιάστατων πινάκων**, στη **διαχείριση γραμμών και στηλών**, και στη χρήση **τυχαίων αριθμών** και **δομών επανάληψης** για επεξεργασία δεδομένων πίνακα.

Ενδεικτική Λύση 7.4 – μέρος α

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

#define ROWS 6
#define COLS 8

int main()
{          // τυχαίες τιμές στον πίνακα και εμφάνιση πίνακα
           int i, j, a, b, max, tmp, type, arr[ROWS][COLS];

           srand(time(NULL));
           for(i = 0; i < ROWS; i++)
           {
                      for(j = 0; j < COLS; j++)
                      {
                                 arr[i][j] = rand()%21;
                                 printf("%5d", arr[i][j]);
                      }
                      printf("\n");
           }
              // [μέρος β…]
           /* Εμφάνιση πίνακα. */
           for(i = 0; i < ROWS; i++)
           {
                      for(j = 0; j < COLS; j++)
                                 printf("%5d", arr[i][j]);
                      printf("\n");
           }
}
```

Ενδεικτική Λύση 7.4 – μέρος β

```
// Επιλογή διαστάσεων για αλλαγή
printf("Enter swap type (1:rows): ");
scanf("%d", &type);
if(type == 1)
        max = ROWS;
else
        max = COLS;
do
{
        printf("Enter dim_1[1-%d]: ", max);
        scanf("%d", &a);
} while(a < 1 || a > max);

do
{
        printf("Enter dim_2[1-%d]: ", max);
        scanf("%d", &b);
} while(b < 1 || b > max);

a--; // Αφαιρούμε 1, η αρίθμηση αρχίζει από 0.
b--;

    // συνέχεια δίπλα
```

```
// Αλλαγή στηλών ή γραμμών
if(type == 1)
{
        for(i = 0; i < COLS; i++)
        {
                tmp = arr[a][i];
                arr[a][i] = arr[b][i];
                arr[b][i] = tmp;
        }
}
else
{
        for(i = 0; i < ROWS; i++)
        {
                tmp = arr[i][a];
                arr[i][a] = arr[i][b];
                arr[i][b] = tmp;
        }
}
```

Άσκηση 7.5

7.5) Να γραφεί πρόγραμμα σε C το οποίο υπολογίζει το **προϊόν δύο πινάκων.**

α) Να διαβαστεί από τον χρήστη ο πίνακας **a**, διαστάσεων **2×3** και ο πίνακας **b**, διαστάσεων **3×2**.

β) Το πρόγραμμα πρέπει να υπολογίσει τον πίνακα **c**, όπου: $\boldsymbol{c = a \times b}$ (Δηλαδή ο **c** θα έχει διαστάσεις **2×2)**.

- Ο υπολογισμός να γίνει με τον τύπο πολλαπλασιασμού πινάκων:

$$c[i][j] = \sum_{k=0}^{M-1} a\,[i][k] \cdot b[k][j]$$

- Να εμφανίζεται ο πίνακας **c** μορφοποιημένος.

Στόχος:
Εξάσκηση σε **πολλαπλασιασμό πινάκων**, χρήση **τρών εμφωλευμένων βρόχων**, διαχείριση δισδιάστατων πινάκων διαφορετικών διαστάσεων, μαθηματικές πράξεις σε δομές δεδομένων.

Ενδεικτική Λύση 7.5 – μέρος α

```
#include <stdio.h> #define N 2
#define M 3
int main()
{
	int i, j, k, a[N][M], b[M][N], c[N][N] = {0};

	for(i = 0; i < N; i++)
	{
		for(j = 0; j < M; j++)
		{
			printf("Enter element a[%d][%d]: ", i, j);
			scanf("%d", &a[i][j]);
		}
	}
	for(i = 0; i < M; i++)
	{
		for(j = 0; j < N; j++)
		{
			printf("Enter element b[%d][%d]: ", i, j);
			scanf("%d", &b[i][j]);
		}
	}}
```

Ενδεικτική Λύση 7.5 – μέρος β

```
for(i = 0; i < N; i++)
        for(j = 0; j < N; j++)
                for(k = 0; k < M; k++)
                        c[i][j] += a[i][k] * b[k][j];

printf("\nArray c = a x b (%dx%d)\n", N, N);
printf("-----------------\n");
for(i = 0; i < N; i++)
{
        for(j = 0; j < N; j++)
                printf("%5d", c[i][j]);
        printf("\n");
}
```

Άσκηση 8.1

8.1) Να γραφεί πρόγραμμα σε C που εισάγει δύο ακέραιους αριθμούς και χρησιμοποιεί **δείκτες (pointers)** για να εμφανίσει:

i. Τις **διευθύνσεις μνήμης** των δύο μεταβλητών.
ii. Τις **τιμές** που περιέχουν οι μεταβλητές, μέσω αποαναφοράς των δεικτών (χρήση *ptr).
iii. Τις **διευθύνσεις μνήμης των ίδιων των δεικτών**.

Στόχος:
Η άσκηση στοχεύει στην κατανόηση της σχέσης ανάμεσα σε: μεταβλητές, διευθύνσεις μνήμης, δείκτες, και αποαναφορά τιμών μέσω δεικτών.

Ενδεικτική Λύση 8.1

```
#include <stdio.h>
int main()
{
        int *ptr1, *ptr2, i, j;

        printf("Enter numbers: ");
        scanf("%d%d", &i, &j);

        ptr1 = &i;
        ptr2 = &j;

        printf("Num1 address = %p\n", ptr1);
        printf("Num2 address = %p\n", ptr2);

        printf("Ptr1 content = %d\n", *ptr1); /* Εμφανίζεται η τιμή του i. */
        printf("Ptr2 content = %d\n", *ptr2); /* Εμφανίζεται η τιμή του j. */

       printf("Ptr1 address = %p\n", &ptr1);
        printf("Ptr2 address = %p\n", &ptr2);
        }
```

Άσκηση 8.2

8.2) Να γραφεί πρόγραμμα σε C που:

- Δηλώνει τρεις ακέραιες μεταβλητές και τρεις δείκτες σε ακέραιο.
- Κάνει τους δείκτες να «δείχνουν» στις αντίστοιχες μεταβλητές.
- Τροποποιεί τις μεταβλητές έμμεσα μέσω των δεικτών, χρησιμοποιώντας πράξεις με αποαναφορά (dereferencing).

Στο τέλος εμφανίζει τις τελικές τιμές των μεταβλητών μέσω των δεικτών.

Στόχος:

Άσκηση στη χρήση δεικτών και στην πρόσβαση τιμών μέσω δεικτών.

Ενδεικτική Λύση 8.2

```
#include <stdio.h>
int main()
{
        int *ptr1, *ptr2, *ptr3, i = 10, j = 20, k = 30;

        ptr1 = &i;
        i = 100;

        ptr2 = &j;
        j = *ptr2 + *ptr1;

        ptr3 = &k;
        k = *ptr3 + *ptr2;
        printf("%d %d %d\n", *ptr1, *ptr2, *ptr3);
        }
```

Άσκηση 8.3

8.3) Να γραφεί πρόγραμμα σε C που:

- Δηλώνει έναν ακέραιο και έναν πραγματικό αριθμό (double), καθώς και δύο δείκτες αντίστοιχου τύπου.
- Κάνει τους δείκτες να δείχνουν στις μεταβλητές.
- Αναθέτει την τιμή της πραγματικής μεταβλητής στον ακέραιο μέσω των δεικτών.
- Εμφανίζει:
  - τη νέα τιμή του ακεραίου,
  - το μέγεθος (σε bytes) ενός δείκτη σε int,
  - το μέγεθος ενός δείκτη σε double,
  - και το μέγεθος του τύπου double.

Στόχος:
Κατανόηση ανάθεσης τιμών μέσω δεικτών και του μεγέθους των τύπων/δεικτών.

Ενδεικτική Λύση 8.3

```
#include <stdio.h> /* ’σκηση Κ.8.6. */
int main(void)
{
        int *ptr1, i = 10;
        double *ptr2, j = 1.234;

        ptr1 = &i;
        ptr2 = &j;

        *ptr1 = *ptr2;
        printf("%d %u %u %u\n", i, sizeof(ptr1), sizeof(ptr2), sizeof(*ptr2));
        }
```

Συνοπτικά

Κατηγορία	Συντακτικό	Περιγραφή
Δήλωση 2D πίνακα	`int arr[R][C];`	Δημιουργεί πίνακα με R γραμμές και C στήλες.
Πρόσβαση στοιχείου	`arr[i][j]`	Τιμή στη γραμμή i και στήλη j.
Εισαγωγή τιμών	`scanf("%d", &arr[i][j]);`	Διάβασμα τιμής σε συγκεκριμένη θέση.
Δήλωση δείκτη	`int *ptr;`	Μεταβλητή που αποθηκεύει διεύθυνση μνήμης.
Ανάθεση διεύθυνσης	`ptr = &x;`	Ο δείκτης δείχνει στη διεύθυνση της μεταβλητής x.
Αποαναφορά	`*ptr`	Πρόσβαση / αλλαγή στην τιμή μεταβλητής που δείχνει ο δείκτης.
Διεύθυνση μεταβλητής	`&x`	Παίρνουμε τη διεύθυνση μνήμης του x.
Μέγεθος pointer	`sizeof(ptr)`	Το μέγεθος ενός δείκτη (ίδιο για όλους τους τύπους).
Pointer σε πίνακα	`ptr = arr;`	Ο ptr δείχνει στο πρώτο στοιχείο του πίνακα.
Μετακίνηση δείκτη	`ptr++, ptr+=2`	Δείχνει στο επόμενο/μεθεπόμενο στοιχείο του πίνακα.
Πρόσβαση με arithmetic	`*(arr + i)`	Ισοδύναμο με `arr[i]`.
Εισαγωγή μέσω pointer	`scanf("%f", ptr);`	Αποθήκευση τιμής στη θέση όπου δείχνει ο δείκτης.

Άσκηση 8.2

8.2) Να γραφεί πρόγραμμα σε C που χρησιμοποιεί δείκτη για να τροποποιήσει στοιχεία ενός πίνακα. Δίνεται ο πίνακας **arr = {10, 20, 30, 40, 50}.** Το πρόγραμμα πρέπει:

i. να θέσει τον δείκτη ptr να δείχνει στο πρώτο στοιχείο και να το αλλάξει σε 3,
ii. να μετακινήσει τον δείκτη δύο θέσεις δεξιά και να αλλάξει το τρίτο στοιχείο σε 5,
iii. να εμφανίσει το άθροισμα arr[0] + arr[2].

Στόχος

Ο στόχος είναι η εξάσκηση στην αριθμητική δεικτών και στην πρόσβαση στοιχείων πίνακα μέσω pointers.

Ενδεικτική Λύση 8.2

```
#include <stdio.h>
int main()
{
	int *ptr, arr[] = {10, 20, 30, 40, 50};

	ptr = arr;
	*ptr = 3;

	ptr += 2;
	*ptr = 5;

	printf("%d\n", arr[0]+arr[2]);
	}
```

Άσκηση 8.3

8.4) Να γραφεί πρόγραμμα σε C που:

- διαβάζει 31 θερμοκρασίες και τις αποθηκεύει σε πίνακα χρησιμοποιώντας **δείκτη (arr + i),**
- ζητά από τον χρήστη μια τιμή-όριο **tmp**,
- αναζητά - με χρήση δείκτη - την **πρώτη** θερμοκρασία στον πίνακα που είναι **μικρότερη από την tmp**,
  - εάν βρεθεί, εμφανίζει την τιμή και τη μέρα (θέση + 1),
  - διαφορετικά, εμφανίζει μήνυμα ότι καμία θερμοκρασία δεν ήταν μικρότερη από την τιμή-όριο.

Στόχος:
Εξάσκηση σε pointer arithmetic και πρόσβαση σε πίνακα μέσω δεικτών.

Ενδεικτική Λύση 8.3

```
#include <stdio.h>
#define SIZE 31

int main()
{
        int i;
        double tmp, arr[SIZE];

        for(i = 0; i < SIZE; i++)
        {
                printf("Enter temperatures: ");
                scanf("%lf", arr+i);          /* Το arr+i είναι ισοδύναμο με &arr[i]. */
        }
        printf("Enter temperature to check: ");
        scanf("%lf", &tmp);
        for(i = 0; i < SIZE; i++)
        {               if(*(arr+i) < tmp)
                        break;
        }
        if(i == SIZE)
                printf("No temperature less than %.1f\n", tmp);
        else
                printf("The first temperature less than %.1f was %.1f in day %d\n", tmp, *(arr+i), i+1);
        }
```

Άσκηση 8.4

8.4) Να γραφεί πρόγραμμα που γεμίζει έναν πίνακα 50 βαθμών χρησιμοποιώντας δείκτη και στη συνέχεια εμφανίζει τους βαθμούς σε **αντίστροφη σειρά**, μετακινώντας τον δείκτη από το τέλος προς την αρχή.

Βήματα:

- Μέσω βρόχου while, να διαβάζει βαθμούς, αποθηκεύοντάς τους στις θέσεις του πίνακα όπου δείχνει ο δείκτης**: scanf("%f", ptr);** και να προχωρά στη επόμενη θέση με ptr++ μέχρι το τέλος του πίνακα.

- Μετά την ολοκλήρωση της εισαγωγής, να μετακινεί τον δείκτη μία θέση πίσω ώστε να δείχνει στο τελευταίο έγκυρο στοιχείο: **ptr--;**

- Να εμφανίζει τους βαθμούς σε αντίστροφη σειρά, ξεκινώντας από το τέλος προς την αρχή του πίνακα, χρησιμοποιώντας ξανά **ptr--** σε δεύτερο βρόχο while.

Στόχος:
Δαχείριση πινάκων αποκλειστικά μέσω δεικτών, αριθμητική δεικτών (μετακίνηση δεικτών μέσα στον πίνακα), ανάγνωσης και εκτύπωσης δεδομένων χωρίς χρήση συμβατικής δεικτοδότησης (arr[i]).

Ενδεικτική Λύση 8.4

```
#include <stdio.h> /

#define SIZE 50

int main()
{
	float *ptr, arr[SIZE];

	ptr = arr;
	while(ptr < arr+SIZE)
	{
		printf("Enter grade: ");
		scanf("%f", ptr);
		ptr++;
	}
	ptr--;

	while(ptr >= arr)
	{
		printf("%f\n", *ptr);
		ptr--;
	}
}
```