2. ΕΙΣΑΓΩΓΗ ΣΤΗ JAVA

Η γλώσσα προγραμματισμού Java

Βασικό συντακτικό, ορισμός μεταβλητών, έλεγχος ροής

ΜΕΤΑΒΛΗΤΕΣ - ΑΝΑΘΕΣΕΙΣ

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 δεδομένα χάνονται όταν ολοκληρωθεί το πρόγραμμα.
- Η μνήμη 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	Περιεχόμενο μνήμης
0000	‘a’
0001	‘b’
0010	‘c’
0011	‘d’
0100	‘e’
0101	‘f’
0110	‘g’
0111	‘h’

Αποθήκευση μεταβλητών

- 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 θέση μνήμης (διεύθυνση) της μεταβλητής θεωρείται το πρώτο byte από το οποίο ξεκινάει η αποθήκευση του της μεταβλητής.
  - Στο παράδειγμα μας η μεταβλητή βρίσκεται στη θέση 0000
  - Αν ξέρουμε την αρχή και το μέγεθος της μεταβλητής μπορούμε να τη διαβάσουμε.
- Άρα μία θέση μνήμης αποτελείται από μία διεύθυνση και το μέγεθος.

Διεύθυνση μνήμης	Περιεχόμενο μνήμης
0000	8.5
0001	
0010	
0011	
0100	
0101	
0110	
0111	

Αποθήκευση μεταβλητών πρωταρχικού τύπου

- Για τις μεταβλητές πρωταρχικού τύπου (char, int, double,…) ξέρουμε εκ των προτέρων το μέγεθος της μνήμης που χρειαζόμαστε.
- 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
  - Συχνά λέμε η θέση μνήμης x για τη μεταβλητή x.

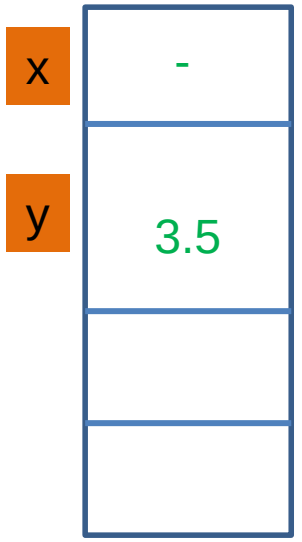
```
int x = 5;
int y = 3;
```

	Διεύθυνση μνήμης	Περιεχόμενο μνήμης
x	0000	5
	0001	
	0010	
	0011	
y	0100	3
	0101	
	0110	
	0111	

Αποθήκευση μεταβλητών

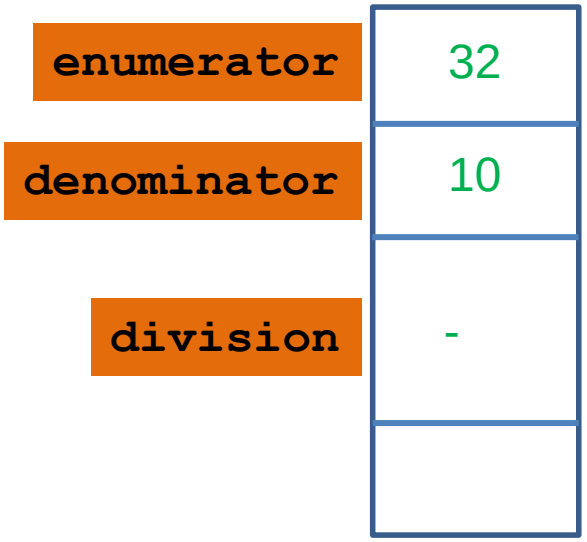
- Μπορούμε να σκεφτόμαστε την μνήμη του υπολογιστή σαν μια σειρά από «κουτάκια» διαφόρων μεγεθών στα οποία μπορούμε να αποθηκεύουμε δεδομένα
  - Το κάθε κουτάκι έχει διεύθυνση και περιεχόμενα
- Όταν ορίζουμε μια μεταβλητή πρωταρχικού τύπου (π.χ., `int x`) αυτό που γίνεται είναι ότι:
  - Δεσμεύουμε ένα κουτάκι κατάλληλου μεγέθους
    - 4 bytes για ένα int
  - Δίνουμε στο κουτάκι το όνομα της μεταβλητής
    - Το κουτάκι `x` αντί για το κουτάκι `0110`
- Γι αυτό και η μεταβλητή ορίζεται μόνο μια φορά.
- Με την ανάθεση αλλάζουμε τα περιεχόμενα του κουτιού
- Αν δεν αρχικοποιήσουμε μια μεταβλητή δεσμεύουμε το κουτί αλλά δεν έχει τιμή
  - Εξαίρεση: Όταν έχουμε πεδία.

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



Αναθέσεις

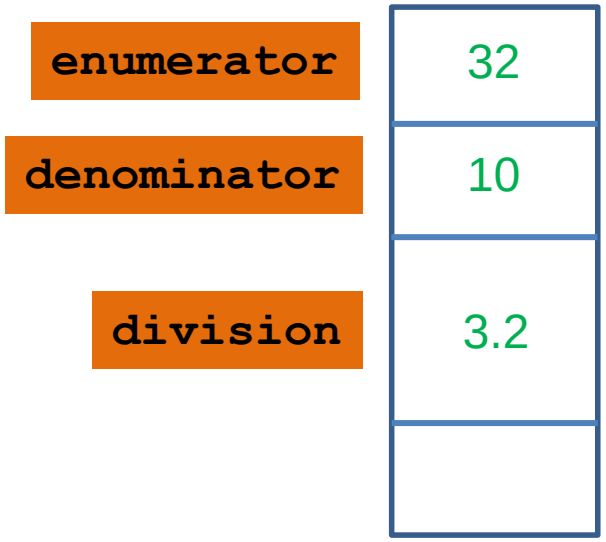
```
class Division
{
  public static void main(String args[])
  {
     int enumerator = 32;
     int denominator = 10;
     double division;
     division =
         enumerator/(double)denominator;
     System.out.println(division);
  }
}
```



Οι μεταβλητές **enumerator** και **denominator** αρχικοποιούνται στις τιμές τους. Η μεταβλητή **division** δεν έχει κάποια αρχική τιμή.

Αναθέσεις

```
class Division
{
  public static void main(String args[])
  {
    int enumerator = 32;
    int denominator = 10;
    double division;
    division =
        enumerator/(double)denominator;
    System.out.println(division);
  }
}
```



Ανάθεση: Διαβάζουμε τα περιεχόμενα των μεταβλητών **enumerator** και **denominator** κάνουμε τον υπολογισμό και αλλάζουμε τα περιεχόμενα της μεταβλητής **division** αποθηκεύοντας το αποτέλεσμα της διαίρεσης.

Division-Err4.java

```
class Division
{
   public static void main(String args[])
   {
      int enumerator;
      int denominator = 10;
      double division = enumerator/denominator;
      System.out.println(division);
   }
}
```

ΛΑΘΟΣ!

Η μεταβλητή enumerator ορίζεται αλλά δεν αρχικοποιείται

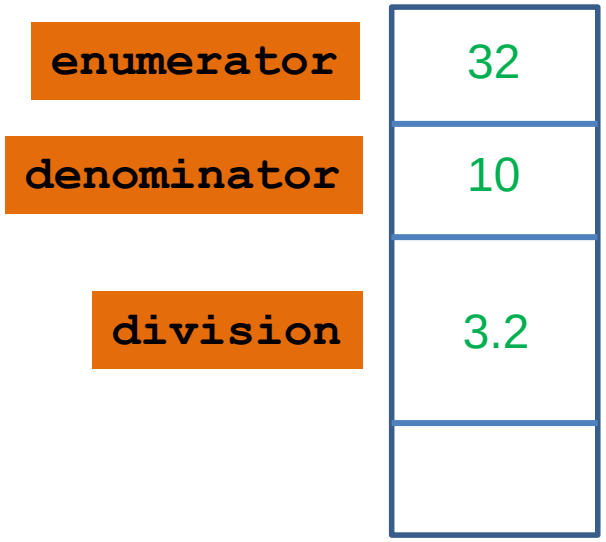
Χρήση μη αρχικοποιημένης μεταβλητής

```
3>javac Division-Err4.java
Division-Err4.java:7: error: variable enumerator might not
have been initialized
                  double division = enumerator/denominator;
                                    ^
1 error
```

Οι μεταβλητές πρέπει να αρχικοποιηθούν πριν χρησιμοποιηθούν.
Εξαίρεση: Τα πεδία των κλάσεων

Αναθέσεις

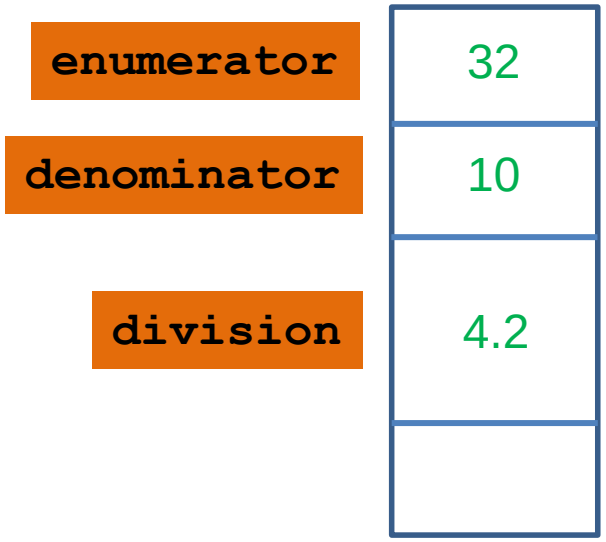
```
class Division
{
  public static void main(String args[])
  {
    int enumerator = 32;
    int denominator = 10;
    double division =
        enumerator/(double)denominator;
    division = division + 1;
    System.out.println(division);
  }
}
```



Μετά τις τρεις πρώτες εντολές η μεταβλητή division έχει την τιμή 3.2

Αναθέσεις

```
class Division
{
  public static void main(String args[])
  {
    int enumerator = 32;
    int denominator = 10;
    double division =
        enumerator/(double)denominator;
    division = division + 1;
    System.out.println(division);
  }
}
```



Ανάθεση division = division + 1;

Πρώτα υπολογίζεται το δεξί μέρος: Διαβάζουμε τα περιεχόμενα της μεταβλητής division και προσθέτουμε +1 και παίρνουμε την τιμή 4.2

Μετά γινεται η εκχώρηση της τιμής 4.2 στη θέση μνήμης division, αλλάζοντας τα περιεχόμενα της μεταβλητής

STRINGS

Strings

- Η κλάση String είναι προκαθορισμένη κλάση της Java που μας επιτρέπει να χειριζόμαστε αλφαριθμητικά.
  - Μια μεταβλητή τύπου String είναι ένα αντικείμενο της κλάσης String
  - Και οι σταθερές είναι αντικείμενα της κλάσης String.
- Η χρήση των αντικειμένων γίνεται σχεδόν πάντα με την κλήση μεθόδων, σε αντίθεση με τους πρωταρχικούς τύπους που μπορούμε να κάνουμε πράξεις.
- Για τα Strings η Java μας δίνει περισσότερη ευελιξία. Μπορούμε να χρησιμοποιήσουμε τον τελεστή "+" για να κάνουμε συνένωση Strings
  - Το αποτέλεσμα της συνένωσης είναι ένα νέο String.
- Υπάρχουν πολλές χρήσιμες μέθοδοι της κλάσης String που θα δούμε αργότερα. Για παράδειγμα:
  - length(): επιστρέφει το μήκος του String
- Μία πολύ χρήσιμη μέθοδος:
  - equals(String x): ελέγχει για ισότητα του String που κάλεσε την μέθοδο και του String x που είναι το όρισμα και επιστρέφει true ή false ανάλογα.

Παράδειγμα Division3.java

```
class Division
{
  public static void main(String args[])
  {
    int enumerator = 32;
    int denominator = 10;
    double division;
    division = enumerator/(double)denominator;
    System.out.println("Result = " + division);
  }
}
```

String concatenation

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

Κλήση μεθόδων

- Ένα αντικείμενο μιας κλάσης μπορεί να καλέσει μεθόδους της κλάσης που επιτελούν μια λειτουργία.
- Η κλήση της μεθόδου γίνεται με τον τελεστή της τελείας .
- Συντακτικό

```
<όνομα μεταβλητής-αντικείμενο>.<όνομα μεθόδου>([ορίσματα]);
```

Παράδειγμα: TestString1.java

```
class StringTest
{
   public static void main(String args[])
   {
      String h = "hello";
      System.out.println(h.length());
      System.out.println("hello".length());
      System.out.println(h.equals("hello"));
   }
}
```

Escape sequences

- Για να τυπώσουμε κάποιους ειδικούς χαρακτήρες (π.χ., τον χαρακτήρα “) χρησιμοποιούμε τον χαρακτήρα \ και μετά τον χαρακτήρα που θέλουμε να τυπώσουμε
  - Π.χ., ακολουθία \”
- Αυτό ισχύει γενικά για ειδικούς χαρακτήρες.

\b	Backspace
\t	Tab
\n	New line
\f	Form feed
\r	Return (ENTER)
\”	Double quote
\’	Single quote
\\	Backslash
\ddd	Octal code
\uXXXX	Hex-decimal code

Παράδειγμα: StringTest2.java

```
class StringTest
{
   public static void main(String args[])
   {
        String s = “hello world”;
        String h = “hello”;
        String w = “world”;
        System.out.println(s);
        System.out.println("\""+h+"\t"+w+"\"\n");
        System.out.println(s.length());
        System.out.println(“hello”.length());
        System.out.println(h.equals("hello"));
   }
}
```

Ρεύματα εισόδου/εξόδου

- Τι είναι ένα ρεύμα? Μια αφαίρεση που αναπαριστά μια πηγή (για την είσοδο), ή ένα προορισμό (για την έξοδο) χαρακτήρων
  - Είσοδος: Μπορεί να είναι το πληκτρολόγιο (standard input) ή ένα αρχείο.
  - Έξοδος: Μπορεί να είναι η οθόνη (standard output) ή ένα αρχείο.
  - Όταν δημιουργούμε το ρεύμα το συνδέουμε με την ανάλογη πηγή, ή προορισμό.

ΕΙΣΟΔΟΣ - ΕΞΟΔΟΣ

Είσοδος & Έξοδος

- 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/εξόδου έχει αποτέλεσμα το λειτουργικό να πάρει ή να στείλει χαρακτήρες από/προς την αντίστοιχη πηγή/προορισμό.

Έξοδος

- Μπορούμε να καλέσουμε τις μεθόδους του System.out:
  - println(String s): για να τυπώσουμε ένα αλφαριθμητικό s και τον χαρακτήρα ‘\n’ (αλλαγή γραμμής)
  - print(String s): τυπώνει το s αλλά δεν αλλάζει γραμμή
  - printf: Formatted output
    - printf(“%d”,myInt); // τυπώνει ένα ακέραιο
    - printf(“%f”,myDouble); // τυπώνει ένα πραγματικό
    - printf(“%.2f”,myDouble); // τυπώνει ένα πραγματικό με δύο δεκαδικά

```
class Output
{
    public static void main(String[] args)
    {
        System.out.print("Hello "); //prints without changing line
        System.out.println("World"); //prints and changes line

        System.out.println(); //adds an extra empty line

        int x = 1;
        int y = 3;
        double d = (double)x/y;
        System.out.println("Division of "+x+" by "+y+" equals "+d);
        System.out.printf("Division of %d by %d equals %.2f\n",x,y,d);
        System.out.println("End of program");
    }
}
```

Escape character sequence το οποίο σημαίνει ότι θέλουμε να προσθέσουμε αλλαγή γραμμής

Είσοδος – Η κλάση Scanner

- Για να διαβάσουμε από την είσοδο θα χρησιμοποιούμε την κλάση Scanner της Java. Η βιβλιοθήκη συνδέει το πρόγραμμα μας με ένα ρεύμα εισόδου και μας επιτρέπει να διαβάζουμε χαρακτήρες από εκεί.

- Για να την χρησιμοποιήσουμε θα πρέπει την «φέρουμε» μέσα στο πρόγραμμα μας, χρησιμοποιώντας την εντολή import
  - import java.util.Scanner;
  - Η εντολή καλείται στην αρχή του αρχείου μας πριν τον ορισμό της κλάσης.

- Μετά πρέπει να δημιουργήσουμε ένα αντικείμενο τύπου Scanner χρησιμοποιώντας την new.
- Το αντικείμενο αρχικοποιείται με παράμετρο το ρεύμα εισόδου. Θα χρησιμοποιήσουμε το ρευμα System.in που αντιστοιχεί στο standard input, που είναι το πληκτρολόγιο.
  - Scanner input = new Scanner(System.in);
  - Μπορείτε να σκεφτείτε αυτή την εντολή σαν να συνδέω ένα πληκτρολόγιο με το πρόγραμμα μου από το οποίο μπορώ να πάρω χαρακτήρες.

Μέθοδοι της κλάσης Scanner

- Έχοντας το αντικείμενο της κλάσης Scanner μπορούμε να καλέσουμε τις μεθόδους της κλάσης με τον τελεστή της τελείας '.'

- Θα χρησιμοποιούμε τις εξής μεθόδους της Scanner για να διαβάσουμε κάτι από την είσοδο
  - nextLine(): διαβάζει μέχρι να βρει τον χαρακτήρα '\n'. Επιστρέφει ένα String με την γραμμή που διάβασε
  - next(): διαβάζει μέχρι να βρει λευκό χαρακτήρα. Επιστρέφει ένα String με αυτό που διάβασε
  - nextInt(): διαβάζει μέχρι να βρει λευκό χαρακτήρα. Αυτό που διάβασε το μετατρέπει σε ακέραιο και το επιστρέφει. Επιστρέφει ένα Integer.
  - nextDouble(): διαβάζει μέχρι να βρει λευκό χαρακτήρα. Αυτό που διάβασε το μετατρέπει σε πραγματικό αριθμό και το επιστρέφει. Επιστρέφει ένα Double.
  - nextBoolean(): διαβάζει μέχρι να βρει λευκό χαρακτήρα. Αυτό που διάβασε (true ή false) το μετατρέπει σε λογική τιμή και το επιστρέφει. Επιστρέφει ένα Boolean.

- Παράδειγμα κλήσης μεθόδου:
  - Με το αντικείμενο input καλούμε την μέθοδο nextLine() γράφοντας:
  - input.nextLine()

Παράδειγμα

Με την εντολή αυτή φέρνουμε την κλάση Scanner μέσα στο πρόγραμμα μας ώστε να μπορούμε να φτιάξουμε αντικείμενα τύπου Scanner

```
import java.util.Scanner;

class TestIO
{
	public static void main(String[] args)
	{
		System.out.println("Say Something:");
		Scanner input = new Scanner(System.in);
		String line = input.nextLine();
		System.out.println(line);
	}
}
```

Ορισμός μεταβλητής String

new: δημιουργεί ένα αντικείμενο τύπου **Scanner** (μία μεταβλητή) με το οποίο μπορούμε πλέον να διαβάζουμε από την είσοδο.

Η μεταβλητή-αντικείμενο Scanner

- Όταν εκτελούμε την εντολή

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

δίνουμε πρόσβαση στο πρόγραμμα μας στο πληκτρολόγιο μέσω του αντικειμένου input.

- Μπορείτε να το σκεφτείτε σαν να συνδέετε ένα πληκτρολόγιο με το πρόγραμμα.
- Μπορούμε να χρησιμοποιήσουμε την μεταβλητή για να διαβάσουμε πολλαπλές τιμές.
- Δεν χρειάζεται να δημιουργείτε ένα καινούριο αντικείμενο για να διαβάσετε μια καινούρια τιμή
  - Όπως δεν χρειάζεται να συνδέσετε ένα καινούριο πληκτρολόγιο κάθε φορά που θέλετε να εισάγετε καινούριες τιμές.

Παράδειγμα

```
import java.util.Scanner;

class TestIO
{
	public static void main(String args[])
	{
		System.out.println("Say Something");
		Scanner input = new Scanner(System.in);
		String line = input.nextLine();
		System.out.println(line);
		System.out.print("Say Something More:");
		line = input.nextLine();
		System.out.println(line);
	}
}
```

Παράδειγμα

```
import java.util.Scanner;

class TestIO2
{
	public static void main(String args[])
	{
		Scanner input = new Scanner(System.in);
		System.out.println("Give student name,
						AM, and grade:");
		String name = input.next();
		int AM = input.nextInt();
		double grade = input.nextDouble();
		System.out.println("Student " + name
				+ " with AM "+ AM
				+ " got grade " + grade);
	}
}
```

Παράδειγμα

```
import java.util.Scanner;

class TestIO3
{
    public static void main(String[] args)
    {
        Scanner input = new Scanner(System.in);
        double d= input.nextDouble();
        System.out.println("division by 4 = " + d/4);
        System.out.println("1+ (division by 4) = " +1+d/4);
        System.out.printf("1+ (division of %.2f by 4) = %.2f",d, 1+d/4);
    }
}
```

Το + λειτουργεί ως concatenation τελεστής μεταξύ Strings, άρα μετατρέπει τους αριθμούς σε Strings

Τι θα τυπώσει αυτό το πρόγραμμα?