Ελληνική Δημοκρατία
Τεχνολογικό Εκπαιδευτικό
Ίδρυμα Ηπείρου

Προγραμματισμός Ι

Ενότητα 7 : **Πίνακες Ι**

Αλέξανδρος Τζάλλας

Τμήμα Μηχανικών Πληροφορικής Τ.Ε

Προγραμματισμός Ι

Ενότητα 7 : Πίνακες Ι

Αλέξανδρος Τζάλλας
Λέκτορας

Άρτα, 2015

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «**Εκπαίδευση και Δια Βίου Μάθηση**» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο ΤΕΙ Ηπείρου**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σκοποί ενότητας

- Να αναλυθεί η έννοια των σύνθετων δομών δεδομένων
- Να γίνει κατανοητή η χρησιμότητα/συντακτικοί κανόνες χρήσης των πινάκων στην pascal.
- Να γίνει κατανοητή η χρησιμότητα/συντακτικοί κανόνες χρήσης των πινάκων μίας διάστασης.
- Να περιγραφεί αλγοριθμικά ο τρόπος συγχώνευσης πινάκων στην pascal
- Να γίνει κατανοητή η χρησιμότητα/συντακτικοί κανόνες χρήσης των πινάκων δύο διάστασης.
- Να γίνει επίλυση πολλαπλών παραδειγμάτων

Περιεχόμενα ενότητας

- Τι είναι ο πίνακας
- Μονοδιάστατοι Πίνακες
- Παράδειγμα
- Συγχώνευση Πινάκων
- Δισδιάστατοι Πίνακες
- Παραδείγματα

Εισαγωγή

- Κατά την κατασκευή ενός προγράμματος παρουσιάζεται πολλές φορές η ανάγκη της δημιουργία δομών δεδομένων (data structures) οι οποίες καθιστούν πιο ευέλικτο και πιο λειτουργικό
- Με τη χρήση των δομών δεδομένων γίνεται πιο εύκολη η αποθήκευση, η πρόσβαση, η αναζήτηση και η ομαδοποίηση δεδομένων σε ένα πρόγραμμα
- Οι πιο απλές δομές δεδομένων είναι οι πίνακες

Τι είναι ο πίνακας$_{1/4}$

- Ο πίνακας είναι μία δομή δεδομένων που αποτελείται από στοιχεία του ίδιου τύπου
- Πρακτικά μοιάζει με έναν κατάλογο ομοειδών στοιχείων (π.χ. ακεραίων)
- Οι πίνακες διακρίνονται σε μονοδιάστατούς και πολυδιάστατούς

Τι είναι ο πίνακας$_{2/4}$

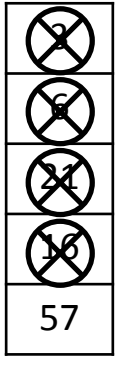
- Οι πίνακες χρειάζονται κυρίως σε προγράμματα με τα οποία βρίσκουμε κάποια στατιστικά στοιχεία (π.χ. μέσο όρο, ελάχιστο, μέγιστο) καθώς και όταν σε ένα πρόγραμμα έχουμε πολλές μεταβλητές του ίδιον τύπου
- Έτσι, όταν έχουμε ένα πρόγραμμα το οποίο περιέχει είκοσι ακέραιες μεταβλητές είναι ευκολότερο να ορίσουμε έναν πίνακα είκοσι ακεραίων από το να ορίσουμε είκοσι μεμονωμένες ακέραιες μεταβλητές

Τι είναι ο πίνακας$_{3/4}$

- Τους πίνακες τους χρησιμοποιούμε πολλές φορές και για **άλλες εφαρμογές** όπως για **ταξινόμηση αριθμών ή επεξεργασία αρχείων**
- Το κυριότερο χαρακτηριστικό που κάνει τους πίνακες να ξεχωρίζουν από τους υπόλοιπους τύπους δεδομένων είναι ότι μπορούμε να εισάγουμε πολλές τιμές στο ίδιο όνομα μεταβλητής και παράλληλα να αποθηκεύεται στη μνήμη τον υπολογιστή η θέση της κάβε τιμής
- Αυτό έχει ως αποτέλεσμα ότι μπορούμε να αναφερόμαστε οποτεδήποτε θελήσουμε σε κάποια από τις τιμές που έχει δώσει ο χρήστης προηγουμένως, σε αντίθεση με τις μεταβλητές.

Τι είναι ο πίνακας$_{4/4}$

- Αν υποθέσουμε ότι η **a** είναι **μία ακέραια μεταβλητή** στην οποία ο χρήστης εισάγει διαδοχικά τις τιμές **3, 6, 21, 16, 57** στη μνήμη του υπολογιστή έχουμε την εξής κατάσταση:



- Δηλαδή κάθε φορά που δίνουμε μία τιμή για τη μεταβλητή α, η προηγούμενη διαγράφεται

Μονοδιάστατοι Πίνακες$_{1/3}$

- Ένας μονοδιάστατος πίνακας είναι μία δομή με την οποία μπορούμε να καταχωρήσουμε σε μια μεταβλητή απεριόριστα στοιχεία του ίδιον τύπου και όχι μόνο ένα όπως έχουμε δει μέχρι τώρα
- Η θέση κάθε στοιχείου του πίνακα προσδιορίζεται μονοσήμαντα από έναν αριθμό
- Μπορούμε να φανταστούμε έναν πίνακα σαν μία παράταξη η οποία αποτελείται από μεταβλητές του ίδιον τύπον (π.χ. integer) καταχωρημένες σε διαφορετικά κελιά μεταξύ τους

Μονοδιάστατοι Πίνακες$_{2/3}$

- Αν υποθέσουμε ότι έχουμε ορίσει και αρχικοποιήσει έναν πίνακα ακεραίων με όνομα **a**, ο οποίος περιέχει **πέντε στοιχεία (π.χ. 3, 6, 21, 16, 57)**, στη μνήμη τον υπολογιστή Θα έχουμε την εξής κατάσταση:

a[1]	a[2]	a[3]	a[4]	a[5]
3	6	21	16	57

- Εδώ το 3 είναι το πρώτο στοιχείο τον πίνακα (a[1]), το 6 το δεύτερο (a[2] το 21 το τρίτο (a[3]), το 16 το τέταρτο (a[4]) και το 57 το πέμπτο (a[5])

Μονοδιάστατοι Πίνακες$_{3/3}$

- Γενικά ένας μονοδιάστατος πίνακας ορίζεται ως εξής:

var όνομα_πίνακα : **array** [a..b] **of** τύπος_στοιχείων;

- Ενώ το i-στοιχείο τον πίνακα αρχικοποιείται ως εξής:

όνομα_πίνακα [i] := τιμή;

(το α[1] είναι το πρώτο στοιχείο του πίνακα, το α[2] είναι το δεύτερο κ.ο.κ.)

- **Τον προηγούμενο πίνακα** θα τον ορίζαμε στις δηλώσεις με την εντολή:

var a: **array** [1..5] **of** integer;

- Ενώ θα τον αρχικοποιούσαμε στο κυρίως πρόγραμμα με τις εντολές:

α[1]:= 3; α[2]:= 6; α[3]:= 21; α[4]:= 16; α[5]:= 57;

Παρατηρήσεις$_{1/2}$

- Όταν δηλώνουμε έναν πίνακα δεν είναι υποχρεωτικό να αρχικοποιήσουμε όλα τα στοιχεία του

  Μπορούμε π.χ. να δηλώσουμε έναν πίνακα πέντε στοιχείων και να αρχικοποιήσουμε μόνο το τρίτο και το τέταρτο δηλαδή να είχαμε στο κυρίως πρόγραμμα τις εντολές

  a[3]:=21; a[5]:=57; και όχι τα υπόλοιπα

- Τα στοιχεία ενός πίνακα μπορεί να είναι ακέραιοι, πραγματικοί, χαρακτήρες ή λογικές μεταβλητές

Παρατηρήσεις$_{2/2}$

- Η επεξεργασία ενός πίνακα πρέπει να αφορά κάθε στοιχείο του ξεχωριστά και όχι ολόκληρο τον πίνακα

π.χ. αν Θέλουμε να **αρχικοποιήσουμε** όλα τα στοιχεία του προηγούμενού πίνακα να είναι ίσα με 4 **δεν** επιτρέπεται η εντολή **α:=4**; αλλά θα γράφαμε:

a[1]:=4; a[2]:=4; a[3]:=4; **a[4]:=4;** a[5]:=4;

ή καλύτερα με την: **fo**r i:=1 **to** 5 **do** a[i]:=4;

ενώ αν θέλαμε να εξισώσουμε όλα τα στοιχεία ενός άλλου πίνακα b πέντε ακεραίων με αυτά τον a θα γράφαμε την εντολή:

for i:=1 **to** 5 **do** b[i]:=a[i]; και όχι b:=a;

Παράδειγμα 1

- Να εκτελεστεί με το χέρι το παρακάτω τμήμα προγράμματος.

i:=1; j:=3; writeln(a[i+j], a[j-i],]); a[2]:=8; a[1+2]:=9; writeln (a[i], a[j]); j:=j+1; a[i]:=a[j]; writeln (a[1], a[4]);

Υποθέτουμε ότι ο a είναι ο εξής μονοδιάστατος πίνακας τεσσάρων ακεραίων στοιχείων:

4	3	7	1

- Να γραφτούν οι εντολές για τη δημιουργία του εξής πίνακα:

5	4	3	2	1

Λύση$_{1/3}$

- Λόγω του δεδομένου αρχικού πίνακα μπορούμε να υποθέσουμε ότι στο πρόγραμμα υπάρχουν οι εντολές:

a[1]:=4; a[2]:=3; a[3]:=7; a[4]:=1;

- στη συνέχεια εκχωρούνται οι τιμές 1 και 3 στις μεταβλητέςi και j αντίστοιχα και εκτυπώνονται στην οθόνη οι τιμές:

a[1+3]= a[4]=1, a[3-1]= a[2]=3

- Τώρα στο δεύτερο στοιχείο τον πίνακα εκχωρείται η τιμή 8 (από 3 πού ήταν πριν) ενώ στο τρίτο **(i+2=1+2=3)** εκχωρείται η τιμή 9 και εκτυπώνονται στην οθόνη οι τιμές **a[1]=4 και a[3]=9**.Τέλος, το j αυξάνεται κατά 1 και γίνεται 4 ενώ το πρώτο στοιχείο του πίνακα (αφού i=1) γίνεται ίσο με 1 που είναι η τιμή του α[4] και στην οθόνη εμφανίζονται οι τιμές α[1]=1 και α[4]=1. Το ζητούμενο tracetable είναι το εξής:

Λύση 2/3

- Το ζητούμενο tracetable είναι το εξής:

i	j	A
1	3	a[1]=4,a[2]=3,a[3]=7,a[4]=1
		a[2]=8,a[3]=9
	2	a[1]=1

Output	
1	3
4	9
1	1

Λύση$_{3/3}$

- Ο ζητούμενος πίνακας είναι ένας πίνακας πέντε ακεραίων επομένως στις δηλώσεις θα έχουμε την εντολή:

var a:**array** [1..5] **of** integer;

και στο κυρίως πρόγραμμα τις εντολές:

a[1]:=5; a[2]:=4; a[3]:=3; a[4]:=2; a[5]:=1;

- Αυτές οι πέντε εντολές θα μπορούσαν να συντομευτούν με την εξής εντολή:

for i:= 5 **do** a[i]:=6-i;

γιατί **a[1]=6-1=5, a[2]=6-2=4 κ.ο.κ.**

Παράδειγμα 2

Να γραφτεί πρόγραμμα το οποίο αρχικά **να διαβάζει** τα στοιχεία **ενός πίνακα a είκοσι πραγματικών αριθμών** και να τους τυπώνει στην οθόνη ανάποδα στη συνέχεια να **διαβάζει τα στοιχεία ενός πίνακα b τριάντα χαρακτήρων** και να τυπώνει στην οθόνη το **πλήθος των εμφανίσεων χαρακτήρα a** και **μετά να διαβάζει τα στοιχεία ενός πίνακα c δεκαπέντε λογικών μεταβλητών** και να τυπώνει στην οθόνη το **πλήθος των true το πλήθος των false**

Λύση$_{1/2}$

- Στο πρόγραμμα αυτό εκτός από τους τρεις πίνακες **a,b,c**
- Πρέπει να δηλώσουμε έναν μετρητή για την επεξεργασία των τριών πινάκων με τη χρήση του **for)**
- Έναν μετρητή **j** για τις εμφανίσεις του a κι έναν μετρητή k για τη μέτρηση των true στον τελευταίο πίνακα
- Δε χρειάζεται ξεχωριστή μεταβλητή για το πλήθος των **false** αφού αυτό ισούται με **15-k**

Λύση 2/2

Όσον αφορά το δεύτερο πίνακα b ελέγχουμε αν κάθε στοιχείο του είναι ίσο με a
και αν είναι τότε αυξάνεται ο μετρητής j κατά 1. Έτσι μόλις τελειώσει αυτός ο
έλεγχος η τιμή του j θα ισούται με το πλήθος των a στον πίνακα b.

Τέλος, στον τελευταίο πίνακα c ελέγχουμε αν κάθε στοιχείο του είναι ίσο με **true**, αν είναι τότε αυξάνεται η τιμή τον k κατά 1 κι έτσι μόλις τελειώσει αυτή η διεργασία η τιμή τον k Θα ισούται με το πλήθος των **true** στον πίνακα c.

```
program epexergasia_pinakon(input,output);
var a: array [ 1..20] of real;
b:                    array [ 1..30] of char;
c:                    array [ 1..15] of boolean;
i, j, k: integer;
begin
j:= 0;
k:= 0;
writeln ('Dwse 20 pragmatikous arithmous: ');
for i:= 1 to 20 do
readln (a[i]);
for i:= 1 to 20 do
writeln ( a[21-i]);
writeln ('Dwse 30 Charakthres: ');
for i:= 1 to 30 do
readln (b[i] );
for i:= 1 to 30 do
if b[i]= 'a' then
j:=j+1;
writeln (' To plhthos twn a ston pinaka b eina: ', j);
writeln (' Dwse 15 logikes metablhtes: ');
for i:=1 to 15 do
readln(c[i]);
for i:=1 to 15 do
if c[i]=true then k:=k+1;
writeln(' To plhthos twn true einai: ', k,' kai to plhthos twn false einai: ', 15-k);
end.
```

Παράδειγμα 3

Να γραφτεί πρόγραμμα το οποίο να δέχεται ως είσοδο του βαθμούς δεκαπέντε μαθημάτων ενός μαθητή και να εμφανίζει το μέσο όρο τους και τον μέγιστο βαθμό

- Εδώ η εύρεση του **μ.ο** μπορεί να γίνει με τη διαδικασία που είχαμε αναπτύξει στο κεφάλαιο της δομής επανάληψης δηλαδή θεωρώντας μεταβλητή **sum** στην οποία προσθέτουμε κάθε φορά τον επόμενο βαθμό στη συνέχεια διαιρώντας την τελική τιμή τον **sum με 15**.
- Ακολουθώντας τον τρόπο όμως είναι πολύ δύσκολο να βρούμε το μέγιστο στοιχείο δεκαπέντε μαθημάτων γιατί στη μεταβλητή **x** κάθε φορά αποθηκεύει τελευταίος βαθμός που δίνει ο χρήστης κι έτσι δεν είναι πρακτικός ο διαδοχικός έλεγχος όλων των βαθμών με το γνωστό αλγόριθμο για την **εύρεση** μέγιστου
- Αυτό το πρόβλημα μπορεί να λυθεί είτε χρησιμοποιώντας δεκαπέντε διαφορετικές μεταβλητές για την αποθήκευση των δεκαπέντε βαθμών **είτε χρησιμοποιώντας έναν πίνακα δεκαπέντε στοιχείων που είναι πολύ πιο εύκολο**

Λύση

```
program grades(input,output);
var a: array [ 1..15] of real;
i: integer;
average, sum, max: real;
begin
sum:= 0.0;
writeln(' Dwse 15 bathmous twn mathimatwn: ');
for i:= 1 to 15 do
begin
readln(a[i]);
sum:= sum+ a[i];
end;
average:= sum/15.0;
max:= a[1];
for i:=2 to 15 do
if a[i] > max then  max:= a[i];
writeln ('O m.o twn mathimatwn einai: ',average, ' o megistos
einai: ',max)
end.
```

Παρατηρήσεις$_{1/2}$

- Η εισαγωγή των βαθμών του μαθητή και η εύρεση τον αθροίσματός τους είναι δύο διεργασίες που Θα μπορούσαν να υλοποιη0ούν με δύο ξεχωριστά **for:**

for i:=1 **to** 15 **do** readln (s[i]); και στη συνέχεια

for i:=1 **to** 15 **do sum:=** sum+ a[i];

- Η εύρεση τον μέγιστου βαθμού γίνεται με το γνωστό τρόπο με τη μόνη διαφορά ότι εδώ έχουμε τους αριθμούς απο0ηκευμένους σε έναν πίνακα λόγω του μεγάλου πλήθους τους
- Έτσι, υποθέτουμε ότι το πρώτο στοιχείο τον πίνακα είναι μέγιστο, αν βρούμε κάποιο άλλο στοιχείο του πίνακα (π.χ. το τρίτο) ότι είναι μεγαλύτερο τον πρώτου τότε υποθέτουμε ότι το τρίτο είναι μέγιστο κ.ο.κ. ώσπου να ολοκληρωθεί αυτός ο έλεγχος και για τους δεκαπέντε βαθμούς
- Στο τέλος θα έχουμε βρει το μέγιστο των βαθμών

Παρατηρήσεις$_{2/2}$

- Στο δεύτερο **for** του προγράμματος ο μετρητής i παίρνει τιμές από το 2 και όχι από το 1 γιατί τότε θα ήταν σα να ελέγχαμε αν **α[1] > α[1]** που δεν ισχύει ποτέ (δηλαδή αποφεύγουμε έναν περιττό έλεγχο)
- Για αυτό το λόγο πάντοτε όταν έχουμε προβλήματα με μέγιστο ή ελάχιστο ο μετρητής του **for** πρέπει να ξεκινάει από το 2 και όχι από το 1

Συγχώνευση Πινάκων$_{1/10}$

- Η συγχώνευση δύο ταξινομημένων πινάκων έχει ως στόχο τη δημιουργία ενός τρίτου επίσης ταξινομημένου πίνακα ο οποίος θα περιέχει όλα τα στοιχεία των προηγούμενων πινάκων
- Για να συγχωνεύσουμε δύο ταξινομημένους πίνακες **a, b** αρκεί να δημιουργήσουμε έναν τρίτο πίνακα **c** ο οποίος θα περιέχει όλα τα στοιχεία των αρχικών πινάκων και θα είναι και αυτός ταξινομημένος

Συγχώνευση Πινάκων 2/10

- Η διαδικασία (αλγόριθμος) που ακολουθούμε είναι η εξής:
  - Στην αρχή συγκρίνουμε τα πρώτα στοιχεία των **δύο πινάκων a, b** και το μικρότερο από αυτά το εκχωρούμε στην πρώτη θέση του **πίνακα c**, μετά συγκρίνουμε το επόμενο στοιχείο από αυτό που εκχωρήσαμε στον **πίνακα c** με το πρώτο του άλλου πίνακα και αυτό που είναι μικρότερο από αυτά τα δύο το τοποθετούμε στη δεύτερη θέση του πίνακα **c**
  - Συνεχίζοντας με αυτόν τον τρόπο συγκρίνουμε κάθε φορά το επόμενο στοιχείο του πίνακα a ή b που εκχωρούμε στον πίνακα c με το στοιχείο **του** άλλον πίνακα στο οποίο βρισκόμαστε κάθε φορά και καταχωρούμε το μικρότερο από αυτά στην αμέσως επόμενη θέση στον πίνακα c
  - Αυτή η διαδικασία επαναλαμβάνεται μέχρι να αντιγραφούν όλα τα στοιχεία ενός εκ των δύο πινάκων α, b στον πίνακα c
  - Μετά απλώς αντιγράφονται τα υπόλοιπα στοιχεία του άλλου πίνακα στον c
  - Έτσι ο c θα περιέχει όλα τα στοιχεία των πινάκων α, b σε αύξουσα σειρά

Συγχώνευση Πινάκων 3/10

- Ας υποθέσουμε έχουμε τους εξής δυο πίνακες **a, b** τους οποίους θέλουμε να συγχωνεύσουμε:

1	4	5	9	11	13	16

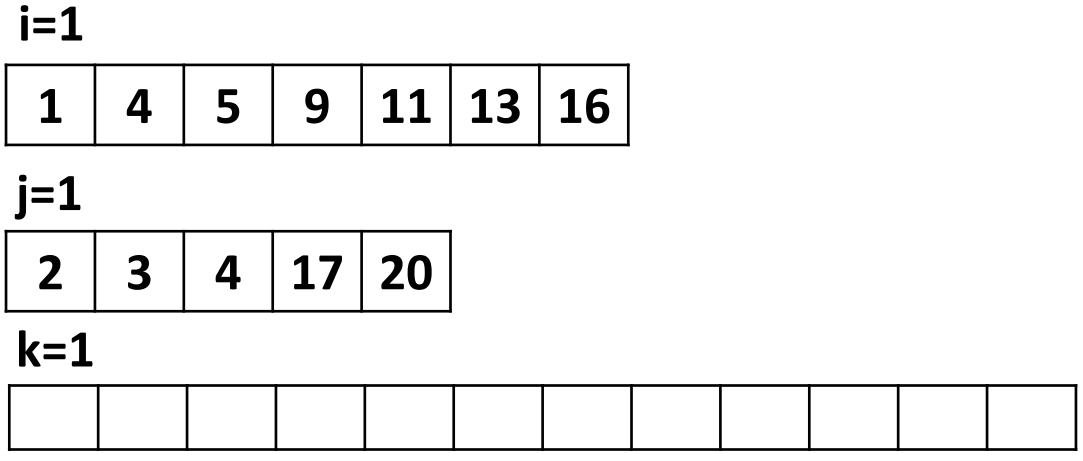
2	3	4	17	20

- Οι, πίνακες αυτοί έχουν μήκη **m=7, n=5,** ενώ ο καινούργιος πίνακας **c** Θα έχει μήκος **m+n=12**

- Θα χρησιμοποιήσουμε τρεις δείκτες **i, j, k**, έναν για κάθε πίνακα, και θα τους αρχικοποιήσουμε ίσους με 1 γιατί οι μεταβλητές i, j δείχνουν κάθε φορά το πρώτο από τα στοιχεία των πινάκων **α, b** αντίστοιχα που δεν έχουν μεταφερθεί στον πίνακα **c** ενώ η μεταβλητή **k** δείχνει κάθε φορά στην πρώτη κενή Θέση τον πίνακα **c** στην οποία πρόκειται να εισαχθεί το επόμενο στοιχείο

Συγχώνευση Πινάκων 4/10

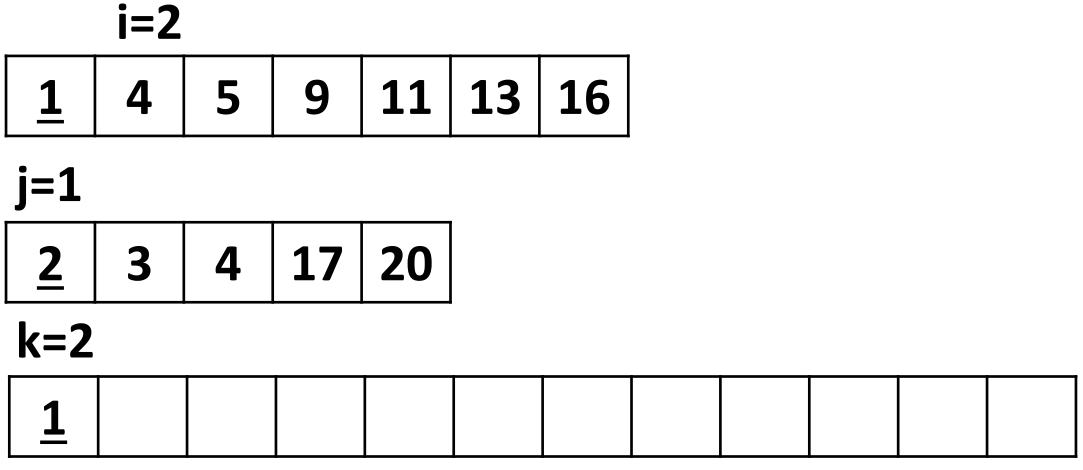
- **Πρώτο βήμα**



Αρχικά όπως είπαμε εντοπίζουμε το μικρότερο από τα στοιχεία που βρίσκονται στην πρώτη Θέση των a και b που εδώ είναι το 1 (1<2) και μεταφέρουμε στην πρώτη Θέση τον πίνακα c

Συγχώνευση Πινάκων 5/10

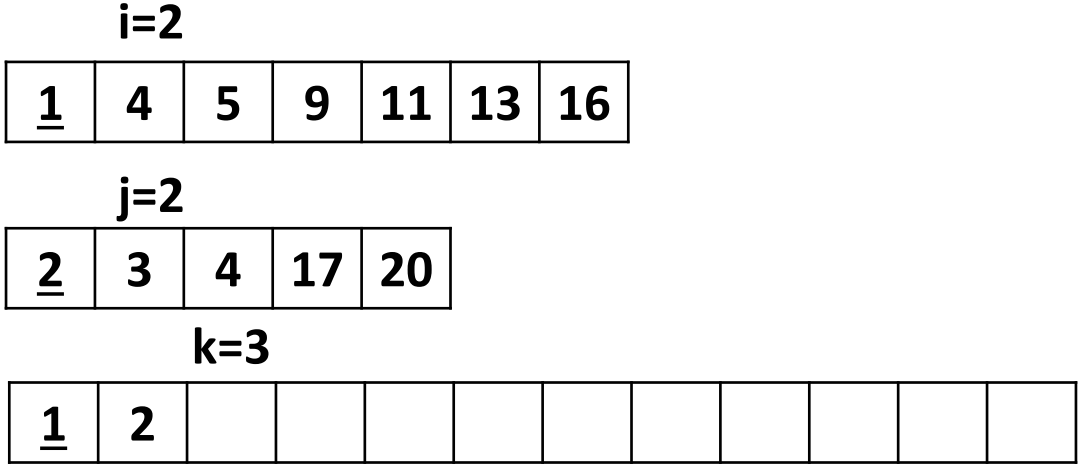
- **Πρώτο βήμα**



Την ίδια στιγμή ο δείκτης k του πίνακα c αυξάνεται κατά 1 για να δείχνει τη θέση στην οποία θα καταχωρηθεί το επόμενο στοιχείο και ο δείκτης i του πίνακα a αυξάνεται επίσης κατά 1 ώστε να δείχνει το πρώτο από τα στοιχεία του που δεν έχουν μεταφερθεί ακόμη στον πίνακα c

Συγχώνευση Πινάκων 6/10

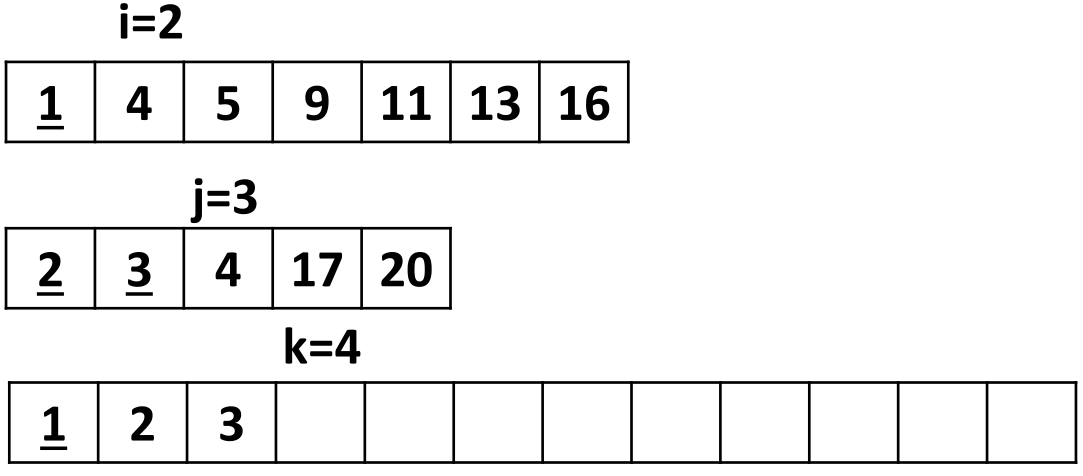
- **Δεύτερο βήμα**



Στο επόμενο βήμα εντοπίζουμε το μικρότερο από τα στοιχεία στη δεύτερη θέση του πίνακα a και στην πρώτη θέση του πίνακα b που είναι το 2 (2<4) και το καταχωρούμε στον πίνακα c. Ομοίως με πριν τα j, k αυξάνουν κατά 1

Συγχώνευση Πινάκων 7/10

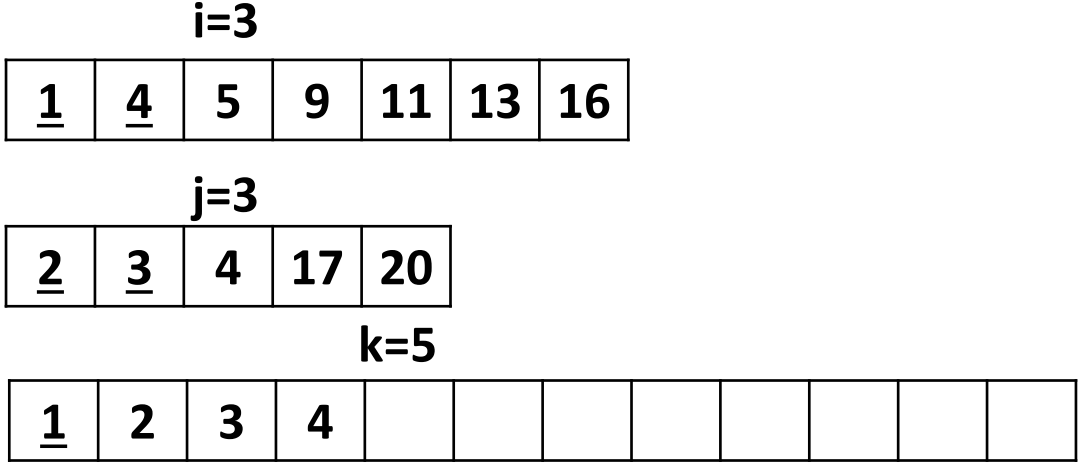
- **Τρίτο βήμα**



Ακριβώς η ίδια διαδικασία επαναλαμβάνεται ώσπου όλα τα στοιχεία τον πίνακα a ή τον πίνακα b να καταχωρηθούν στον πίνακα c (εδώ 3<4)

Συγχώνευση Πινάκων 8/10

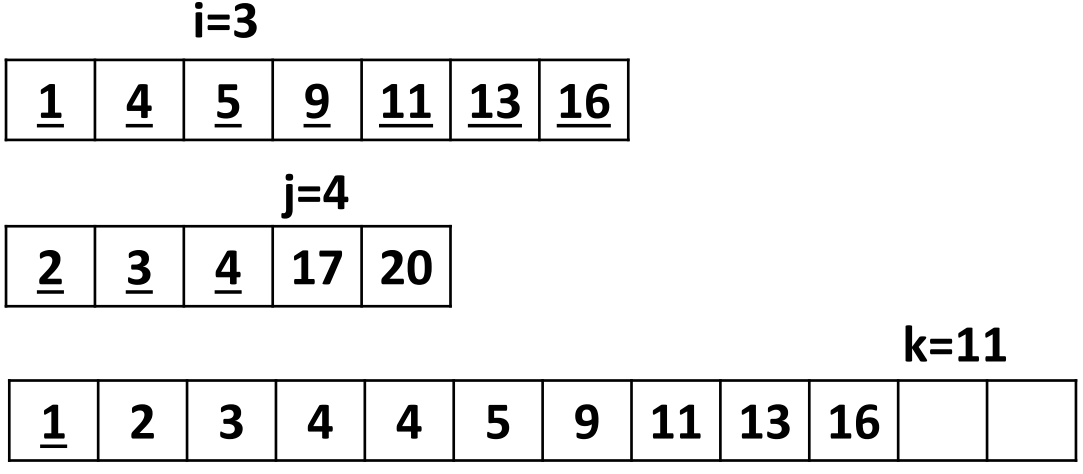
- **Τέταρτο βήμα**



Εδώ επειδή συγκρίνουμε το δεύτερο στοιχείο του πίνακα a που είναι το 4 και το τρίτο στοιχείο του πίνακα b που είναι το 4 καταχωρούμε στον πίνακα c όποιο θέλουμε π.χ. το πρώτο με τους τρεις δείκτες i, j, k να μεταβάλλονται ανάλογα

Συγχώνευση Πινάκων 8/10

- **Δέκατο βήμα**



Σε αυτό το σημείο όλα τα στοιχεία του πίνακα a έχουν μεταφερθεί στον πίνακα c οπότε για να μην κάνουμε άσκοπες συγκρίσεις μεταφέρουμε απ' ευθείας τα υπόλοιπα στοιχεία του πίνακα b στον πίνακα c

Συγχώνευση Πινάκων 9/10

- **Ενδέκατο βήμα**

1	4	5	9	11	13	16

2	3	4	17	20

1	2	3	4	4	5	9	11	13	16	17	20

Δισδιάστατοι Πίνακες$_{1/4}$

- Ένας δισδιάστατος πίνακας είναι μία δομή δεδομένων με την οποία μπορούμε να καταχωρήσουμε σε μία μεταβλητή απεριόριστα στοιχεία του ίδιον τύπου
- Η Θέση κάθε στοιχείου στον πίνακα προσδιορίζεται μονοσήμαντα από ένα ζεύγος αριθμών
- Μπορούμε, σε αντίθεση με τον μονοδιάστατο πίνακα, να φανταστούμε το δισδιάστατο πίνακα σα μία παράταξη διατεταγμένη σε δύο διαστάσεις η οποία αποτελείται από μεταβλητές τον ίδιον τύπον καταχωρημένες σε διαφορετικά κελιά μεταξύ τους

Δισδιάστατοι Πίνακες$_{2/4}$

- Αν υποθέσουμε ότι έχουμε ορίσει έναν δισδιάστατο πίνακα χαρακτήρων με δύο γραμμές και τρεις στήλες που έχει όνομα a και τιμές τις `h' , '1', '1', '4', `s' , ` *' , στη μνήμη τον υπολογιστή Θα έχουμε την εξής κατάσταση:

a[1,1]	a[1,2]	a[1,3]
‘h’	**‘i’**	**‘/’**
‘4’	**‘s’**	**‘*’**
a[1,1]	**a[1,2]**	**a[1,3]**

Εδώ ισχύει: a[1,1]='h', a[1,2]='i', a[1,3]='/’, a[2,1]='4', a[2,2]='s'

Δισδιάστατοι Πίνακες$_{3/4}$

- Γενικά ένας δισδιάστατος πίνακας ορίζεται ως εξής:

  var όνομα_πίνακα: **array [α..b, c..d] of** τύπος_δεδομένων;

- Ενώ το (i, j) στοιχείο τον πίνακα ορίζεται ως εξής:

  όνομα_πίνακα [i,j]:= τιμή

  (π.χ. το α[1,1] είναι το στοιχείο στην πρώτη γραμμή και την πρώτη στήλη, το α[1,2] είναι το στοιχείο στην πρώτη γραμμή και τη δεύτερη στήλη κ.ο.κ.).

- Τον προηγούμενο πίνακα Θα τον ορίζαμε στις δηλώσεις με την εντολή:

a[1,1]	**a[1,2]**	**a[1,3]**
‘h’	**‘i’**	**‘/’**
‘4’	**‘s’**	**‘*’**
a[1,1]	**a[1,2]**	**a[1,3]**

Δισδιάστατοι Πίνακες$_{4/4}$

- Τον προηγούμενο πίνακα Θα τον ορίζαμε στις δηλώσεις με την εντολή:

a[1,1]	**a[1,2]**	**a[1,3]**
‘h’	**‘i’**	**‘/’**
‘4’	**‘s’**	**‘*’**
a[1,1]	**a[1,2]**	**a[1,3]**

var a: **array** [1..2,1..3] **of** char;

- Ενώ θα αρχικοποιούσαμε τα στοιχεία τον στο κυρίως πρόγραμμα ως εξής: **a[1,1]:='h'; a[1,2]:='i'; a[1,3]:='/'; a[2,1]:='4'; a[2,2]:='s'; a[2,3]:=‘*';**
- Ως προς την επεξεργασία των στοιχείων του δισδιάστατου πίνακα ισχύουν ακριβώς οι ίδιες παρατηρήσεις με το μονοδιάστατο.

Παράδειγμα 3

i. Να εκτελεστεί το παρακάτω τμήμα προγράμματος. Υποθέτουμε ότι ο α είναι ο εξής δισδιάστατος πίνακας:

3	**4**	**-6**	**9**
8	**7**	**-5**	**2**

ii. i:=2; j:=2; write (a[i,j]); ί:= — 1; j := + 1;

iii. write (a[i+ 1,j]); write (a[a[2,4],a[1,2]]);

ii. Να γραφτούν οι εντολές για τη δημιουργία τον εξής πίνακα:

1	**1**	**1**
2	**2**	**2**
5	**5**	**5**

Λύση 1/3

i) i:=2; j:=2; write (a[i,j]); í:= - 1; j := + 1; write (a[i+ 1,j]); write (a[a[2,4],a[1,2]]);

- Πρώτα αρχικοποιούμε τις μεταβλητές **i και j** να είναι ίσες **με 2** και στην οθόνη εκτυπώνεται το στοιχείο της δεύτερης γραμμής και της δεύτερης στήλης τον πίνακα που είναι **ίσο με 7**
- Στη συνέχεια, το i μειώνεται **κατά 1** και **γίνεται 1**, το **j αυξάνεται κατά i=1** και γίνεται **ίσο με 3** ενώ στην οθόνη εκτυπώνεται το στοιχείο της δεύτερης γραμμής και της τρίτης στήλης που είναι **ίσο με -5**.
- Τέλος, στην οθόνη εμφανίζεται το στοιχείο **a[a[2,4], a[1,2]]**
- Το **a[2,4]** ισούται **με 2** ενώ το **a[1,2]** ισούται **με 4** κι έτσι για να βρούμε το **a[a[2,4], a[1,2]]** αρκεί να βρούμε το **a[2,4]** που ισούται **με 2**

Output: 7 -5 2

Λύση$_{2/3}$

i) **i:=2; j:=2; write (a[i,j]); ί:= - 1; j := + 1; write (a[i+ 1,j]); write (a[a[2,4],a[1,2]]);**

i	j	a
2	2	a[2,2]=7
1 (=2-1)	3 (=2+1)	a[2,3]=-5
		a[2,4]=2

Λύση$_{3/3}$

ii)

- Ο ζητούμενος πίνακας έχει τρεις γραμμές και τρείς στήλες άρα στις δηλώσεις θα έχουμε την εντολή:

```
var b:array[1..3,1..3] of integer;
```

- Ενώ στο κυρίως πρόγραμμα θα έχουμε τις εντολές: b[1,1]:=1; b[1,2]:=1 b[1,3]:=1 ;b[2,1]:=2; b[2,2]:=2 ;b[2,3]:=2; b[3,1]:=5; b[3,2]:=5;b[3,3]:=2; ή πιο απλά:

```
for i:=1 to 3 do b[1,i]:=1;

for i:=1 to 3 do b[2,i]:=2;

for i:=1 to 3 do b[3,i]:=5;
```

Παράδειγμα 4

i. Να εκτελεστεί το παρακάτω τμήμα προγράμματος:

```
for i:=1 to 3 do
    for j:=1 to 3 do
    a[i,j]:=a[a[i,j]], a[j,i]]; Όπου αρχικά ο a είναι ο πίνακας;
```

2	**1**	**3**
3	**3**	**1**
1	**2**	**1**

Λύση

i) Με αυτές τις εντολές τροποποιούμε τα στοιχεία του πίνακα a. Οι μεταβλητές i,j παίρνουν διαδοχικά τις τιμές 1,2,3 όπως φαίνεται αναλυτικά στο tracetable που ακολουθεί:

i	j	a[i,j]
1	1	a[1,1]=a[a[1,1],a[1,1]]=a[2,2]=3
	2	a[1,2]=a[a[1,2],a[2,1]]=a[1,3]=3
	3	a[1,3]=a[a[1,3],a[3,1]]=a[3,1]=1
2	1	a[2,1]=a[a[2,1],a[1,2]]=a[3,1]=1
	2	a[2,2]=a[a[2,2],a[2,2]]=a[3,3]=1
	3	a[2,3]=a[a[2,3],a[3,2]]=a[1,2]=1
3	1	a[3,1]=a[a[3,1],a[1,3]]=a[1,3]=3
	2	a[3,2]=a[a[3,2],a[2,3]]=a[2,1]=3
	3	a[3,3]=a[a[3,3],a[3,3]]=a[1,1]=2

Βιβλιογραφία

Βλαχάβας Ι. (1994). Η γλώσσα προγραμματισμού Pascal. Εκδόσεις Γαρταγάνης Διονύσιος.

Κάβουρας Ι.Κ. (1999). Δομημένος Προγραμματισμός με Pascal. Εκδόσεις Κλειδάριθμος.

Αλεβίζου Θ., & Καμπουρέλης Α. (1995). Μαθήματα Προγραμματισμού: Εισαγωγή με τη Γλώσσα Pascal. Εκδόσεις Παπασωτηρίου.

Cooper D. (1993). Oh! Pascal!, An Introduction to Computing, του. Εκδόσεις Norton.

Larry R.N. (1998). Advanced Programming in Pascal with Data Structures. Εκδόσεις Macmillan USA.

Τσελίκης Γ.Σ., Τσελίκας Ν.Δ. (2012). C: από τη Θεωρία στην Εφαρμογή (Β' Έκδοση). Εκδόσεις Παπασωτηρίου.

Aho A.V., Hopcroft J.E., & Ullman J.D. (1974). The design and analysis of computer algorithms. Εκδόσεις Addison Wesley.

Abelson H., Sussman G.J., Sussman J. (1985). Structure and Interpretation of Computer Programs, MIT Press, McGraw Hill Book Company.

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

[1] http://creativecommons.org/licenses/by-nc-nd/4.0/deed.el

Τέλος Ενότητας

Επεξεργασία: Ευάγγελος Καρβούνης

Άρτα, 2015

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο
ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επενδύοντας στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ
ΕΣΠΑ 2007-2013
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Τέλος Ενότητας

Πίνακες Ι