ΤΕΙ ΣΕΡΡΩΝ

Προγραμματισμός Ι

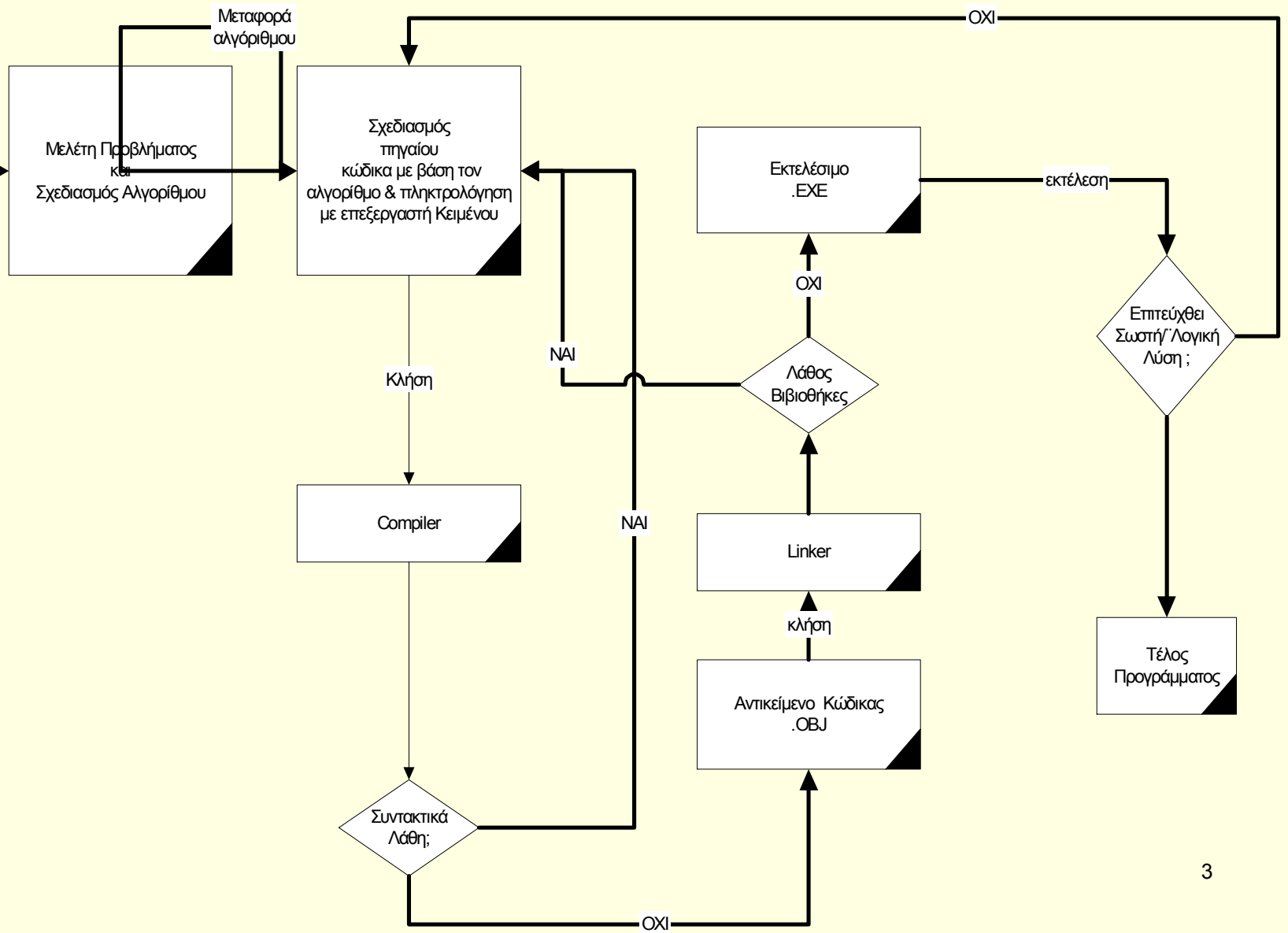
ΤΕΛΕΣΤΕΣ - ΕΚΦΡΑΣΕΙΣ

Τμήμα Πληροφορικής & Επικοινωνιών
Δρ. Θεόδωρος Γ. Λάντζος
http://www.teiser.gr/icd/staff/lantzos
lantzos@teiser.gr

Πώς δημιουργούμε πρόγραμμα Η/Υ;

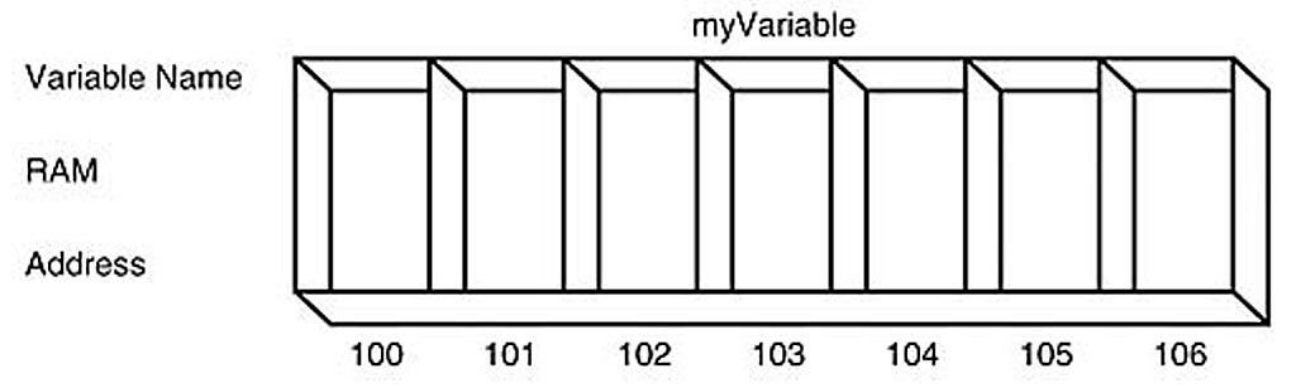
1. Ανάλυση του προβλήματος
2. Επινόηση & Σχεδιασμός λύσης (Αλγόριθμο)
3. Μεταφορά Αλγορίθμου σε Κώδικα (πηγαίο)
4. Μεταγλώττιση πηγαίου κώδικα
5. Παραγωγή του αντικείμενου κώδικα
6. Σύνθεση του αντικείμενου κώδικα
7. Εκτέλεση προγράμματος

Στάδια Υλοποίησης Προγράμματος

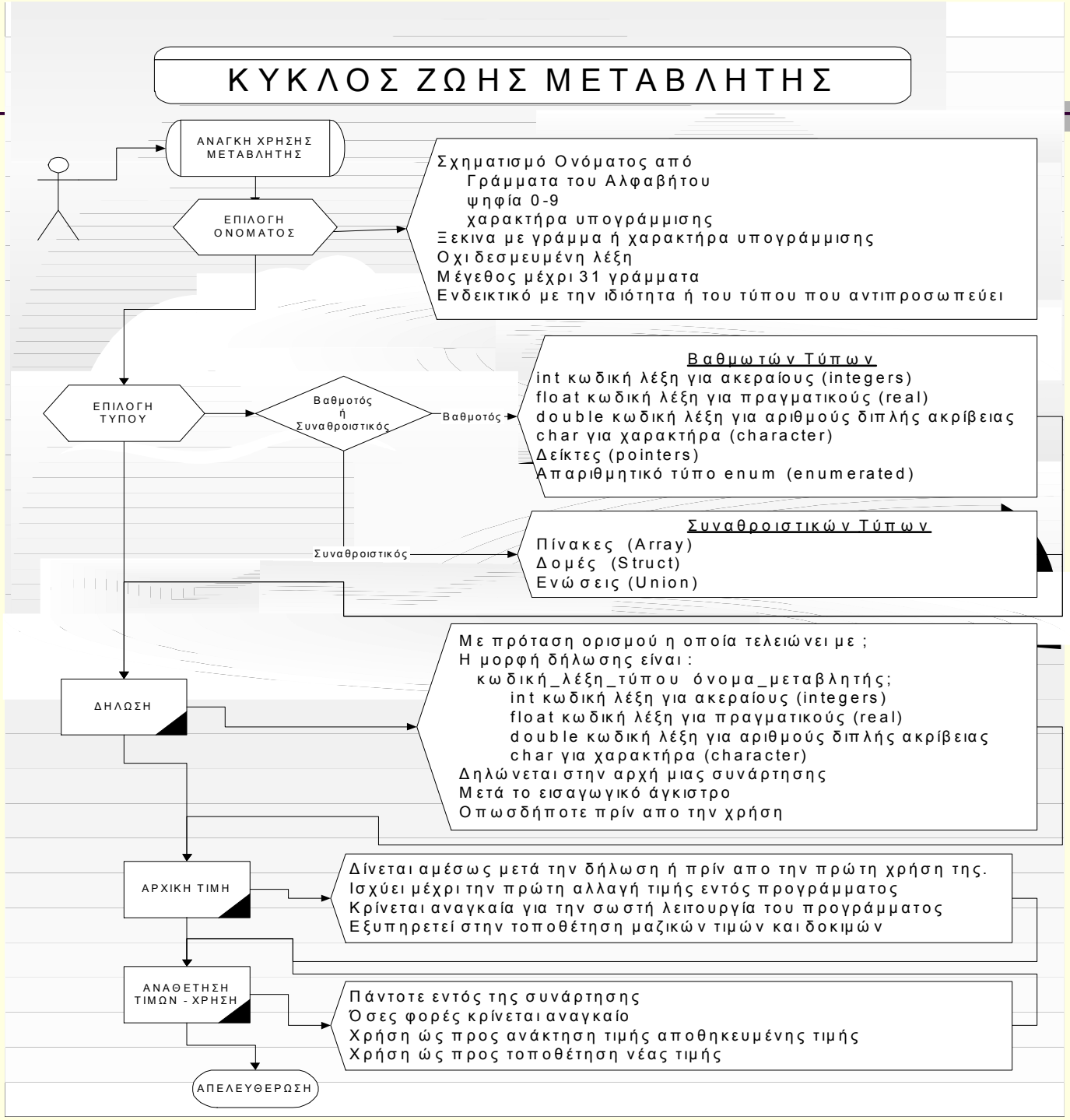


Τι είναι μεταβλητή;

- Μεταβλητή είναι μια περιοχή στην μνήμη του υπολογιστή στην οποία μπορούμε να αποθηκεύσουμε μια τιμή και να την ανακτήσουμε. Το όνομα μιας μεταβλητής είναι άμεσα συνδεδεμένο με την διεύθυνση με την οποία είναι αποθηκευμένο το δεδομένο.
- Την μνήμη του υπολογιστή μπορούμε να την φανταστούμε σαν μια σειρά άδεια κελιά τα οποία είναι στοιχισμένα σε μια διαδοχική σειρά. Το κάθε κελί ή θέση μνήμης αριθμείτε σειρι[illegible]ύσεις μνήμ[illegible]

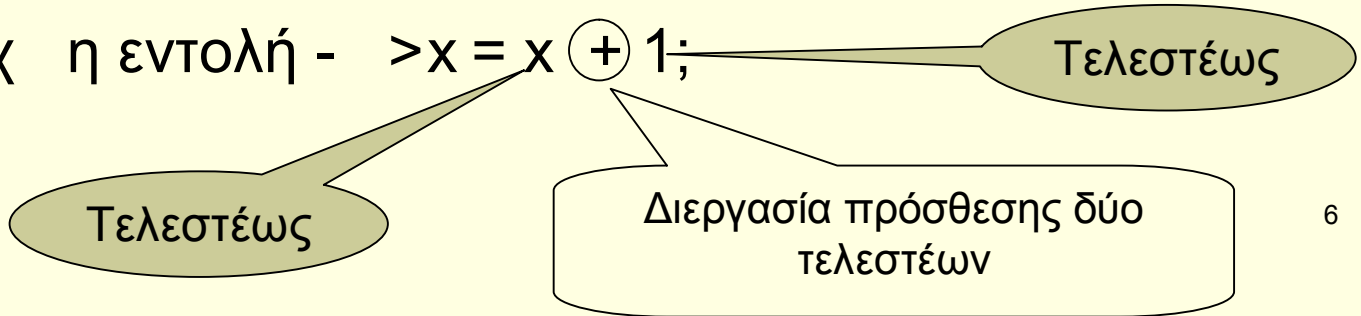


ΚΥΚΛΟΣ ΖΩΗΣ ΜΕΤΑΒΛΗΤΗΣ



Τελεστές - Τελεστέοι

- Τελεστές (operators) - Σύμβολα ή λέξεις που αναπαριστούν συγκεκριμένες διεργασίες, οι οποίες εκτελούνται επί ενός ή περισσότερων δεδομένων.
- Τελεστέοι (operands) – Δεδομένα τα οποία βρίσκονται σε μεταβλητές, σταθερές ή κλήσεις συναρτήσεων
- Χρήση Τελεστών για το σχηματισμό Εκφράσεων
- Μια έκφραση στον προγραμματισμό μπορεί να αποτελείται από έναν ή περισσότερους τελεστέους και από έναν ή περισσότερους τελεστές.
  - Π.χ η εντολή - >x = x + 1;



Τελεστές & Ταξινόμηση

Ταξινόμηση ανάλογα με :

- Α. τον αριθμό των τελεστέων στους οποίους δρουν
  1. Μοναδιαίους (unary)
  2. Δυαδικούς (binary)
  3. Τριαδικούς (ternary)
- Β. Την διεργασία που εκτελούν

Κατηγορία	Ενδεικτικοί τελεστές
Αριθμητικοί	+ - * /
Λογικοί	&& || !
Συσχετιστικοί	> >= == !=
Διαχείριση bits	>> & ! ^
Διαχείριση μνήμης	& [] . ->

Σύμβολα δυαδικών τελεστών

Δυαδικός Τελεστής	Σύμβολο
Μικρότερο	<
Μικρότερο ή ίσο	<=
Ίσο	==
Διάφορο	!=
Μεγαλύτερο	>
Μεγαλύτερο ή ίσο	>=
Πρόσθεση	+
Αφαίρεση	-
Πολλαπλασιασμός	*
Διαίρεση πραγματικών	/
Πηλίκο διαίρεσης ακεραίων	/
Υπόλοιπο διαίρεσης ακεραίων	%

Σημειογραφία Τελεστών και C

- Η θέση του τελεστή ανάμεσα σε τελεστέους παρουσιάζει διαφορετική σημειογραφία κάθε φορά.
- Ένθετου τελεστή a+b (μεταξύ τελεστέων)
- Προπορευόμενου τελεστή +ab (πρίν από τελεστέους)
- Παρελκόμενου τελεστή ab+. (μετά από τους τελεστέους)

1. Αποφυγή χρήσης τελεστών ανάμεσα σε μεικτούς τύπους τελεστέων

2. Υπάρχει διάκριση ανάμεσα στην διαίρεση ακεραίων και διαίρεση αριθμών Κινητής υποδιαστολής. Συνεπώς, η διαίρεση 7/2 θα δώσει το πηλίκο που είναι 3 Και όχι το 3.5. Για την επίτευξη σωστού αποτελέσματος πρέπει ο ένας από τους Δύο τελεστέους να είναι αριθμός κινητής υποδιαστολής. (7.0/2 = 3.5)

Γλώσσα C – Κατηγορίες Εκφράσεων

- Σταθερές (περιέχουν μόνο σταθερές τιμές)
- Ακέραιες και Κινητής υποδιαστολής (μετά από άμεσες και έμμεσες μετατροπές τύπων δίνουν αποτέλεσμα τύπου ακεραίου ή κινητής υποδιαστολής)
- Εκφράσεις Δείκτη (Εκφράσεις με τιμή μία διεύθυνση. Εκφράσεις με περιεχόμενο μεταβλητές τύπου δείκτη, τελεστή διεύθυνσης (&), αλφαριθμητικές σταθερές και ονόματα πινάκων)

Προτεραιότητα & Προσεταιριστικότητα Τελεστών

- Σχηματισμός ένθετων εκφράσεων με την χρήση παρενθέσεων
  - (((n+5)<=a)&&q)
- Η διαδοχική παράθεση τελεστών
  - 5*6-4 A. (5*6)-4 = 26 B. 5*(6-4)=10
- Προς αποφυγή των παραπάνω δυσχερειών οι τελεστές ταξινομούνται σε επίπεδα προτεραιότητας με τον κανόνα ότι οι τελεστές υψηλού επιπέδου δρουν επί των τελεστέων πριν από τους τελεστές χαμηλότερου επιπέδου
- Η ύπαρξη περισσότερων τελεστών στο ίδιο επίπεδο προτεραιότητας επιβάλλει τον προσδιορισμό κατεύθυνσης εφαρμογής από αριστερά προς τα δεξιά
  - Πχ. 8-5-2 = 1
- Τελεστές +,-,*,/ είναι αριστερής προσεταιριστικότητα
- Τελεστής = (ανάθεσης) δεξιά προσεταιριστικότητα

Πίνακας προτεραιότητας & προσεταιριστικότητας

Τελεστές	Προσεταιριστικότητα
() [] ->	Από αριστερά προς τα δεξιά
! ~ ++ -- + - * & (τύπος) sizeof	Από αριστερά προς τα δεξιά
* / % (αριθμητικοί τελεστές)	Από αριστερά προς τα δεξιά
+ - (αριθμητικοί τελεστές)	»
<< >>	»
< <= > >=	»
== ! =	»
&	»
^	»
|	»
&&	»
||	»
?:	Από δεξιά προς τα αριστερά
= += -= *= &= ^= |= <<= >>=	»
‘	»

Τελεστές αύξησης και μείωσης

- Τελεστής αύξησης (increment operator) συμβολίζεται με ++

  Π.χ num = num + 1; ισοδυναμεί με num++;

- Αντίστοιχα, Τελεστής μείωσης (decrement operator) συμβολίζεται με –
  - Πχ. num = num – 1; Ισοδυναμεί με num- - ;
- Οι τελεστές αύξησης και μείωσης μπορούν να δυσκολέψουν την κατανόηση πράξεων κυρίως όταν παρουσιάζονται ως προπορευόμενοι ή παρελκόμενοι
  - Π.χ. y= x-- + y; ή z= ++x + y;

Τελεστές αύξησης και μείωσης & προτεραιότητα

- Σε περίπτωση προπορευόμενου τελεστή το σύστημα πρώτα εκτελεί την αύξηση ή μείωση και μετά χρησιμοποιεί την νέα τιμή της μεταβλητής στην έκφραση
  - Παραδείγματα z=++x + y;
- Αντιθέτως, στην περίπτωση του παρελκόμενου τελεστή το σύστημα πρώτα χρησιμοποιεί την τιμή της μεταβλητής για τον υπολογισμό της τιμής της έκφρασης και μετά εκτελεί την αύξηση ή μείωση της τιμής της μεταβλητής
  - Παραδείγματα z= x++ +y;

Τελεστές ανάθεσης

- Σκοπός αυτών είναι να εκτελούν κάποια πράξη ανάμεσα στους τελεστέους και να εκχωρούν το αποτέλεσμα σε έναν από τους τελεστέους
  - Πχ. x*=10 ισοδυναμεί με x=x*10;
- Στην κατηγορία των τελεστών ανάθεσης έχουμε και τους τελεστές διαχείρισης δυαδικών ψηφίων (bitwise operator) >>= <<= &= ^= |=

Συσχετιστικοί Τελεστές (relational operators)

- Συγκρίνουν δύο τελεστέους
- Το αποτέλεσμα της έκφρασης με συσχετιστικούς τελεστές είναι ΑΛΗΘΕΣ(1) ή ΨΕΥΔΕΣ (0)
  - Π.χ 3<2 επιστρέφει 0
  - Π.χ. 2 == 2 επιστρέφει 1

Συσχετιστικοί Τελεστές	Δράση
<	Μικρότερο
>	Μεγαλύτερο από
<=	Μικρότερο ή ίσον από
>=	Μεγαλύτερο ή ίσον από
==	Ίσο
!=	Διάφορο

Εξετάζοντας του αριθμητικούς τελεστές με τους συσχετιστικούς παρατηρούμε ότι και οι δύο δρουν αριθμητικές εισόδους με την διαφορά ότι οι συσχετιστικοί έχουν μια δίτιμη έξοδο

Λογικοί Τελεστές

- Δρουν επι ενός ή δύο τελεστέων
- Έχουν σαν βάση την δίτιμη άλγεβρα Boole
- Οι είσοδοι και έξοδοι μπορούν να λάβουν μόνο δύο τιμές TRUE και FALSE

Τελεστής	Δράση
&&	Λογικό AND
||	Λογικό OR
!	Λογικό NOT

p	q	p&&q	p||q	!p
T	T	T	T	F
T	F	F	T	
F	T	F	T	T
F	F	F	F	

Π.Χ (10+5)<(12+8) ->15<20 -> TRUE
Πχ (10>5)||(-8>10) -> true || false -> true

Μετατροπές τύπων

- Όταν έχουμε τελεστέους διαφορετικών τύπων αυτοί μετατρέπονται σε ενιαίο
  - (α) Με αυτόματο τρόπο (Υπονοούμενη)
  - (β) Με ρητή εντολή του προγραμματιστή
- Α. Υπονοούμενη
  - 3.0+ ½- >3.0 και όχι 3.5
  - Όταν έχουμε δύο τύπους δεδομένων, ο στενότερος μετατρέπεται στον ευρύτερο χωρίς απώλεια
  - Οι τύποι της γλώσσας ταξινομούνται ανάλογα με το μέγεθος μνήμης που απαιτούν για αποθήκευση
    - Char < int < long < float < double
  - Στην γλώσσα C οι αριθμητικές εκφράσεις μετατρέπουν αυτόματα τον τύπο char σε int και το float σε double

Παράδειγμα υπονοούμενων μετατροπών

```
# include <stdio.h>
main()
{
   char ch;
   int i;
   float fl;
   fl=i=ch='A'; //(1)
   printf("ch=%c, i=%d, fl=%2.2f,\n",ch,I,fl);
   ch=ch+1;
   i=fl+2*ch;
   printf("ch=%c, i=%d, fl=%2.2f,\n",ch,I,fl);
}
```

Β. Ρητές μετατροπές

- Μετατροπή μιας τιμής σε διαφορετικό τύπο κατόπιν εντολής του προγραμματιστή
- Η διαδικασία ονομάζεται προσαρμογή
- Ο τελεστής μετατροπής είναι μοναδιαίος και έχει την μορφή (τύπος δεδομένων)
- Τοποθετείται μπροστά από μια έκφραση για να μετατρέψει την τιμή της στον περικλειόμενο σε παρενθέσεις τύπο
  - 5/2 -> 2
  - (float) 5 / (float) 2 -> 2.5

Τελεστές sizeof

- Επιστρέφει τον αριθμό των bytes που η τιμή της έκφρασης ή ο τύπος δεδομένων καταλαμβάνει στη μνήμη
- Ενεργεί επάνω σε εκφράσεις sizeof(x+y)

  Το σύστημα δεν υπολογίζει την τιμή της έκφρασης (βλέπε παράδειγμα)
- Και σε τύπο δεδομένων πχ. sizeof(int)

```
#include <stdio.h>
#include <conio.h>

main()
{

   int x,y;

   printf("x=%d",sizeof(x));
   printf("\ny=%d",sizeof(y));
   printf("\nx+y=%d",sizeof(x+y));
   getch();
}
```

Το πρόγραμμα επιστρέφει

```
x=4
y=4
x+y=4
```