Κεφάλαιο 4 – Τελεστές

Ο τελεστής =

Ο τελεστής `=` χρησιμοποιείται για την απόδοση τιμής σε μία μεταβλητή. Π.χ. με την εντολή `a = 10;` η τιμή της μεταβλητής `a` γίνεται `10`, ενώ με την εντολή `a = k;` η τιμή της μεταβλητής `a` γίνεται ίση με την τιμή τη μεταβλητής `k`.

Αν χρησιμοποιείται πολλές φορές σε μία εντολή εκχώρησης, τότε η τελική τιμή εκχώρησης αποθηκεύεται σε όλες τις μεταβλητές. Π.χ. με την εντολή:

```
int a,b,c;
a = b = c = 10;
```

οι τιμές των μεταβλητών `a`, `b` και `c` γίνονται ίσες με `10`.

Οι μαθηματικοί τελεστές

Οι μαθηματικοί τελεστές **`+`**, **`-`**, **`*`**, **`/`** χρησιμοποιούνται για την εκτέλεση των γνωστών μαθηματικών πράξεων. Ο τελεστής **`%`** χρησιμοποιείται για να βρούμε το **υπόλοιπο** της διαίρεσης δύο **ακεραίων** αριθμών.

Παρατηρήσεις

1) Ο τελεστής **`%`** εφαρμόζεται **μόνο** σε ακεραίους αριθμούς.

2) Ο μεταγλωττιστής εκτελεί **πιο γρήγορα** αριθμητικές πράξεις που περιέχουν τους τελεστές **`+`** και **`-`** από ότι τους τελεστές **`*`** και **`/`**. Π.χ. η πράξη 4+4 εκτελείται πιο γρήγορα από την πράξη 2*4.

Ο τελεστής ++

Ο τελεστής αύξησης **`++`** μπαίνει **πριν ή μετά** από το όνομα μίας μεταβλητής. Σε κάθε περίπτωση η τιμή της μεταβλητής **αυξάνεται** κατά ένα. Το επόμενο πρόγραμμα εμφανίζει `Num = 5`, γιατί η εντολή `a++` είναι ισοδύναμη με:
`a = a+1 = 4+1 = 5`.

```
#include <stdio.h>
int main()
{
      int a = 4;
      a++; /* Ισοδύναμο με a = a+1; */
      printf("Num = %d\n",a);
      return 0;
}
```

Όταν ο τελεστής **++** χρησιμοποιηθεί σε μία εντολή εκχώρησης **μετά** από το όνομα της μεταβλητής, τότε **πρώτα** χρησιμοποιείται η τρέχουσα τιμή της μεταβλητής και **μετά** αυτή αυξάνεται κατά ένα. Στο επόμενο πρόγραμμα με την εντολή `b = a++;` πρώτα αποθηκεύεται στη μεταβλητή `b` η τρέχουσα τιμή της μεταβλητής `a` και μετά αυξάνεται η τιμή της `a` κατά 1.

```
#include <stdio.h>
int main()
{
      int a,b;

      a = 4;
      b = a++;
      printf("a = %d b = %d\n",a,b);
      return 0;
}
```

Επομένως, το πρόγραμμα θα εμφανίσει `a = 5` και `b = 4`.

Αντίθετα, όταν ο τελεστής **++** χρησιμοποιηθεί σε μία εντολή εκχώρησης **πριν** από το όνομα της μεταβλητής, τότε **πρώτα** αυξάνεται η τιμή της μεταβλητής κατά ένα και **μετά** αυτή χρησιμοποιείται. Στο επόμενο πρόγραμμα με την εντολή `b = ++a;` πρώτα αυξάνεται η τιμή της `a` κατά 1 και μετά αυτή αποθηκεύεται στη μεταβλητή `b`.

```
#include <stdio.h>
int main()
{
      int a,b;

      a = 4;
      b = ++a;
      printf("a = %d b = %d\n",a,b);
      return 0;
}
```

Επομένως, το πρόγραμμα θα εμφανίσει `a = 5` και `b = 5`.

Ο τελεστής --

Ο τελεστής μείωσης **--** όπως και ο τελεστής αύξησης **++** μπαίνει **πριν ή μετά** από το όνομα μίας μεταβλητής. Σε κάθε περίπτωση, η τιμή της μεταβλητής **μειώνεται** κατά ένα. Το επόμενο πρόγραμμα εμφανίζει `Num = 3`, γιατί η εντολή `a--` είναι ισοδύναμη με `a = a-1 = 4-1 = 3`.

```
#include <stdio.h>
int main()
{
      int a = 4;
      a--; /* Ισοδύναμο με a = a-1; */
      printf("Num = %d\n",a);
      return 0;
}
```

Όταν ο τελεστής **--** χρησιμοποιείται σε μία εντολή εκχώρησης πριν ή μετά το όνομα της μεταβλητής, τότε ισχύουν οι ίδιοι κανόνες που ισχύουν για τον τελεστή **++**.

Κ.4.1 Ποια είναι η έξοδος του παρακάτω προγράμματος ?

```
#include <stdio.h>
int main()
{
      int a,b;

      a = 4;

      b = a--;
      printf("a = %d b = %d\n",a,b);

      b = --a;
      printf("a = %d b = %d\n",a,b);
      return 0;
}
```

Απάντηση: Με την εντολή `b = a--;` πρώτα αποθηκεύεται στη μεταβλητή `b` η τρέχουσα τιμή της `a` και μετά αυτή μειώνεται κατά 1. Επομένως, η πρώτη `printf()` θα εμφανίσει `a = 3` και `b = 4`.

Κεφάλαιο 7 – Πίνακες

Ένας πίνακας στη **C** είναι μία συλλογή δεδομένων που αποτελείται από στοιχεία του **ίδιου τύπου** (π.χ. πίνακας ακεραίων αριθμών, πίνακας πραγματικών αριθμών, πίνακας χαρακτήρων, ...). Όλοι οι πίνακες δεσμεύουν **συνεχόμενες θέσεις** στη μνήμη του υπολογιστή και διακρίνονται σε πίνακες μίας διάστασης και πολλών διαστάσεων. Στη συνέχεια, θα παρουσιάσουμε τη συνηθέστερη μορφή πινάκων που είναι οι **μονοδιάστατοι** και οι **διδιάστατοι** πίνακες.

Μονοδιάστατοι πίνακες

Για να ορίσουμε έναν μονοδιάστατο πίνακα πρέπει να δηλώσουμε το όνομα του πίνακα, τον τύπο δεδομένων των στοιχείων του πίνακα και το πλήθος των στοιχείων του πίνακα. Η γενική περίπτωση ορισμού ενός μονοδιάστατου πίνακα είναι:

```
τύπος_στοιχείων_πίνακα όνομα_πίνακα [πλήθος_στοιχείων_πίνακα]
```

Το όνομα του πίνακα πρέπει να είναι μοναδικό, δηλαδή να μην υπάρχει άλλη μεταβλητή με το ίδιο όνομα. Το **πλήθος** των στοιχείων στον πίνακα πρέπει να περιβάλλεται από **αγκύλες** `[ ]` αμέσως μετά το όνομα του πίνακα. Ο **τύπος** στοιχείων του πίνακα μπορεί να είναι **οποιοσδήποτε** τύπος δεδομένων (π.χ. `int`, `float`, `char`, κτλ…).

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

Παρομοίως, η εντολή `float array[10];` δηλώνει έναν πίνακα με όνομα `array`, ο οποίος περιέχει 10 στοιχεία. Καθένα από αυτά τα στοιχεία είναι ένας πραγματικός αριθμός (`float`).

Κατά τη δήλωση ενός πίνακα, ο μεταγλωττιστής δεσμεύει ένα **μπλοκ μνήμης** για να αποθηκεύσει τα στοιχεία του πίνακα. Αυτό το μπλοκ της μνήμης δεσμεύεται από μία περιοχή μνήμης που ονομάζεται **στοίβα (stack)**. Τα στοιχεία του πίνακα αποθηκεύονται σε **διαδοχικές** θέσεις μνήμης.

Π.χ. με τη δήλωση `int array[10];` ο μεταγλωττιστής δεσμεύει 40 θέσεις (οκτάδες) μνήμης για να αποθηκεύσει τα 10 στοιχεία του πίνακα. Αυτό συμβαίνει, γιατί κάθε ακέραιος αριθμός απαιτεί 4 θέσεις (οκτάδες) μνήμης.

Για να αναφερθούμε σε κάποιο στοιχείο του πίνακα γράφουμε το όνομα του πίνακα συνοδευόμενο από τον **δείκτη θέσης** του στοιχείου μέσα σε **αγκύλες `[]`**. Ο **δείκτης θέσης** είναι ένας ακέραιος αριθμός ή μία ακέραια μεταβλητή ή έκφραση, η οποία προσδιορίζει τη θέση του συγκεκριμένου στοιχείου στον πίνακα.

Το πρώτο στοιχείο ενός πίνακα με μέγεθος ***n*** στοιχεία αποθηκεύεται στη θέση [0] του πίνακα, το δεύτερο στοιχείο στη θέση [1], το τρίτο στη θέση [2], ... κ.ο.κ., με αποτέλεσμα το τελευταίο στοιχείο να αποθηκεύεται στη θέση [n-1].

Η κύρια χρησιμότητα των πινάκων είναι ότι με τους πίνακες επιτρέπεται η **ομαδοποίηση** μεταβλητών **του ίδιου τύπου** με το **ίδιο** όνομα. Για παράδειγμα, χωρίς την ύπαρξη πίνακα, αν θέλαμε να δημιουργήσουμε ένα πρόγραμμα, το οποίο να διαβάζει τους βαθμούς 1000 φοιτητών θα έπρεπε να δηλώσουμε 1000 διαφορετικές πραγματικές μεταβλητές. Όμως, με τη δήλωση:

`float grades[1000];` δημιουργείται ένας πίνακας, ο οποίος περιέχει 1000 στοιχεία και έτσι δεν χρειάζεται να δηλώσουμε 1000 διαφορετικές μεταβλητές. Οι μεταβλητές του προγράμματός μας θα είναι οι `grades[0]`, `grades[1]`, ... `grades[999]` και σε καθεμία από αυτές θα αποθηκεύεται ο βαθμός του αντίστοιχου φοιτητή.

Ακολουθούν μερικά παραδείγματα χρήσης πινάκων:

```
int arr[10]; /* Δήλωση πίνακα με στοιχεία 10 ακεραίους. */
int i,j;

i = j = 2;
arr[0] = 100; /* Η τιμή του 1ου στοιχείου του πίνακα γίνεται 100. */
arr[2+3] = 200; /* Η τιμή του 6ου στοιχείου του πίνακα γίνεται 200. */
arr[9] = arr[0]; /* Η τιμή του τελευταίου στοιχείου του πίνακα, το οποίο είναι αποθηκευμένο στην 9η θέση του πίνακα, γίνεται ίση με την τιμή του 1ου στοιχείου, δηλαδή 100. */
arr[i+j] = 300; /* Η τιμή του i+j στοιχείου του πίνακα, δηλαδή του 5ου στοιχείου αφού i+j= 2+2 = 4 γίνεται 300 */
```