Προχωρημένος Προγραμματισμός

Εισαγωγή στον αντικειμενοστραφή προγραμματισμό με τη Java

ΕΛΕΥΘΕΡΙΟΣ ΚΟΣΜΑΣ

ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ 2022-2023 | ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Περίληψη

Σήμερα ...

- Θα παρουσιάσουμε το απλούστερο δυνατό πρόγραμμα Java
- Θα παρουσιάσουμε τον τρόπο μεταγλώττισης και εκτέλεσης προγραμμάτων Java
- Θα συζητήσουμε στυλ προγραμματισμού (στοίχιση κώδικα, ονοματολογία)

- Θα συζητήσουμε για τύπους , μεταβλητές και τιμές
- Θα παρουσιάσουμε τον τρόπο εισόδου/εξόδου δεδομένων από το πρόγραμμά μας με τη χρήση βασικών ροών δεδομένων

- Θα υλοποιήσουμε απλά προγράμματα

"Hello World"

Το πρώτο μας πρόγραμμα σε Java

Πρόγραμμα Java

Βασική Δομή - 1ο πρόγραμμα: "Hello World"

- το όνομα του αρχείου που κρατάει το πρόγραμμα είναι **`X.java`**
  - όπου **`X`** το όνομα του προγράμματος
  - στο παράδειγμα μας ονομάζουμε το πρόγραμμα μας: **`HelloWorld.java`**
- μέσα στο **`X.java`** πρέπει να έχουμε μια κλάση με το όνομα `X`
  - `class X`
  - στο παράδειγμα μας: `class HelloWorld`
- η κλάση `X` θα πρέπει να περιέχει μια μέθοδο `main` η οποία είναι το σημείο εκκίνησης του προγράμματος μας
  - `public static void main(String[] args)`

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

- Το **όνομα** του **.java αρχείου** και το **όνομα** της **κλάσης** (που περιέχει τη μέθοδο **main**) θα πρέπει να είναι πάντα τα **ίδια**!

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java II

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

- λέξεις σε **κόκκινο** είναι **δεσμευμένες** λέξεις

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java III

Ορίζει την κλάση

Όνομα της κλάσης

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java IV

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.      public static void main(String args[])
6.      {
7.          // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.          System.out.println("Hello world!");
9.      }
10. }
```

- Τα άγκιστρα **{ … }** ορίζουν ένα λογικό **block** του κώδικα
  - αυτό μπορεί να είναι μία **κλάση**, μία **μέθοδος**, ένα **if statement**
  - οι μεταβλητές που ορίζουμε μέσα σε ένα λογικό block, έχουν **εμβέλεια** μέσα στο block

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java V

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

Ορισμός της μεθόδου main

- **public** και **static**: θα τα συζητήσουμε αργότερα

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java VI

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

Ορισμός της μεθόδου main

- τύπος επιστρεφόμενης τιμής μεθόδου
  - **void**: η συνάρτηση δεν επιστρέφει τίποτα

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java VII

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

Ορισμός της μεθόδου main

- το όνομα της μεθόδου
  - **main**: ειδική περίπτωση που σηματοδοτεί το σημείο εκκίνησης του προγράμματος

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java VIII

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

Ορισμός της μεθόδου main

- ορίσματα της μεθόδου
  - ένας πίνακας (**args**) από Strings που αντιστοιχούν στις παραμέτρους με τις οποίες τρέχουμε το πρόγραμμα

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java IX

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.     public static void main(String args[])
6.     {
7.       // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.       System.out.println("Hello world!");
9.     }
10. }
```

Ορισμός της μεθόδου main

- `String`: κλάση που χειρίζεται συμβολοσειρές
- στη Java χρειάζεται να ορίσουμε τον τύπο της κάθε μεταβλητής που χρησιμοποιούμε
  - strongly typed language

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java X

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

- /* … */ : σχόλιο πολλαπλών γραμμών
- // … : σχόλιο μίας γραμμής

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java XI

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

- κάθε εντολή στη Java τελειώνει με το **;**

Πρόγραμμα Java

1ο πρόγραμμα: "Hello World" - Αρχείο HelloWorld.java XII

```
1.  /* Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού */
2.
3.  class HelloWorld
4.  {
5.    public static void main(String args[])
6.    {
7.      // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.      System.out.println("Hello world!");
9.    }
10. }
```

το "Hello World" είναι ένα **αντικείμενο** της κλάσης **String**

αντικείμενο **System.out**: ορίζει το ρεύμα εξόδου

μέθοδος **println**: i) τυπώνει το **String αντικείμενο** που δίνεται ως όρισμα και ii) αλλάζει γραμμή

Πρόγραμμα Java
Μεταγλώττιση (compiling)

- Η μεταγλώττιση του πηγαίου κώδικα γίνεται με την εντολή **javac** ως εξής:

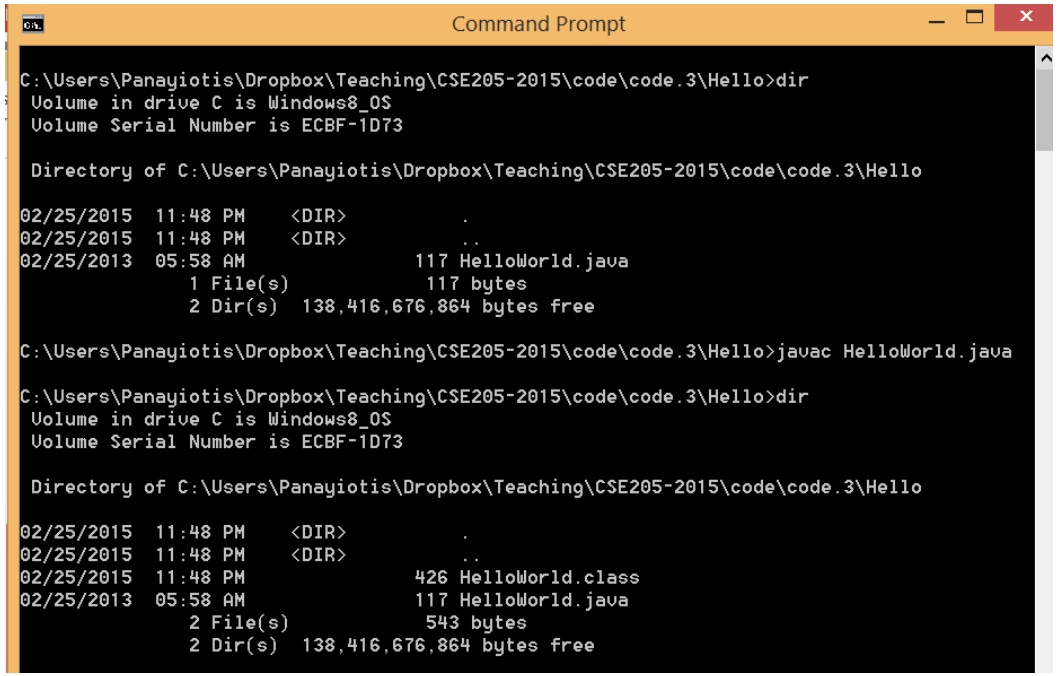
```
javac <.java αρχείο>
```

  - το αποτέλεσμα είναι η δημιουργία ενός **.class** αρχείου που περιέχει τον ενδιάμεσο κώδικα (bytecode)

παράδειγμα: Το 1ο μας πρόγραμμα: "Hello World"

```
javac HelloWorld.java
```

  - το αποτέλεσμα είναι η δημιουργία του **HelloWorld.class**



Πρόγραμμα Java
Εκτέλεση (running)

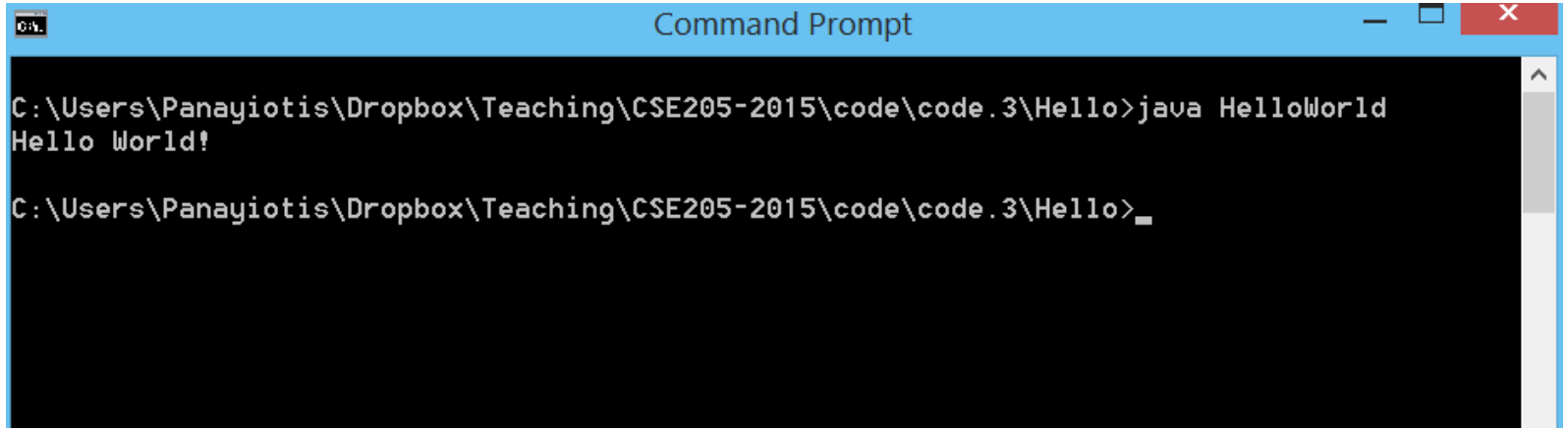
- Η εκτέλεση του κώδικα γίνεται με την εντολή **java** ως εξής:

```
java <όνομα .class αρχείο χωρίς επίθεμα>
```

παράδειγμα: Το 1ο μας πρόγραμμα: "Hello World"

```
java HelloWorld
```

χωρίς κανένα επίθεμα!



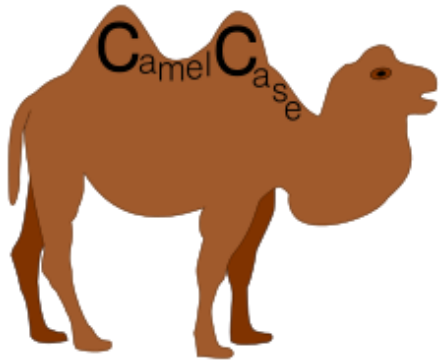
Στυλ Προγραμματισμού
Στοίχιση κώδικα

```
1.  // Το πρόγραμμα εκτυπώνει "Hello World" στο παράθυρο του τερματικού
2.
3.  class HelloWorld
4.  {
5.     public static void main(String args[])
6.     {
7.        // εκτύπωσε "Hello World" στο παράθυρο του τερματικού
8.        System.out.println("Hello world!");
9.     }
10. }
```

- οι εντολές μέσα σε ένα block του κώδικα ξεκινάνε ένα tab πιο μπροστά από το προηγούμενο
- όλες οι εντολές σε ένα block είναι στοιχισμένες
- τα άγκιστρα είναι στοιχισμένα με την εντολή που ορίζει το block

Στυλ Προγραμματισμού
Ονόματα

- τα ονόματα των κλάσεων ξεκινάνε με κεφαλαίο γράμμα
  - παράδειγμα: `HelloWorld`
- τα ονόματα των πεδίων, μεθόδων και αντικειμένων ξεκινάνε με μικρό γράμμα
- χρησιμοποιούμε ολόκληρες λέξεις (και συνδυασμούς τους) για τα ονόματα
  - ✓ δεν πειράζει αν βγαίνουν μεγάλα ονόματα
- χρησιμοποιούμε το στυλ CamelCase
  - όταν για ένα όνομα έχουμε πάνω από μία λέξη, τις συνενώνουμε και στο σημείο συνένωσης κάνουμε το πρώτο γράμμα της λέξης κεφαλαίο
  - παράδειγμα: `printName` (όχι **`print_name`**)
- χρησιμοποιούμε κεφαλαία και '_' για τις σταθερές
  - παράδειγμά: **`PI_NUMBER`**



Τύποι και τιμές

1. πρωταρχικοί τύποι
2. αποθήκευση μεταβλητών στη μνήμη του υπολογιστή
3. αναθέσεις - ασφάλεια τύπων
4. προκαθορισμένες κλάσεις

Πρόγραμμα Java

2ο Πρόγραμμα: Λόγος δύο ακεραίων

- φτιάξτε ένα πρόγραμμα που τυπώνει το λόγο δύο ακεραίων

Πρόγραμμα Java

2ο Πρόγραμμα: Λόγος δύο ακεραίων ΙΙ

```
1.  // Το πρόγραμμα εκτυπώνει το λόγο δύο ακεραίων στο παράθυρο του τερματικού
2.
3.  class Division
4.  {
5.     public static void main(String args[])
6.     {
7.        int x = 32;
8.        int y = 10;
9.        double division;
10.       division = x / (double) y;
11.       System.out.println("Result = " + division);
12.    }
13. }
```

Πρόγραμμα Java

2ο Πρόγραμμα: Λόγος δύο ακεραίων III

```
1.  // Το πρόγραμμα εκτυπώνει το λόγο δύο ακεραίων στο παράθυρο του τερματικού
2.
3.  class Division
4.  {
5.     public static void main(String args[])
6.     {
7.        int x = 32;
8.        int y = 10;
9.        double division;
10.       division = x / (double) y;
11.       System.out.println("Result = " + division);
12.    }
13. }
```

- ορισμός μεταβλητών
- η Java είναι strongly typed γλώσσα: κάθε μεταβλητή θα πρέπει να έχει έναν τύπο
- οι τύποι **int** και **double** είναι πρωταρχικοί/βασικοί τύποι (primitive types)
- εκτός από τους βασικούς τύπους, όλοι οι άλλοι τύποι είναι κλάσεις

Πρωταρχικοί/Βασικοί τύποι (primitive types)

Όνομα τύπου	Τιμή	Μέγεθος
`boolean`	true/false	1 bit
`char`	Χαρακτήρας (Unicode)	2 bytes
`byte`	Ακέραιος	1 byte
`short`	Ακέραιος	2 bytes
`int`	Ακέραιος	4 bytes
`long`	Ακέραιος	8 bytes
`float`	Πραγματικός	4 bytes
`double`	Πραγματικός	8 bytes

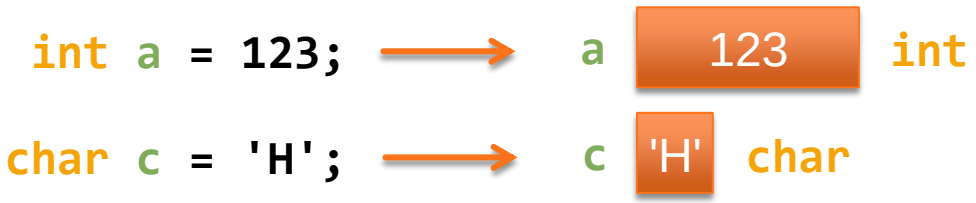
- όταν ορίζουμε μια μεταβλητή δεσμεύεται ο αντίστοιχος χώρος στη μνήμη
- το όνομα της μεταβλητής αντιστοιχίζεται με αυτό το χώρο στη μνήμη

Τύποι και Αντικείμενα

- ένας τύπος ορίζει (για ένα αντικείμενο)
  - ένα σύνολο από πιθανές τιμές και
  - ένα σύνολο πράξεων
- μία τιμή είναι ένα σύνολο από bits στη μνήμη
  - ερμηνεύονται σύμφωνα με έναν τύπο

τύπος	τιμή
`int`	`0, 1, 123, -6`
`char`	`'H', '12'`
`String`	`"Hello", "World"`

- ένα αντικείμενο είναι κάποια μνήμη που διατηρεί την τιμή ενός συγκεκριμένου τύπου
  - μία μεταβλητή είναι ένα αντικείμενο με όνομα
  - μία δήλωση είναι μία εντολή που δίνει όνομα σε ένα αντικείμενο
  - ένας ορισμός είναι μία δήλωση που δεσμεύει μνήμη για ένα αντικείμενο



Η μνήμη του υπολογιστή

η κύρια μνήμη (main memory) του υπολογιστή

- κρατάει τα δεδομένα και τις εντολές για την εκτέλεση των προγραμμάτων
  - η μνήμη είναι προσωρινή, τα περιεχόμενά της χάνονται όταν ολοκληρωθεί το πρόγραμμα
- είναι χωρισμένη σε bytes (8 bits)
  - ο χώρος που χρειάζεται για ένα χαρακτήρα ASCII
  - το κάθε byte έχει μια διεύθυνση, με την οποία μπορούμε να προσπελάσουμε τη συγκεκριμένη θέση μνήμης
    - Random Access Memory (RAM)
    - σε 32-bit συστήματα μια διεύθυνση είναι 32 bits
    - σε 64-bit συστήματα μια διεύθυνση είναι 64 bits

Διεύθυνση μνήμης (παρουσιάζονται μόνο τα 4 λιγότερο σημαντικά από τα 32 ή 64 bits)	Περιεχόμενο μνήμης
0000	a
0001	b
0010	c
0011	d
0100	e
0101	f
0110	g
0111	h

Η μνήμη του υπολογιστή
Αποθήκευση μεταβλητών - Θέσεις μνήμης

- η κύρια μνήμη (main memory) του υπολογιστή κρατάει τις μεταβλητές ενός προγράμματος
- μια μεταβλητή μπορεί να απαιτεί χώρο περισσότερο από 1 byte
  - π.χ., οι μεταβλητές τύπου **`double`** χρειάζονται **8 bytes**
  - ☞ η μεταβλητή τότε αποθηκεύεται σε συνεχόμενα bytes στη μνήμη
- η **διεύθυνση** της μεταβλητής θεωρείται το **πρώτο byte** από το οποίο ξεκινάει η αποθήκευση του της μεταβλητής
  - στο παράδειγμα μας η μεταβλητή βρίσκεται στη διεύθυνση **0000**
  - αν ξέρουμε τη διεύθυνση και το μέγεθος της μεταβλητής μπορούμε να τη διαβάσουμε
- επομένως, η θέση μνήμης μιας μεταβλητής αποτελείται από μία διεύθυνση και ένα μέγεθος

Διεύθυνση μνήμης (παρουσιάζονται μόνο τα 4 λιγότερο σημαντικά από τα 32 ή 64 bits)	Περιεχόμενο μνήμης
0000	8.5
0001	
0010	
0011	
0100	
0101	
0110	
0111	

Η μνήμη του υπολογιστή
Αποθήκευση μεταβλητών - Πρωταρχικοί τύποι

```
int x = 5;
int y = 3;
```

- το μέγεθος μνήμης που απαιτούν οι μεταβλητές πρωταρχικού τύπου (`char`, `int`, `double`, ...) είναι προκαθορισμένο
  - π.χ. 4 bytes για έναν ακέραιο (`int`)

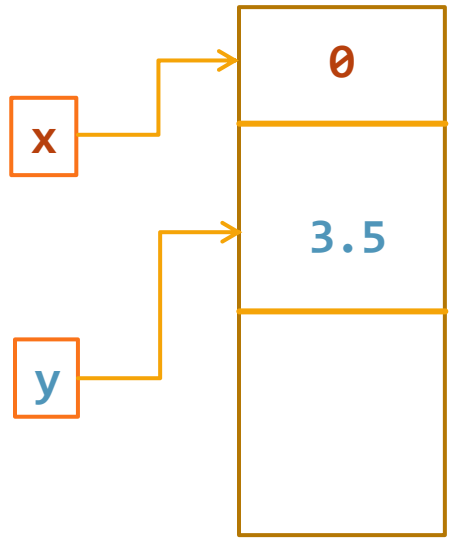
- όταν ο μεταγλωττιστής δει τη δήλωση μιας μεταβλητής πρωταρχικού τύπου δεσμεύει μια θέση μνήμης αντίστοιχου μεγέθους
  - η δήλωση μιας μεταβλητής → ουσιαστικά δίνει ένα όνομα σε μία θέση μνήμης
  - η δήλωση και ο ορισμός μιας μεταβλητής, δε διαχωρίζονται (όπως στη C/C++)
  - συχνά λέμε η θέση μνήμης `x` για τη μεταβλητή `x`

Διεύθυνση μνήμης (παρουσιάζονται μόνο τα 4 λιγότερο σημαντικά από τα 32 ή 64 bits)	Περιεχόμενο μνήμης
0000	5
0001	
0010	
0011	
0100	3
0101	
0110	
0111	

Η μνήμη του υπολογιστή
Αποθήκευση μεταβλητών

- μπορούμε να σκεφτόμαστε την μνήμη του υπολογιστή σαν μια σειρά από «κουτάκια» διαφόρων μεγεθών στα οποία μπορούμε να αποθηκεύουμε δεδομένα
  - το κάθε κουτάκι έχει διεύθυνση και περιεχόμενα
- όταν ορίζουμε μια μεταβλητή πρωταρχικού τύπου αυτό που γίνεται είναι ότι:
  1. δεσμεύουμε ένα κουτάκι κατάλληλου μεγέθους
     - 4 bytes για ένα `int`
  2. δίνουμε στο κουτάκι το όνομα της μεταβλητής
     - γι' αυτό η μεταβλητή ορίζεται μόνο μια φορά
- με την ανάθεση αλλάζουμε τα περιεχόμενα του κουτιού
- αν δεν αρχικοποιήσουμε μια μεταβλητή η Java την αρχικοποιεί στο μηδέν
  - ή την αντίστοιχη αρχική τιμή για τις μη αριθμητικές μεταβλητές

```
int x;
double y = 3.5;
```



Πρόγραμμα Java

2ο Πρόγραμμα: Λόγος δύο ακεραίων IV

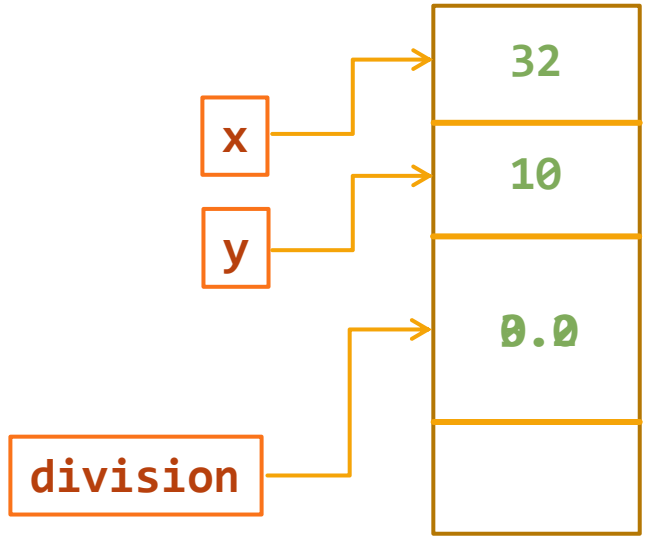
```
1.  // Το πρόγραμμα εκτυπώνει το λόγο δύο ακεραίων στο παράθυρο του τερματικού
2.
3.  class Division
4.  {
5.     public static void main(String args[])
6.     {
7.        int x = 32;
8.        int y = 10;
9.        double division;
10.       division = x / (double) y;
11.       System.out.println("Result = " + division);
12.    }
13. }
```

- ανάθεση (assignment)
  1. αποτίμηση της τιμής της έκφρασης στο δεξιό μέλος του "=" και
  2. μετά ανάθεση της τιμής στην μεταβλητή στο αριστερό μέλος

Πρόγραμμα Java

2ο Πρόγραμμα: Λόγος δύο ακεραίων V

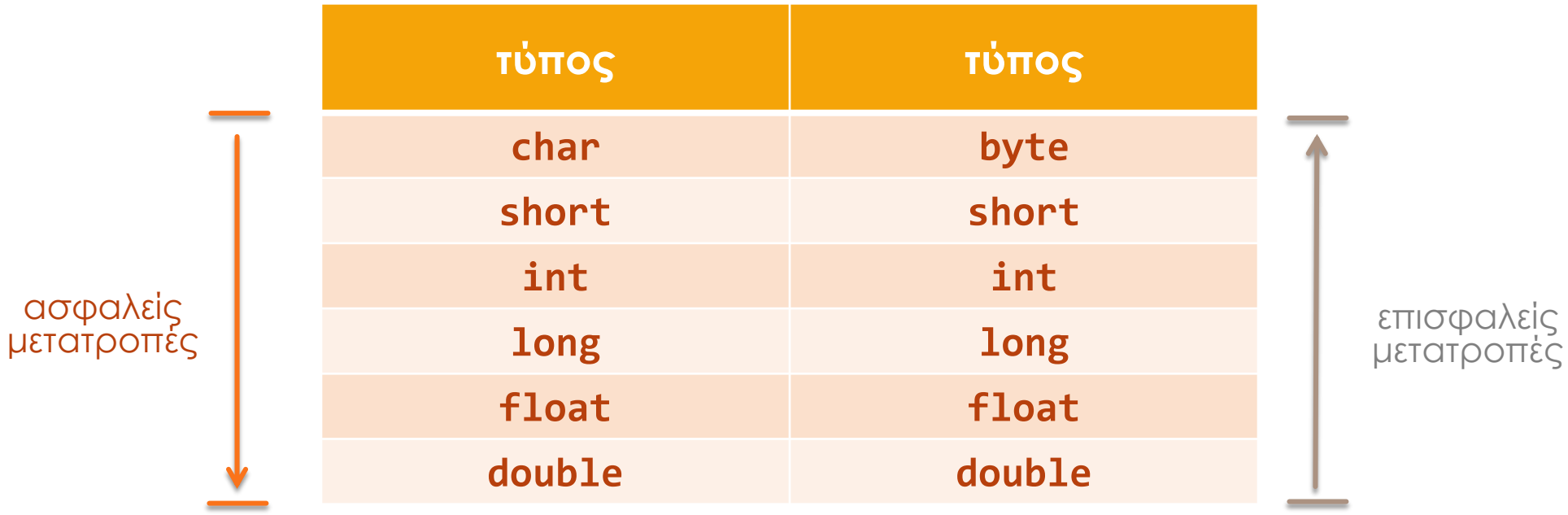
```
1.  // Το πρόγραμμα εκτυπώνει το λόγο δύο ακεραίων
στο παράθυρο του τερματικού
2.
3.  class Division
4.  {
5.     public static void main(String args[])
6.     {
7.        int x = 32;
8.        int y = 10;
9.        double division;
10.       division = x / (double) y;
11.       System.out.println("Result = " + division);
12.    }
13. }
```



- ανάθεση (assignment)
  1. διαβάζουμε τα περιεχόμενα των μεταβλητών x και y
  2. κάνουμε τον υπολογισμό
  3. αλλάζουμε τα περιεχόμενα της μεταβλητής division (αποθηκεύοντας το αποτέλεσμα της διαίρεσης)

Μεταβλητές
Αναθέσεις

- στην ανάθεση κατά κανόνα, η τιμή του δεξιού μέρους θα πρέπει να είναι ίδιου τύπου με την μεταβλητή του αριστερού μέρους
  - υπάρχουν εξαιρέσεις όταν υπάρχει συμβατότητα μεταξύ τύπων



τύπος	τύπος
char	byte
short	short
int	int
long	long
float	float
double	double

Μεταβλητές
Μία τεχνική λεπτομέρεια

- στη μνήμα τα πάντα είναι απλά bits
- ο τύπος είναι αυτός που δίνει νόημα στα bits

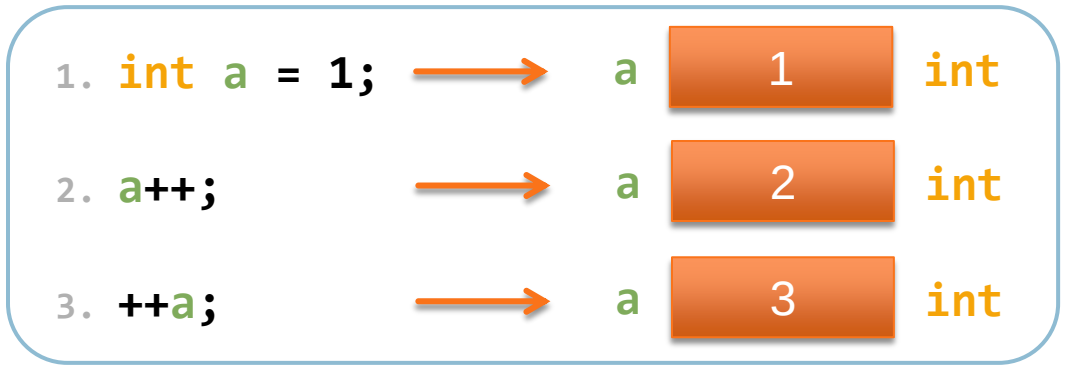
bits / δυαδικό	ακέραιος (int)	χαρακτήρας (char)
01111000	120	'x'
01100001	97	'a'
01000001	65	'A'
00110000	48	'0'

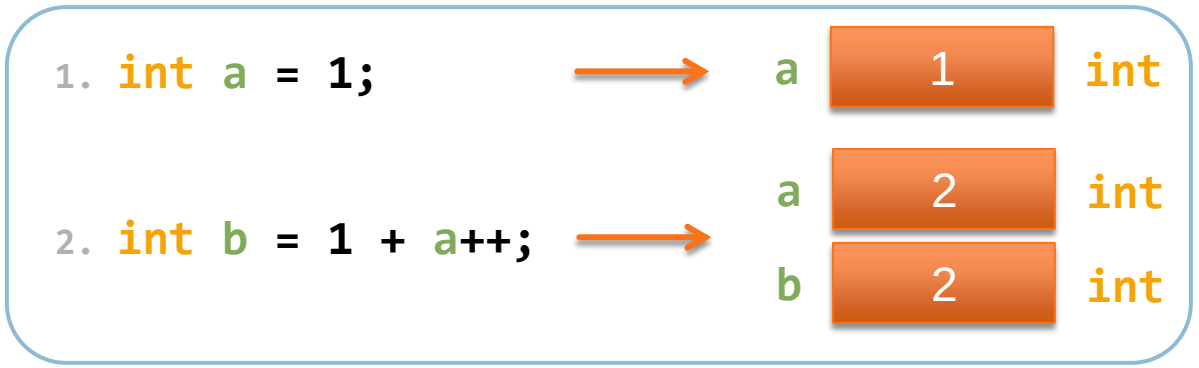
```
1.      ...
2.      char c = 'x';
3.      System.out.println(c);      // εκτύπωσε την τιμή του χαρακτήρα c, η οποία είναι 'x'
4.      int i = c;
5.      System.out.println(i);      // εκτύπωσε την ακέραια τιμή του χαρακτήρα c, η οποία είναι 120
6.      ...
```

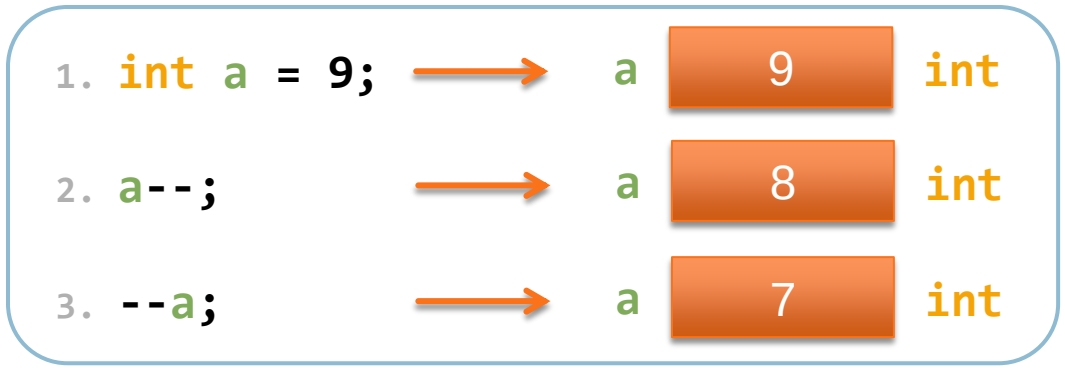
- ☞ Αυτό ισχύει και στον «αληθινό κόσμο»!
  - ? τι σημαίνει 42; → δε γνωρίζουμε μέχρι να μάθουμε τη μονάδα μέτρησης που χρησιμοποιείται
    - ? μέτρα; πόδια; μοίρες; βαθμοί κελσίου; το νούμερο μίας οδού;

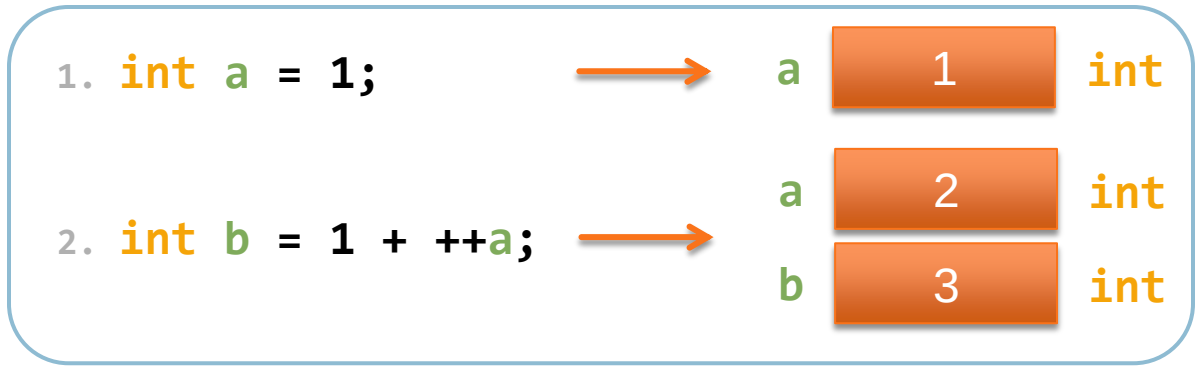
Μεταβλητές

Σύνθετοι τελεστές εκχώρησης









- `a++;` ⟷ `++a;` ⟷ `a = a+1;` και `a--;` ⟷ `--a;` ⟷ `a = a-1;`
- γενικά, για ένα δυαδικό τελεστή op ισχύει: `a op= b` ⟷ `a = a op b`
  - ☞ `a+=b`, `a-=b`, `a*=b`, `a/=b`, `a%=b`

Πρόγραμμα Java

2ο Πρόγραμμα: Λόγος δύο ακεραίων IV

```
1.  // Το πρόγραμμα εκτυπώνει το λόγο δύο ακεραίων στο παράθυρο του τερματικού
2.
3.  class Division
4.  {
5.     public static void main(String args[])
6.     {
7.        int x = 32;
8.        int y = 10;
9.        double division;
10.       division = x / (double) y;
11.       System.out.println("Result = " + division);
12.    }
13. }
```

- μετατροπή τύπου (type casting)
  - **(double) y** → μετατρέπει την τιμή της μεταβλητής **y** σε **double**
  - αν δεν γίνει η μετατροπή: η διαίρεση μεταξύ ακεραίων μας δίνει πάντα ακέραιο
    - π.χ. το **32/10** δίνει αποτέλεσμα **3** και όχι **3.2**! *(ενώ **32/(double)10** → **3.2**)*

Πρόγραμμα Java

2ο Πρόγραμμα: Λόγος δύο ακεραίων V

```
1.  // Το πρόγραμμα εκτυπώνει το λόγο δύο ακεραίων στο παράθυρο του τερματικού
2.
3.  class Division
4.  {
5.     public static void main(String args[])
6.     {
7.        int x = 32;
8.        int y = 10;
9.        double division;
10.       division = x / (double) y;
11.       System.out.println("Result = " + division);
12.    }
13. }
```

- ο τελεστής "+" μεταξύ αντικείμενων της κλάσης **String** συνενώνει (concatenates) τα δύο **String**
- μεταξύ ενός **String** και ενός βασικού τύπου, ο βασικός τύπος μετατρέπεται σε **String** και γίνεται η συνένωση

Προκαθορισμένες κλάσεις
String

- η κλάση `String` είναι προκαθορισμένη κλάση της Java που μας επιτρέπει να χειριζόμαστε συμβολοσειρές
  - ο τελεστής "+" επιτρέπει τη συνένωση
- μέθοδοι της κλάσης `String`:
  - `length()`: μήκος της συμβολοσειράς
  - `equals(String x)`: ελέγχει για ισότητα του αντικειμένου που κάλεσε τη μέθοδο και του ορίσματος `x`
  - `trim()`: αφαιρεί κενά στην αρχή και το τέλος της συμβολοσειράς
  - `split(char delim)`: σπάει το string σε πίνακα από `Strings` με βάση το χαρακτήρα `delim`
  - μέθοδοι για να αναζητηθεί μία συμβολοσειρά μέσα σε μία άλλη
  - ...

```
String s1 = "Hello World";
int length = s1.length();
```

Είσοδος - Έξοδος

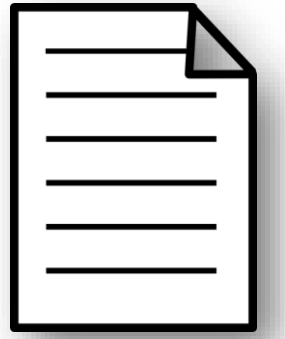
Βασικά ρεύματα Εισόδου/Εξόδου

Ρεύματα Εισόδου/Εξόδου

Ορισμός

μία αφαίρεση που αναπαριστά

1. μια πηγή, για την είσοδο χαρακτήρων
   - πληκτρολόγιο
   - αρχείο



2. ένα προορισμό, για την έξοδο χαρακτήρων
   - οθόνη
   - αρχείο

- όταν δημιουργούμε το ρεύμα το συνδέουμε με την ανάλογη πηγή ή προορισμό

Ρεύματα Εισόδου/Εξόδου

Βασικά ρεύματα

- τα βασικά ρεύματα εισόδου/εξόδου είναι έτοιμα αντικείμενα τα οποία ορίζονται σαν πεδία (στατικά) της κλάσης `System`
  - `System.out`
  - `System.in`
  - `System.err`

- μέσω αυτών (και άλλων βοηθητικών αντικειμένων) γίνεται η είσοδος και έξοδος δεδομένων ενός προγράμματος
  - μια εντολή εισόδου έχει ως αποτέλεσμα το λειτουργικό σύστημα να λάβει χαρακτήρες από την αντίστοιχη πηγή
  - μια εντολή εξόδου έχει ως αποτέλεσμα το λειτουργικό σύστημα να στείλει χαρακτήρες προς τον αντίστοιχο προορισμό

Ρεύματα Εισόδου/Εξόδου

Βασικά ρεύματα - Έξοδος

το βασικό ρεύμα εξόδου `System.out` υποστηρίζει τις μεθόδους:

1. `println(String s)`: τυπώνει τη συμβολοσειρά `s`, ακολουθούμενη από τον χαρακτήρα `'\n'` (αλλαγή γραμμής)
   - ισοδύναμα: τυπώνει τη συμβολοσειρά `s` και αλλάζει γραμμή
2. `print(String s)`: τυπώνει τη συμβολοσειρά `s` χωρίς να αλλάξει γραμμή
3. `printf`: μορφοποιημένη έξοδος
   - `printf("%d",myInt);  // τυπώνει ένα ακέραιο αριθμό`
   - `printf("%f",myDouble);  // τυπώνει έναν πραγματικό αριθμό`
   - `printf("%.2f",myDouble);  // τυπώνει έναν πραγματικό αριθμό με δύο δεκαδικά ψηφία`

Ρεύματα Εισόδου/Εξόδου

Βασικά ρεύματα - Έξοδος - Ειδικοί χαρακτήρες (escape sequences)

για να τυπώσουμε ειδικούς χαρακτήρες

- π.χ., τον χαρακτήρα `"`

χρησιμοποιούμε τον χαρακτήρα `\` και μετά τον χαρακτήρα που θέλουμε να τυπώσουμε

- π.χ., ακολουθία `\"`

ειδικός χαρακτήρας	αποτέλεσμα
`\b`	backspace
`\t`	tab
`\n`	new line
`\f`	form feed
`\r`	carriage return
`\"`	double quote
`\'`	single quote
`\\`	backslash
`\ddd`	octal code
`\uXXXX`	Hex-demical code

Ρεύματα Εισόδου/Εξόδου

Βασικά ρεύματα - Έξοδος - Παράδειγμα

```
1.  // Δοκιμή κλάσης String και εκτύπωσης ειδικών χαρακτήρων
2.
3.  class StringTest
4.  {
5.     public static void main(String args[])
6.     {
7.        String s = "Hello World";
8.        String h = "Hello";
9.        String w = "World";
10.       System.out.println(s);                  // εκτυπώνει: Hello World
11.       System.out.println(h + "\t" + w);       // εκτυπώνει: Hello     World
12.       System.out.println(s.length());         // εκτυπώνει: 11
13.       System.out.println("hello2".length()); // εκτυπώνει: 6
14.    }
15. }
```

Ρεύματα Εισόδου/Εξόδου

Βασικά ρεύματα - Είσοδος

- χρησιμοποιούμε την κλάση `Scanner` της Java

```
import java.util.Scanner;
```

η οποία προσφέρει έναν απλό σαρωτή κειμένου που μπορεί να διαβάζει πρωταρχικούς τύπους και συμβολοσειρές

- αρχικοποιείται με το ρεύμα εισόδου, π.χ. το πληκτρολόγιο:

```
Scanner in = new Scanner(System.in);
```

  - εντολή `new`: δημιουργεί ένα αντικείμενο τύπου `Scanner` με το οποίο μπορούμε πλέον να διαβάζουμε από την είσοδο
  - το αντικείμενο αυτό αναπαριστά το πληκτρολόγιο, στο παράδειγμά μας
  - το ίδιο αντικείμενο χρησιμοποιείται για να διαβάσουμε πολλαπλές τιμές

Ρεύματα Εισόδου/Εξόδου

Βασικά ρεύματα - Είσοδος - Scanner

- μπορούμε να καλέσουμε μεθόδους της `Scanner` για να διαβάσουμε κάτι από την είσοδο
  - `nextLine()`: διαβάζει μέχρι να βρει το χαρακτήρα `'\n'`
  - `next()`: διαβάζει την επόμενη συμβολοσειρά μέχρι να βρει λευκό χαρακτήρα (whitespace)
  - `nextInt()`: διαβάζει τον επόμενο `int`
  - `nextDouble()`: διαβάζει τον επόμενο `double`
  - `nextBoolean()`: διαβάζει τον επόμενο `boolean`

λευκοί χαρακτήρες (whitespaces)	
`' '`	space
`'\t'`	tab
`'\n'`	new line
`'\r'`	carriage return

- μπορείτε να ελέγξετε εάν ένας χαρακτήρας `c` είναι λευκός χρησιμοποιώντας τη μέθοδο:

```
Character.isWhitespace(c)
```

η οποία επιστρέφει μία boolean τιμή (`true/false`)

Ρεύματα Εισόδου/Εξόδου

Βασικά ρεύματα - 1ο παράδειγμα

```
1.  // δοκιμή ρευμάτων εισόδου/εξόδου: 1ο παράδειγμα - ανάγνωση γραμμής χαρακτήρων
2.
3.  import java.util.Scanner;    /* φέρνουμε την κλάση Scanner μέσα στο πρόγραμμα μας ώστε
4.                               να μπορούμε να φτιάξουμε αντικείμενα τύπου Scanner */
5.  class TestIO1
6.  {
7.     public static void main(String args[])
8.     {
9.        System.out.println("Say something:");      // εκτυπώνει στην οθόνη: Say something
10.       Scanner input = new Scanner(System.in);    // αρχικοποίηση ρεύματος εισόδου (πληκτρολόγιο)
11.       String line = input.nextLine();            // διαβάζει τη γραμμή που έδωσε ο χρήστης
12.       System.out.println(line);                  // εκτυπώνει τη γραμμή
13.    }
14. }
```

Ρεύματα Εισόδου/Εξόδου

Βασικά ρεύματα - 2ο παράδειγμα

```
1.  // δοκιμή ρευμάτων εισόδου/εξόδου: 2ο παράδειγμα - ανάγνωση πραγματικού αριθμού
2.
3.  import java.util.Scanner;    /* φέρνουμε την κλάση Scanner μέσα στο πρόγραμμα μας ώστε
4.                               να μπορούμε να φτιάξουμε αντικείμενα τύπου Scanner */
5.  class TestIO2
6.  {
7.     public static void main(String args[])
8.     {
9.        System.out.println("Give a real number:");   // εκτυπώνει στην οθόνη: Give a real number
10.       Scanner input = new Scanner(System.in);      // αρχικοποίηση ρεύματος εισόδου (πληκτρολόγιο)
11.       double d = input.nextDouble();               /* διαβάζει τον ακέραιο που έδωσε ο χρήστης και
12.                                                    έστω ότι ο χρήστης δίνει το 8 */
13.       System.out.println("division by 4 = " + d/4);          // εκτυπώνει: division by 4 = 2
14.       System.out.println("1 + (division by 4) = " + 1 + d/4);   // εκτυπώνει: 1 + division by 4 = 12
15.       System.out.println("1 + (division by 4) = " + (1 + d/4));  // εκτυπώνει: 1 + division by 4 = 3
16.                                                    // εκτυπώνει: 1 + (division of 8.00 by 4) = 3.00
17.       System.out.printf("1 + (division of %.2f by 4) = %.2f", d, 1 + d/4);
18.    }
19. }
```

Το + λειτουργεί ως τελεστής συνένωσης Strings → άρα οι αριθμοί μετατρέπονται σε συμβολοσειρές

οι παρενθέσεις ορίζουν ότι το '+' θα εκτελέσει την αριθμητική πράξη

Απλός Υπολογισμός

Απλός Υπολογισμός

1ο Πρόγραμμα: Μετατροπή ιντσών (inch) σε εκατοστόμετρα (cm)

φτιάξτε ένα πρόγραμμα που

1. ζητά από το χρήστη μία τιμή σε ίντσες
2. τη μετατρέπει σε εκατοστόμετρα
3. την εκτυπώνει στην οθόνη

✍ σημείωση: μία ίντσα είναι 2.54 εκατοστόμετρα

Απλός Υπολογισμός

1ο Παράδειγμα

```
1.  // μετατροπή ιντσών (inch) σε εκατοστρόμετρα (cm)
2.
3.  import java.util.Scanner;   /* φέρνουμε την κλάση Scanner μέσα στο πρόγραμμα μας ώστε
4.                              να μπορούμε να φτιάξουμε αντικείμενα τύπου Scanner */
5.  class InchToCmConverter
6.  {
7.     public static void main(String args[])
8.     {
9.        final double cmPerInch = 2.54;                                  // πλήθος cm ανά inch
10.       System.out.println("Please give a distance value in inch: ");
11.       Scanner input = new Scanner(System.in);
12.       double dist = input.nextDouble();
13.       System.out.println(dist + "in = " + dist * cmPerInch + "cm");
14.    }
15. }
```

- η χρήση του `final` δηλώνει ότι η τιμή της μεταβλητής δε μπορεί να αλλάξει

- *πώς μπορώ να ζητάω επαναλαμβανόμενα τιμές απόστασης (και να τις μετατρέπω σε εκατοστόμετρα), με μία εκτέλεση του προγράμματος;*

Απλός Υπολογισμός

2ο Παράδειγμα

```
1.  // επαναληπτική μετατροπή ιντσών (inch) σε εκατοστρόμετρα (cm)
2.
3.  import java.util.Scanner;       /* φέρνουμε την κλάση Scanner μέσα στο πρόγραμμα μας ώστε
4.                                  να μπορούμε να φτιάξουμε αντικείμενα τύπου Scanner */
5.  class InchToCmConverter
6.  {
7.     public static void main(String args[])
8.     {
9.        final double cmPerInch = 2.54;
10.       Scanner input = new Scanner(System.in);
11.       double dist = 1;
12.       while (dist != 0) {
13.              System.out.println("Please give a distance value in inch (give 0 to terminate): ");
14.              dist = input.nextDouble();
15.              System.out.println(dist + "in = " + dist * cmPerInch + "cm");
16.       }
17.    }
18. }
```

- οι εντολές που περιέχονται στο σύνολο εντολών της `while`, εκτελούνται επαναληπτικά έως ότου η συνθήκη της `while` αποτιμηθεί ως ψευδής

Τι είναι τελικά ο προγραμματισμός;

Τι είναι τελικά ο προγραμματισμός;

- Συμβατικοί ορισμοί
  - λέγοντας σε έναν πολύ γρήγορο βλάκα ακριβώς τι να κάνει
  - ένα σχέδιο για την επίλυση ενός προβλήματος σε έναν υπολογιστή
  - καθορισμός της σειράς εκτέλεσης ενός προγράμματος

- Ορισμός από άλλο τομέα (ακαδημαϊκός χώρος)
  - ένα πρόγραμμα είναι μια οργανωμένη και κατευθυνόμενη συσσώρευση πόρων για την επίτευξη συγκεκριμένων στόχων

Τι είναι τελικά ο προγραμματισμός; II

Ο ορισμός που θα χρησιμοποιήσουμε:

1. καθορισμός της δομής και της συμπεριφοράς ενός προγράμματος, και
2. δοκιμή ότι το πρόγραμμα εκτελεί την εργασία του σωστά και με αποδεκτή απόδοση
   - ποτέ μην ξεχάσετε να ελέγξετε ότι λειτουργεί

- Λογισμικό == ένα ή περισσότερα προγράμματα

Προγραμματισμός

- Ο προγραμματισμός είναι θεμελιωδώς απλός
  - απλά δηλώστε τι πρέπει να κάνει η μηχανή
- Τότε γιατί είναι δύσκολος ο προγραμματισμός;
  1. θέλουμε "η μηχανή" να κάνει περίπλοκα πράγματα
     - και οι υπολογιστές είναι νευρικά, ανελέητα, χαζά θηρία
  2. ο κόσμος είναι πιο περίπλοκος από όσο θα θέλαμε να πιστέψουμε
     - έτσι δεν γνωρίζουμε πάντα τις συνέπειες του τι θέλουμε
  3. «ο προγραμματισμός είναι κατανόηση»
     - όταν μπορείτε να προγραμματίσετε μια εργασία, την καταλαβαίνετε
     - όταν προγραμματίζετε, περνάτε σημαντικό χρόνο προσπαθώντας να κατανοήσετε την εργασία που θέλετε να αυτοματοποιήσετε
  4. ο προγραμματισμός είναι τόσο πράξη όσο και θεωρία
     - αν είστε απλώς πρακτικοί →παράγετε μη κλιμακούμενα, μη διατηρήσιμα προγράμματα
     - αν είστε μόνο θεωρητικοί → παράγετε παιχνίδια

Σύνοψη

- “Hello World”
- Μεταγλώττιση, εκτέλεση
- Τύποι και τιμές
  - πρωταρχικοί τύποι
  - αποθήκευση μεταβλητών στη μνήμη του υπολογιστή
  - αναθέσεις - ασφάλεια μετατροπών
  - προκαθορισμένες κλάσεις
- Είσοδος/έξοδος
  - βασικές ροές
  - ειδικοί χαρακτήρες
  - λευκοί χαρακτήρες
- Τι είναι τελικά ο προγραμματισμός;