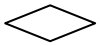
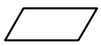
1 ΕΙΣΑΓΩΓΗ – ΤΥΠΟΙ ΔΕΔΟΜΕΝΩΝ

Αλγόριθμος

Μια σειρά από σαφή και καθορισμένα βήματα, τα οποία οδηγούν στη λύση ενός προβλήματος, περιγραφή του κάθε βήματος με λόγια και λέξεις-κλειδιά, π.χ. **διάβασε**, **υπολόγισε**, **εμφάνισε**, **αν**, **διαφορετικά**, **για όσο**, **για**.

Διάγραμμα Ροής (Λογικό Διάγραμμα)

Μια σειρά από σαφή και καθορισμένα βήματα, τα οποία οδηγούν στη λύση ενός προβλήματος με ειδικά σχήματα για την κάθε ενέργεια π.χ. **ορθογώνιο**  για την ανάθεση τιμής, **ρόμβος**  για έλεγχο ή επανάληψη, **παραλληλόγραμμο**  για εισαγωγή δεδομένων, ταινία  για εμφάνιση αποτελεσμάτων κ.λ.π..

Πρόγραμμα

Μετατροπή των παραπάνω βημάτων σε **εντολές** που μπορούν να μεταφραστούν από ένα πρόγραμμα στον Ηλεκτρονικό Υπολογιστή (Μεταγλωττιστής ή Διερμηνέας μιας Γλώσσας Προγραμματισμού), με λέξεις-κλειδιά, π.χ. **read**, **print**, **if**, **else**, **for**, **while**.

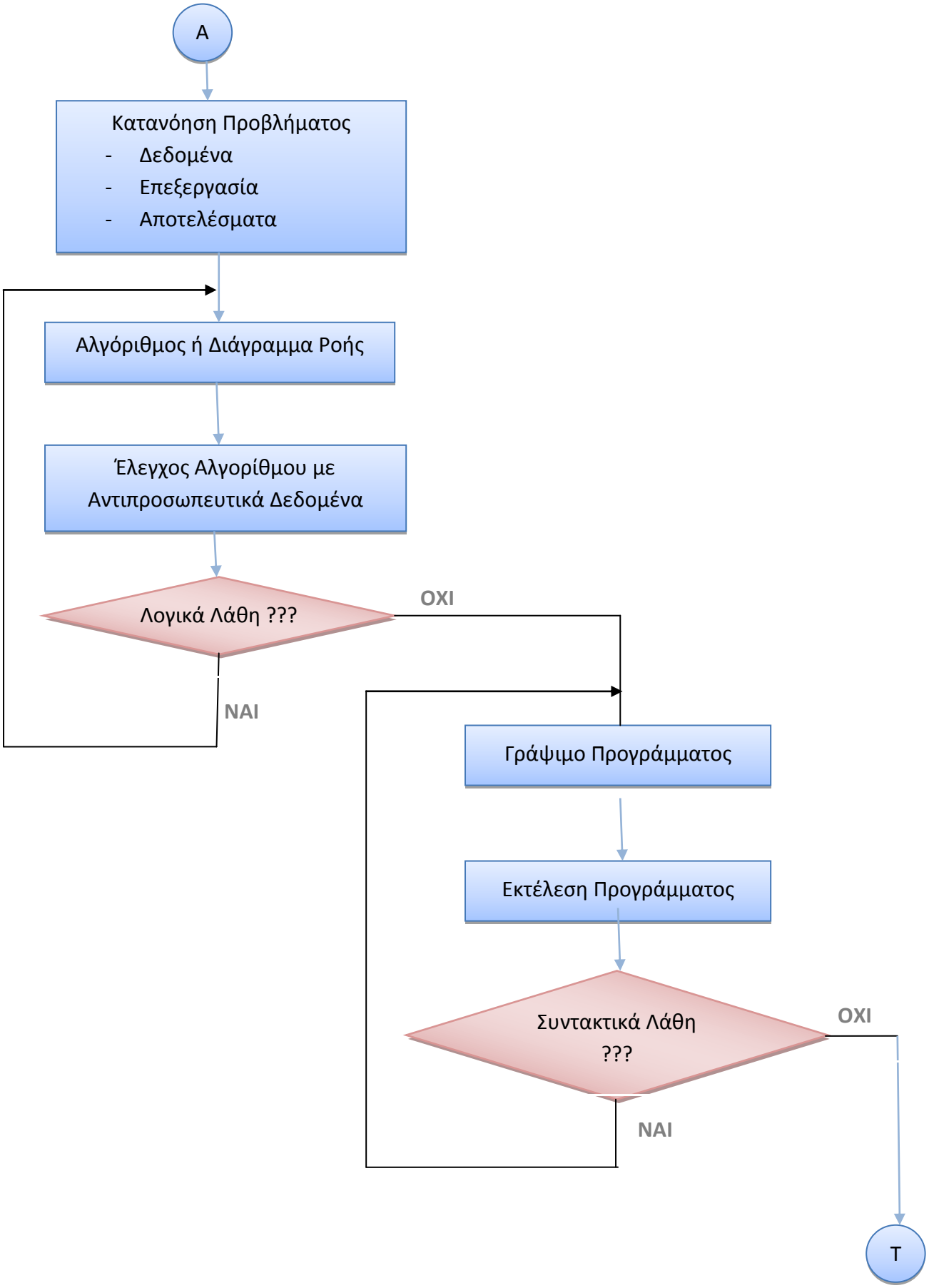
Γλώσσες Προγραμματισμού

Γλώσσα Μηχανής → Συμβολική Γλώσσα (ASSEMBLY) → Γλώσσες Υψηλού Επιπέδου (BASIC, FORTRAN, COBOL, PASCAL, C, Dephi, Visual Basic, C++, Java).

Μεταβλητές

Ονόματα στα αγγλικά, τα οποία αντιπροσωπεύουν θέσεις μνήμης, στις οποίες θα αποθηκεύονται δεδομένα, αριθμοί, χαρακτήρες κ.λ.π.. Συνήθως τα ονόματα που επιλέγονται έχουν σχέση με την ποσότητα που θα αποθηκεύσουν, π.χ. **num** για κάποιον αριθμό, **sum** για άθροισμα, **mo** για το Μέσο Όρο.

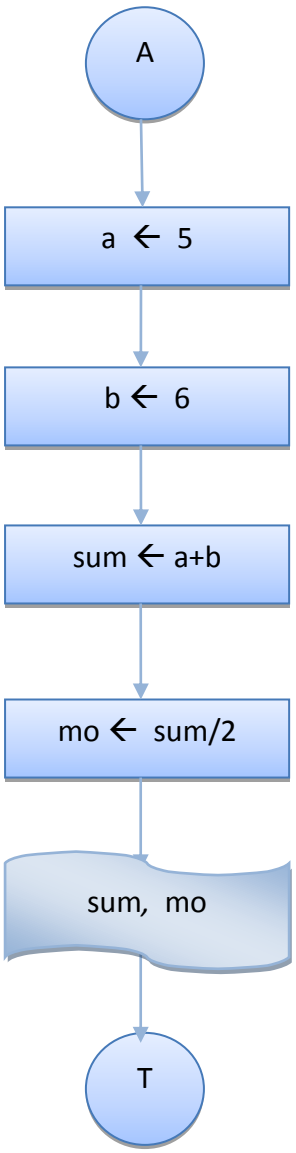
ΔΙΑΓΡΑΜΜΑ ΕΡΓΑΣΙΩΝ ΓΙΑ ΜΕΘΟΔΙΚΟ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ



1. 1 Ένα Απλό Πρόγραμμα

Να γραφεί πρόγραμμα, το οποίο θα δίνει τις τιμές 5 και 6 σε δύο μεταβλητές **num1** και **num2** και θα υπολογίζει και θα εμφανίζει το άθροισμά τους **sum** και το μέσο όρο **mo**.

ΛΟΓΙΚΟ ΔΙΑΓΡΑΜΜΑ



ΑΛΓΟΡΙΘΜΟΣ

1. Δίνω την τιμή 5 στο a (**a ← 5**)
2. Δίνω την τιμή 6 στο b (**b ← 6**)
3. Βρίσκω το άθροισμα (**sum ← a + b**)
4. Βρίσκω τον μέσο όρο (**mo ← sum/2**)
5. Εμφανίζω τις τιμές των **sum, mo**

ΠΡΟΓΡΑΜΜΑ

```
public class SumMO1 {
/*
Πρόγραμμα που δίνει τις τιμές 5 και 6 σε 2 ακέραιες μεταβλητές και βρίσκει
και εμφανίζει το άθροισμά τους και το Μέσο Όρο, ο οποίος είναι μεταβλητή
τύπου double

*/
    public static void main(String[] args) {

        // Δήλωση των ακέραιων μεταβλητών num1, num2
        int num1, num2;

        // Δήλωση της ακέραιας μεταβλητής sum για το άθροισμα
        int sum;

        // Δήλωση της πραγματικής μεταβλητής mo για το Μέσο Όρο
        double mo;

        // Ανάθεση των τιμών 5 και 6 στις ακέραιες μεταβλητές num1, num2
        num1 = 5;
        num2 = 6;

        // Υπολογισμός του αθροίσματος sum
        sum = num1 + num2;

        // Υπολογισμός του Μέσου Όρου mo
        mo = sum/2;

        // Εμφάνιση  του αθροίσματος sum και του Μέσου Όρου mo
        System.out.println("Άθροισμα = " + sum + " Μέσος Όρος = " + mo);
        }
}
```

Έξοδος Προγράμματος

```
Άθροισμα = 11 Μέσος Όρος = 5.0
```

- Μέσος Όρος = 5, γιατί γίνεται ακέραια διαίρεση του 11/2 και αποκόπτεται το δεκαδικό μέρος.

Λέξεις – κλειδιά στον ορισμό του προγράμματος :

`class SumMO1` : Ορισμός της κλάσης `Sum_mo_1`. Ο ορισμός περιλαμβάνει μόνο μία μέθοδο, τη `main()`.

`public` : Η λέξη **`public`** δηλώνει ότι η μέθοδος είναι προσπελάσιμη από παντού.

`static` : Η λέξη **`static`** δηλώνει ότι η μέθοδος είναι προσπελάσιμη ακόμη και αν δεν έχουν δημιουργηθεί αντικείμενα της κλάσης.

`void` : Σημαίνει ότι η μέθοδος `main()` δεν επιστρέφει καμιά τιμή.

`main()` : Η βασική μέθοδος.

ΠΑΡΑΤΗΡΗΣΕΙΣ

1. Οι αγκύλες **`{}`** πηγαίνουν ανά ζεύγη και περικλείουν αυτόνομα κομμάτια κώδικα.
2. Όλες οι εντολές τελειώνουν με το ελληνικό ερωτηματικό "**`;`**".
3. Τα Σχόλια πολλών γραμμών αρχίζουν με το σύμβολο "**`/*`**" και τελειώνουν με το ύμβολο "**`*/`**".
4. Τα Σχόλια μιας γραμμής ή μετά από μια εντολή αρχίζουν με το σύμβολο "**`//`**".
5. Οι μεταβλητές πρέπει να δηλώνουν κάποιο τύπο ανάλογα με την ποσότητα που θα αποθηκεύσουν, π.χ. `int num1, num2` για τους αριθμούς, `double mo` για το Μέσο Όρο.
6. Τα ονόματα των μεταβλητών μπορούν να αρχίζουν από οποιοδήποτε γράμμα ή τα σύμβολα "**`_`**" "**`$`**", αλλά <u>όχι</u> από ψηφίο. Π.χ. `num1` **<u>και όχι</u>** `1num`.
7. Η δήλωση μιας μεταβλητής περιλαμβάνει τον τύπο των δεδομένων που θα αποθηκεύσει και το όνομά της.

   Παράδειγμα

   ```
   double mo; // Δήλωση της πραγματικής μεταβλητής mo για το Μέσο Όρο
   ```

8. Μια δήλωση μπορεί να περιλαμβάνει περισσότερες από μια μεταβλητές.

   Παράδειγμα

   ```
   int num1, num2; // Δήλωση των ακέραιων μεταβλητών num1, num2
   ```

9. Για την εμφάνιση των αποτελεσμάτων χρησιμοποιούμε την εντολή **`System.out.println(<μηνύματα και ονόματα μεταβλητών>);`**

Η εντολή `System.out.println()` μπορεί **μέσα στις παρενθέσεις** να περιλαμβάνει :

- **Ονόματα** μεταβλητών, οπότε εμφανίζει απλώς τις τιμές τους.

Παράδειγμα :

```
System.out.println(mo);     // Θα εμφανίσει ΜΟΝΟ την τιμή της μεταβλητής mo
```

- **Ονόματα** μεταβλητών και **μηνύματα**, οπότε εμφανίζει και μηνύματα και τις τιμές των μεταβλητών.

Παράδειγμα :

```
System.out.println("mo =  " + mo);    // Θα εμφανίσει το μήνυμα mo  = και την τιμή της μεταβλητής mo
```

- **Τίποτα**, οπότε γίνεται απλώς αλλαγή γραμμής.

```
System.out.println();          // αλλαγή γραμμής
```

1.2 ΤΥΠΟΙ ΔΕΔΟΜΕΝΩΝ

Τα δεδομένα που μπορούν να αποθηκευθούν σε μια μεταβλητή μπορεί να είναι ακέραιοι αριθμοί (π.χ. ο αριθμός `5`), πραγματικοί αριθμοί ή αριθμοί κινητής υποδιαστολής (π.χ. ο αριθμός `5.5`), χαρακτήρες (π.χ. το γράμμα `X`), συμβολοσειρές (π.χ. το όνομα `Nikos`).

Οι **ακέραιες** μεταβλητές ανάλογα με τα `bits` που περιλαμβάνουν και το μέγιστο αριθμό που μπορούν να αποθηκεύσουν φαίνονται στον επόμενο πίνακα :

Τύπος	**bits**	**Ελάχιστη - Μέγιστη Τιμή**	
byte	8	-128	127
short	16	-32628	32627
int	32	$-2*10^{9}$	$2*10^{9}$
long	64	$-9*10^{18}$	$9*10^{18}$

Οι **πραγματικές** (**Κινητής Υποδιαστολής**) μεταβλητές ανάλογα με τα `bits` που περιλαμβάνουν συνολικά, τα `bits` της `mantissa` και το μέγιστο αριθμό που μπορούν να αποθηκεύσουν φαίνονται στον επόμενο πίνακα :

Τύπος	**Συνολικά bits**	**bits mantissa**	**Ελάχιστη -Μέγιστη Τιμή**
float	32	23	$-3.4*10^{38}$ $3.4*10^{38}$
double	64	52	$-1.7*10^{308}$ $1.7*10^{308}$

❖ Όλες οι μέθοδοι δέχονται `double` παραμέτρους.

Το σύνολο **χαρακτήρων** της `Java` είναι το `Unicode` και όχι το `ASCII`. Χρησιμοποιεί `16` bits και όχι `8`, όπως το σύνολο των χαρακτήρων `ASCII`, με τη δυνατότητα αποθήκευσης χαρακτήρων από όλες τις γλώσσες (από `0-65536`, ενώ οι χαρακτήρες από `0-127` είναι όπως και στις άλλες γλώσσες προγραμματισμού, σύνολο `ASCII`).

```
char ch;     // Δήλωση μεταβλητής που θα αποθηκεύσει χαρακτήρες

ch = 'X';    // Ανάθεση τιμής σε μεταβλητή που αποθηκεύει το χαρακτήρα X

ch = 88;    // Ανάθεση τιμής σε μεταβλητή που αποθηκεύει το αριθμητικό
              ισοδύναμο του χαρακτήρα X
```

Οι λογικές (**boolean**) μεταβλητές αποθηκεύουν τις τιμές **true-false**, σαν το αποτέλεσμα κάποιας σύγκρισης. Χρησιμοποιούνται σε εντολές ελέγχου και επανάληψης.

```
boolean b; // Δήλωση της λογικής μεταβλητής b
b = true; // Ανάθεση της τιμής true  στη λογική μεταβλητή b
System.out.println(10 > 9); // Εμφανίζει true
```

ΚΥΡΙΟΛΕΚΤΙΚΕΣ ΣΤΑΘΕΡΕΣ

Πολλές φορές χρειάζεται να χρησιμοποιήσουμε συγκεκριμένους αριθμούς, χαρακτήρες ή ονόματα σε εντολές εκχώρησης, σε εκφράσεις ή σε συγκρίσεις, όπως στο προηγούμενο παράδειγμα που δώσαμε στη μεταβλητή `num1` την τιμή `5`, τα οποία χαρακτηρίζονται σαν κυριολεκτικές σταθερές και μπορεί να είναι :

- Σταθερές ακεραίων, π.χ. `10, -100.` Είναι εξ ορισμού τύπου `int`. Αν θέλουμε να ορίσουμε σταθερά τύπου `long`, βάζουμε ένα `L` ή `l` στο τέλος, π.χ. `long a = 10L`.
- Σταθερές κινητής υποδιαστολής, π.χ. `2.35`. Είναι εξ ορισμού τύπου `double`. Για να ορίσουμε σταθερά τύπου `float` , βάζουμε ένα `F` ή `f` στο τέλος, π.χ. `float a=2.35F`.
- Σταθερές συμολοσειρές (`Strings`). Είναι αλυσίδες χαρακτήρων π.χ. “`kostas`” και χρησιμοποιούνται συνήθως σε εντολές `println`.

ΧΑΡΑΚΤΗΡΕΣ ΔΙΑΦΥΓΗΣ

Χρησιμοποιούνται σε εντολές `println` και είναι το “`\t`” για tab και το “`\n`” για αλλαγή γραμμής.

Παράδειγμα

Η εντολή `System.out.println(“a\tb\nc\t+d\n”);` θα έχει σαν αποτέλεσμα την παρακάτω εμφάνιση :

```
a  b
c  d
```

Οι χαρακτήρες απόστροφος και διπλά εισαγωγικά για να χρησιμοποιθούν ή να εκτυπωθούν πρέπει να προηγείται μια ανάποδη παύλα.

Παράδειγμα

```
ch1 =  ‘\’’;
ch2 = ’\”’;
```

ΑΡΧΙΚΟΠΟΙΗΣΗ ΜΕΤΑΒΛΗΤΩΝ

Μαζί με τη δήλωση κάποιας μεταβλητής μπορούμε να εκχωρήσουμε και κάποια αρχική τιμή.

Παράδειγμα

```
int a;   // Δήλωση Μεταβλητής a
a = 10; // Ανάθεση του 10 σαν αρχική τιμή στη μεταβλητή a
```

Το ίδιο αποτέλεσμα έχουμε με την εντολή

```
int a = 10;   // Δήλωση - Ανάθεση του 10 σαν αρχική τιμή στη μεταβλητή a
```

❖ Αρχικοποίηση μεταβλητών μπορεί να γίνει σε περισσότερες από 1 μεταβλητές.

Παράδειγμα

```
int a = 5, b = 6, sum; // Αρχικοποίηση a, b, δήλωση sum
```

ΔΥΝΑΜΙΚΗ ΑΡΧΙΚΟΠΟΙΗΣΗ

Μπορεί να γίνει δήλωση μιας μεταβλητής στο σημείο του προγράμματος που θέλουμε να **υπολογίσουμε** και να **αποθηκεύσουμε** σε μια μεταβλητή το αποτέλεσμα μιας αριθμητικής έκφρασης.

Παράδειγμα

```
int sum = a + b;
double mo = sum/2;
```

❖ Προσοχή, η μεταβλητή αν υπάρχει σε `block` (`if`, `while`, `for`) είναι τοπική, έχει εμβέλεια μόνο στο `block`.

ΑΝΑΘΕΣΗ-ΕΚΧΩΡΗΣΗ ΤΙΜΗΣ

Γενικός Τύπος : `<όνομα_μεταβλητής> = <έκφραση>;`

Παράδειγμα

```
a = 10; // Ανάθεση του 10 σαν αρχική τιμή στη με ταβλητή a
```

❖ Υπάρχει η δυνατότητα στη `Java` να έχουμε αλυσίδα εκχωρήσεων.

Παράδειγμα

```
x = y = z = 10; // Το z παίρνει την τιμή 10, την οποια παίρνει το y και μετά το x
```

ΑΡΙΘΜΗΤΙΚΟΙ ΤΕΛΕΣΤΕΣ

Οι τελεστές πράξεων που χρησιμοποιούνται στις αριθμητικές εκφράσεις είναι οι παρακάτω :

`+, -, *` : Τελεστές για πρόσθεση, αφαίρεση και πολλαπλασιασμό, όπως και στα μαθηματικά.

`/` : Τελεστής για Διαίρεση

- Με ακέραιους γίνεται ακέραια διαίρεση

Παράδειγμα

```
int a = 10;
int b = a/3;        // Αποτέλεσμα της διαίρεσης → b = 3

float a = 10.0f;
float b = a/3;      // Αποτέλεσμα της διαίρεσης → b = 3.333….
```

`%` : Τελεστής για το υπόλοιπο της διαίρεσης 2 αριθμών, ακέραιων ή κινητής υποδιαστολής.

Παράδειγμα

```
int a = 10;
int b = a%3;   //  Αποθηκεύεται στη μεταβλητή b το υπόλοιπο της διαίρεσης a/3 = 1

float a = 10.5f;
float b = a%3; //Αποθηκεύεται στη μεταβλητή b το υπόλοιπο της διαίρεσης a/3 =  1.5
```

1. 3 Πρώτη Τροποποίηση Προγράμματος 1.1

Να τροποποιηθεί το πρόγραμμα 1.1, το οποίο θα δίνει τις τιμές 5 και 6 σε δύο μεταβλητές **num1** και **num2** και θα υπολογίζει και θα εμφανίζει το άθροισμά τους **sum** και το μέσο όρο **mo**, ώστε να μπορεί να εμφανίζει το σωστό αποτέλεσμα (η μεταβλητή `sum` να δηλωθεί τύπου `double`, ώστε να ΜΗ γίνει ακέραια διαίρεση στο μέσο όρο).

ΠΡΟΓΡΑΜΜΑ

```
public class SumMO2 {
/*
Πρόγραμμα που δίνει τις τιμές 5 και 6 σε 2 ακέραιες μεταβλητές και βρίσκει
και εμφανίζει το άθροισμά τους και το Μέσο Όρο. To άθροισμα και ο Μέσος Όρος
είναι μεταβλητές τύπου double
*/
    public static void main(String[] args) {

        // Δήλωση των ακέραιων μεταβλητών num1, num2
        int num1, num2;

        // Δήλωση της πραγματικής μεταβλητής sum για το άθροισμα
        double sum;

        // Δήλωση της πραγματικής μεταβλητής mo για το Μέσο Όρο
        double mo;

        // Ανάθεση των τιμών 5 και 6 στις ακέραιες μεταβλητές num1, num2
        num1 = 5;
        num2 = 6;

        // Υπολογισμός του αθροίσματος sum
        sum = num1 + num2;

        // Υπολογισμός του Μέσου Όρου mo
        mo = sum/2;

        // Εμφάνιση  του αθροίσματος sum και του Μέσου Όρου mo
        System.out.println("Άθροισμα = " + sum + " Μέσος Όρος = " + mo);
        }
}
```

Έξοδος Προγράμματος

```
Άθροισμα = 11.0 Μέσος Όρος = 5.5
```

- Το Άθροισμα είναι `11.0` και ο Μέσος Όρος είναι `5.5`

ΤΕΛΕΣΤΕΣ ΣΥΝΤΟΜΩΝ ΕΚΧΩΡΗΣΕΩΝ

Υπάρχει η δυνατότητα στη Java να έχουμε σύντομες εκχωρήσεις .

Παράδειγμα

```
x = x + y;     Συντομευμένος τελεστής    x += y;
x = x – y;     Συντομευμένος τελεστής    x -= y;
x = x * y;     Συντομευμένος τελεστής    x *= y;
x = x/y;       Συντομευμένος τελεστής    x /= y;
x = x%y        Συντομευμένος τελεστής    x %= y;
```

ΤΕΛΕΣΤΕΣ ΠΡΟΣΑΥΞΗΣΗΣ-ΠΡΟΜΕΙΩΣΗΣ ++, --

Η Java διαθέτει τους παρακάτω τελεστές προσαύξησης ή προμείωσης :

```
x = x + 1;    τελεστές προσαύξησης     x++; ++x;
x = x – 1;    τελεστές προμείωσης      x--; --x;
```

- Αν το ++, -- βρίσκεται **πριν** τη μεταβλητή, **πρώτα** ενημερώνεται η τιμή της μεταβλητής και **μετά** χρησιμοποιείται.

Παράδειγμα

```
x = 10;
y = x++;    // Το y παίρνει την τιμή του x που ήταν 10, το x αυξάνεται ΜΕΤΑ και γίνεται
              11 ( x = 11, y = 10 )

x = 10;
y = ++x;    // ΠΡΩΤΑ αυξάνεται το x και γίνεται 11 και ΜΕΤΑ το y παίρνει την τιμή του x
              που έγινε 11 ( x = 11, y = 11 )
```

ΜΕΤΑΤΡΟΠΗ ΤΥΠΩΝ ΣΕ ΕΚΧΩΡΗΣΕΙΣ

Όταν σε μια αριθμητική έκφραση έχουμε μεταβλητές διαφορετικού τύπου γίνεται **διευρυμένη μετατροπή**, όπου ο τύπος της μεταβλητής στο δεξί μέρος της έκφρασης μετατρέπεται στον τύπο της μεταβλητής στο αριστερό μέρος της έκφρασης. Η μετατροπή γίνεται από το μικρότερο τύπο στο μεγαλύτερο με την παρακάτω σειρά:

byte → short → int → long → float → double

Για να γίνει διευρυμένη μετατροπή θα πρέπει ο τύπος προορισμού (αριστερά) να είναι μεγαλύτερος από τους τύπους των μεταβλητών στο δεξί μέρος της έκφρασης. Η διευρυμένη μετατροπή δεν ισχύει για τους τύπους `boolean` και `char`.

Εξαίρεση αποτελεί ο τύπος `char`, όπου μπορεί να αποθηκευτεί ένας ακέραιος `int` σε μεταβλητή τύπου `char`.

Παράδειγμα

```
char ch = 88; // Είναι το ίδιο με την εντολή char ch = 'X'
```

Στην περίπτωση που δεν ισχύουν τα παραπάνω ή αν θέλουμε να αποθηκεύσουμε το περιεχόμενο της μεταβλητής κάποιου τύπου σε μεταβλητή διαφορετικού τύπου χρησιμοποιούμε τη **διανομή – casting,** όπου πριν τη **μεταβλητή** ή την **έκφραση** (η οποία πρέπει να είναι μέσα σε **παρενθέσεις**, αν θέλουμε η διανομή να ισχύσει για το αποτέλεσμα της έκφρασης) βάζουμε μέσα σε **παρενθέσεις** τον τύπο στον οποίο θέλουμε να μετατραπεί η τιμή της μεταβλητής ή της έκφρασης. Σ' αυτές τις περιπτώσεις χρειάζεται προσοχή, γιατί μπορεί να μην αποθηκευθεί σωστά η τιμή μιας μεταβλητής σε κάποια άλλη ή να μή χωράει.

Παράδειγμα

```
double d = 10.5;
int i1 = (int)d;          // Αποθηκεύεται στη μεταβλητή i1 το 10, χάνεται το 0.5
int i2 = (int)(d/3);      // Αποθηκεύεται στη μεταβλητή i2 το 3, χάνεται το 0.5
double d1 = (double)i1;   // Αποθηκεύεται στη μεταβλητή d1 το 10.0

byte b1 = (byte)i2;       // Αποθηκεύεται στη μεταβλητή b1 το 3

byte b2 = (byte)(i2+200)//Αποθηκεύεται στη μεταβλητή b2 το 1, γιατί η τιμή
                          i2+200=203 είναι μεγαλύτερη από τη μέγιστη τιμή 127
                          που μπορεί να αποθηκεύσει μια μεταβλητή τύπου byte

byte b = 88;
char ch1 = (char) b;  // Αποθηκεύεται στη μεταβλητή ch1 ο χαρακτήρας 'X'

int i = 88;
char ch2 = (char) i;  // Αποθηκεύεται στη μεταβλητή ch2 ο χαρακτήρας 'X'
```

1. 4 Δεύτερη Τροποποίηση του Προγράμματος 1.1

Να τροποποιηθεί το πρόγραμμα 1.1, το οποίο θα δίνει τις τιμές 5 και 6 σε δύο μεταβλητές **num1** και **num2** και θα υπολογίζει και θα εμφανίζει το άθροισμά τους **sum** και το μέσο όρο **mo**, ώστε να μπορεί να εμφανίζει το σωστό αποτέλεσμα (**η μεταβλητή sum να δηλωθεί τύπου int, αλλά να γίνει μετατροπή - casting στον υπολογισμό του μέσο όρου, ώστε να ΜΗ γίνει ακέραια διαίρεση**).

ΠΡΟΓΡΑΜΜΑ

```
public class SumMO3 {
/*
Πρόγραμμα που δίνει τις τιμές 5 και 6 σε 2 ακέραιες μεταβλητές και βρίσκει
και εμφανίζει το άθροισμά τους και το Μέσο Όρο, ο οποίος είναι μεταβλητή
τύπου double. Η μεταβλητή sum θα δηλωθεί τύπου int, αλλά θα γίνει μετατροπή -
casting στον υπολογισμό του μέσο όρου, ώστε να ΜΗ γίνει ακέραια διαίρεση
*/
    public static void main(String[] args) {

        // Δήλωση των ακέραιων μεταβλητών num1, num2
        int num1, num2;

        // Δήλωση της ακέραιας μεταβλητής sum για το άθροισμα
        int sum;

        // Δήλωση της πραγματικής μεταβλητής mo για το Μέσο Όρο
        double mo;

        // Ανάθεση των τιμών 5 και 6 στις ακέραιες μεταβλητές num1, num2
        num1 = 5;
        num2 = 6;

        // Υπολογισμός του αθροίσματος sum
        sum = num1 + num2;

        // Υπολογισμός  Μέσου Όρου mo με μετατροπή-casting του sum σε double
        mo = (double)sum/2;

        // Εμφάνιση  του αθροίσματος sum και του Μέσου Όρου mo
        System.out.println("Άθροισμα = " + sum + " Μέσος Όρος = " + mo);
        }
}
```

Έξοδος Προγράμματος

```
Άθροισμα = 11 Μέσος Όρος = 5.5
```

- Παρόλο που το άθροισμα είναι ακέραιος (= 11) με τη μετατροπή - casting του αθροίσματος στον υπολογισμό του μέσο όρου (**mo = (double)sum/2**) ΔΕ γίνεται ακέραια διαίρεση του 11/2 και ΔΕΝ αποκόπτεται το δεκαδικό μέρος, οπότε ο Μέσος Όρος = 5.5.

ΥΠΟΛΟΓΙΣΜΟΣ ΕΚΦΡΑΣΕΩΝ

Στον υπολογισμό των εκφράσεων, οι τιμές των μεταβλητών προάγονται σε αντίστοιχες τιμές των μεταβλητών μεγαλύτερων τύπων με την παρακάτω σειρά :

char, byte, short → int → long → float → double

- **Προσοχή σε πράξεις με μεταβλητές τύπου `char`, `byte`, `short` σε εκφράσεις στο δεύτερο μέλος της εντολής ανάθεσης τιμής χρειάζεται διανομή, γιατί οι μεταβλητές των παραπάνω τύπων προάγονται αυτόματα σε `int`.**

Παράδειγμα

```
byte b = 10;
int i = b * b;      // i = 100, byte → μετατροπή σε int

b = b * b;          // Πρόβλημα, b*b → μετατροπή σε int
b = (byte)(b*b);    // i = 100, byte → μετατροπή σε int → (byte)
                       μετατροπή σε byte
short s = 10;
s = s + 1;             // Πρόβλημα, s+1 → μετατροπή σε int
s = (short)(s + 1);    // s+1 → μετατροπή σε int → (short)
                       μετατροπή σε short

char ch = 'a';
ch = ch+1;             // Πρόβλημα, ch+1 → μετατροπή σε int
ch =(char)(ch+1);      // ch+1 → μετατροπή σε int → (char) μετατροπή
                          σε char
```

- Η ακέραια διαίρεση, αν είναι ατελής, χρειάζεται διανομή.

Παράδειγμα

```
int i = 10/3;                  // Αποτέλεσμα i = 3
double d = (double)(10/3);     // Αποτέλεσμα i = 3.0, η διανομή γίνεται στο
                                  αποτέλεσμα της ακέραιας διαίρεσης 10/3 = 3
d = (double)10/3;              // Αποτέλεσμα i = 3.333, η διανομή γίνεται
                                  στο 10 → μετατροπή σε 10.0/3 = 3.333
```