Περισσότερα για τις Κλάσεις, τα Αντικείμενα και τις Μεθόδους

Σύνοψη των βασικών λειτουργιών στον Αντικειμενοστρεφή Προγραμματισμό

1) Ορισμός Κλάσης:

```
[ορατότητα] class <όνομα-κλάσης>
{
// μεταβλητές
// δομητές-κατασκευαστές
/ /μέθοδοι
}
```

2) Ορισμός μεταβλητής:

```
[ορατότητα] <τύπος> <όνομα> [=τιμή];
public int x1 = 5;
private float Synt_FPA=0.19f ;
private double Arhiko_Poso ;
public String Eponymia = "Nikas Nikos";
```

3) Ορισμός δομητή-κατασκευαστή:

```
[public] <όνομα δομητή-κλάσης>( [παράμετροι] )
{
 //αρχικοποίηση πεδίων-μεταβλητών
}
```

Αρχικός (default) δομητής:

```
[public] <όνομα δομητή-κλάσης>()
{
 //αρχικοποίηση πεδίων-μεταβλητών με τιμές 0
}
```

Παράδειγμα:

Στην κλάση Box, ο αρχικός δομητής είναι:

```
public Box() {}
```

4) Ορισμός μεθόδου:

```
[ορατότητα] <τύπος> <όνομα> ( [παράμετροι] )
{
 //σώμα μεθόδου
}
```

Τύποι Μεθόδων

Α) <u>Λειτουργικοί (operator) μέθοδοι:</u> Υπολογίζουν χωρίς να επιστρέφουν αποτέλεσμα, δηλαδή είναι τύπου void. Συνήθως εκτελούν τις εντολές του σώματός τους.

```
public void method()  // και παράμετροι
{
 // δηλώσεις;
 // υπολογισμοί;
}
```

Β) <u>Τροποποιητικές (mutator ή modifier) μέθοδοι</u> *<u>(αλλαγής ή τοποθέτησης τιμής)</u>*<u>:</u>

Δέχονται πάντα κάποια παράμετρο, την τιμή της οποίας συνήθως «περνάνε» σε κάποια private μεταβλητή της κλάσης τους. Συνήθως είναι τύπου void. Π.χ.

```
public void setLastName(String n)
{
 LastName = n;
}
```

Γ) Πρόσβασης ή απόκτησης τιμής (accessor) μέθοδοι:

Είναι πάντα public μέθοδοι και επιστρέφουν (return) μια private μεταβλητή του ίδιου τύπου με αυτόν που ορίζονται. Π.χ.

```
int getRadius()
{
  return (radius);
}
```

Δ) Μεικτού τύπου:

Συνήθως είναι συνδυασμοί των Β) και Γ). Π.χ.

```
double Akatharistos_Misthos(int mer, double hmer, float yper) {
 double Akm;
 Akm = mer * hmer + 0.2 * hmer * yper;
 return Akm;
}
```

5) Δημιουργία και αρχικοποίηση αντικειμένου (στιγμιότυπου) κλάσης:

<Κλάση> <αντικείμενο> = new <Κλάση> ( ); //με ή χωρίς παραμέτρους

Παραδείγματα:

Box MyBox1 = new Box();

Box MyBox1 = new Box(10, 20, 15);

6) Κλήση μεθόδου αντικειμένου

Οι μέθοδοι μιας κλάσης ανήκουν ουσιαστικά στα αντικείμενά της. Η κλήση μιας μεθόδου αντικειμένου γίνεται με την εντολή:

<αντικείμενο>.<μέθοδος>( );

Προσοχή!! Δεν ισχύει για κλήση μεθόδου κλάσης.

Η δεσμευμένη λέξη - this

Χρησιμοποιείται μέσα σε μεθόδους όταν γίνεται αναφορά στο τρέχον αντικείμενο της μεθόδου. Ο δομητής box() θα μπορούσε να γραφεί:

```
Box(double x, double y, double z) {
  this.width = x;
  this.height = y;
```

```
  this.depth = z;
 }
```

Αν οι μεταβλητές των αντικειμένων έχουν το ίδιο όνομα με τις τοπικές μεταβλητές, τότε υπερισχύουν οι τοπικές μεταβλητές και 'κρύβουν' τις τιμές των μεταβλητών των αντικειμένων. Στις περιπτώσεις αυτές χρησιμοποιούμε και πάλι την this :

```
 Box(double width, double height, double depth) {
  this.width = width;
  this.height = height;
  this.depth = depth;
 }
```

Συλλογή σκουπιδιών (Garbage collection)

Η java διαγράφει αυτόματα τα αντικείμενα που δεν χρησιμοποιούνται και ελευθερώνει την μνήμη που καταλαμβάνουν. Η τεχνική αυτή ονομάζεται 'συλλογή σκουπιδιών'. Η τεχνική αυτή λειτουργεί ως εξής: όταν μέσα στο πρόγραμμα δεν υπάρχει κάποια αναφορά σε κάποιο αντικείμενο, τότε θεωρείται ότι το αντικείμενο δεν χρειάζεται πλέον και διαγράφεται. Αν θέλουμε να καλέσουμε τον συλλογέα σκουπιδιών, τότε καλούμε μια **pubic static μέθοδο** της **System** την **gc** (από το garbage collection). Δηλαδή **System.gc();** Η εντολή αυτή κάνει αμέσως αποκομιδή των απορριμμάτων και επιστρέφει μνήμη στο σύστημα. Πριν ο gc αποκομίσει ένα αντικείμενο καλεί την μέθοδο **finalize()** για αυτό το αντικείμενο (θα την δούμε αμέσως μετά).

Η μέθοδος Finalize()

Είναι η μέθοδος που χρησιμοποιείται όταν πριν από την διαγραφή ενός αντικειμένου απαιτείται η εκτέλεση κάποιων άλλων εργασιών. Η java πριν την διαγραφή ενός αντικειμένου καλεί την μέθοδο αυτή και εκτελεί τον κώδικα που βρίσκεται στο σώμα της finalize(). Η γενική μορφή της μεθόδου είναι :

```
protected void finalize()
{
// κώδικας της μεθόδου
}
```

Η δεσμευμένη λέξη protected χρησιμοποιείται για να μην μπορεί να προσπελαθεί ο κώδικας της finalize() από κώδικα εξωτερικά της κλάσης.

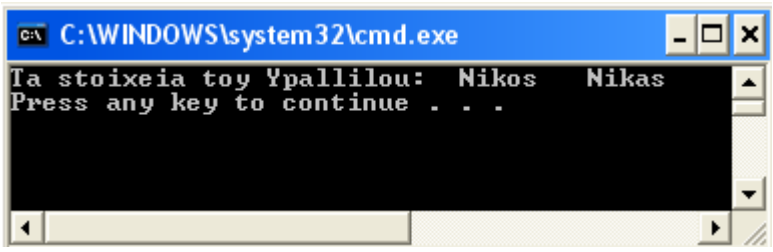
Η μέθοδος toString()

Όλες οι κλάσεις της Java είναι υποκλάσεις της κλάσης **Object** που περιέχει πολλές σημαντικές μεθόδους που δεν θα αναφέρουμε ή αναλύσουμε προς το παρόν (μπορείτε να τις διαβάσετε από το βιβλίο, σελ. 298). Όμως θα αναλύσουμε μία από αυτές τις μεθόδους την **String toString()** η οποία χρησιμοποιείται για να 'εμφανίσει' τα στοιχεία ενός αντικειμένου. Η μέθοδος αυτή επιστρέφει μια συμβολοσειρά (String) που περιγράφει το αντικείμενο από το οποίο καλείται. Η κλήση της γίνεται συνήθως μέσα από την μέθοδο **println.** Όταν καλείται η μέθοδος (π.χ. εμφάνιση αντικειμένου μέσα από την println), τότε γίνεται υπερφόρτωση της μεθόδου που κληρονομεί από την Object.

Παράδειγμα

```
class ToStringExample {
   String firstName;
   String lastName;

   ToStringExample(String fn, String ln) {
      this.firstName = fn;
      this.lastName = ln;
   }
   // Υπερφόρτωση (Override) την μέθοδο toString()
   public String toString() {
      return firstName  + "    " +lastName;
   }
}

public class ToString {
   public static void main(String[] args) {

      ToStringExample ypallilos = new ToStringExample("Nikas", "Nikos");
      System.out.println("Ta stoixeia toy Ypallilou:  " + ypallilos);
   }
}
```

Αν εκτελέσουμε το πρόγραμμα θα πάρουμε τα παρακάτω αποτελέσματα :



Η κλάση Math – Μέθοδοι της κλάσης – Μαθηματικές Συναρτήσεις

Εκτός από την εκτέλεση απλών αριθμητικών πράξεων υπάρχουν και οι πιο σύνθετοι υπολογισμοί που απαιτούν μαθηματικές ή άλλες επιστημονικές πράξεις. Για τον λόγο αυτό στην κλάση **Math.class** (του πακέτου **java.lang**) της Java έχουν ενσωματωθεί οι κυριότερες μαθηματικές συναρτήσεις των οποίων η λειτουργία είναι όμοια με εκείνη των μεθόδων. Μια τέτοια συνάρτηση - μέθοδος δέχεται μία ή περισσότερες παραμέτρους και αφού τις επεξεργαστεί επιστρέφει το αντίστοιχο αποτέλεσμα. Οι συναρτήσεις αυτές μπορούν να κληθούν χωρίς τη δημιουργία αντικειμένου της κλάσης Math, αλλά απλά γράφοντας το όνομα της κλάσης και την τελεία – dot (**.**), π.χ. **Math.min(x, y).** Όλες οι μέθοδοι της Math επιστρέφουν αποτέλεσμα τύπου double.

Αλγεβρικές συναρτήσεις

abs(int x) abs(long x) abs(float x) abs(double x)	Επιστρέφουν την απόλυτη τιμή του x.
max(int x, int y) max(long x, long y) max(float x, float y) } max(double x, double y)	Επιστρέφουν τον μέγιστο των x και y.
min(int x, int y) min(long x, long y) min(float x, float y) } min(double x, double y)	Επιστρέφουν τον μικρότερο των x και y.

ceil(double x)	Στρογγυλοποίηση προς τον μικρότερο ακέραιο, αλλά όχι μικρότερο του x (>=x) (π.χ. ceil(6.2)=7.0 και ceil(-6.2) = -6.0)
floor(double x)	Στρογγυλοποίηση προς τον μεγαλύτερο ακέραιο, αλλά όχι μεγαλύτερο του x (<=x)(π.χ. floor(9.2) = 9.0 και floor(-9.8) = -10.0)
rint(double x)	Πλησιέστερος ακέραιος στο x
round(float x)	Πλησιέστερος int στο float x
round(double x)	Πλησιέστερος long στο double x

Ημιτονοειδείς συναρτήσεις

sin(double x)	Ημίτονο του x (το x σε ακτίνια)
cos(double x)	Συνημίτονο του x (το x σε ακτίνια)
tan(double x)	Εφαπτομένη του x (το x σε ακτίνια)
asin(double x)	Τόξο ημίτονου x
acos(double x)	Τόξο συνημίτονου x
atan(double x)	Τόξο εφαπτομένης x
atan2(double x, double y)	Τόξο εφαπτομένης x/y

Εκθετικές συναρτήσεις

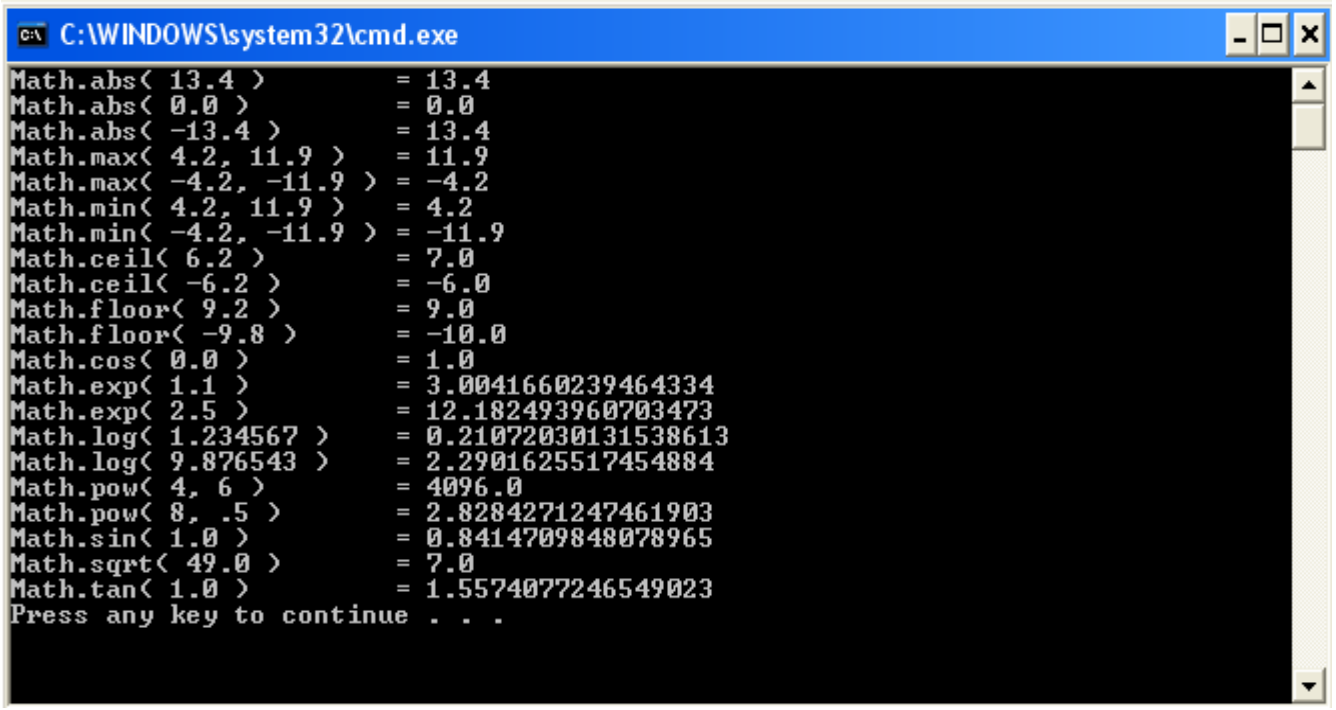
exp(double x)	Επιστρέφει την e^x
log(double x)	Επιστρέφει τον φυσικό λογάριθμο του x
pow(double x, double y)	Επιστρέφει την x^y
sqrt(double x)	Επιστρέφει την τετραγωνική ρίζα της x (ή $x^{1/2}$)

Παράδειγμα:

```
public class MathTest {
  public static void main( String args[] )  {
    System.out.println( "Math.abs( 13.4 )        = " + Math.abs( 13.4 ) );
    System.out.println( "Math.abs( 0.0 )         = " + Math.abs( 0.0 ) );
    System.out.println( "Math.abs( -13.4 )       = " + Math.abs( -13.4 ) );

    System.out.println( "Math.max( 4.2, 11.9 )   = " + Math.max( 4.2, 11.9 ) );
    System.out.println( "Math.max( -4.2, -11.9 ) = " + Math.max(  -4.2, -11.9  ) );
    System.out.println( "Math.min( 4.2, 11.9 )   = " + Math.min( 4.2, 11.9 ) );
    System.out.println( "Math.min( -4.2, -11.9 )  = " + Math.min( -4.2, -11.9 ) );

    System.out.println( "Math.ceil( 6.2 )          = " + Math.ceil( 6.2 ) );
    System.out.println( "Math.ceil( -6.2 )         = " + Math.ceil( -6.2 ) );
    System.out.println( "Math.floor( 9.2 )         = " + Math.floor( 9.2 ) );
    System.out.println( "Math.floor( -9.8 )        = " + Math.floor( -9.8 ) );

    System.out.println( "Math.cos( 0.0 )           = " + Math.cos( 0.0 ) );
    System.out.println( "Math.exp( 1.1 )           = " + Math.exp( 1.1 ) );
```

```
    System.out.println( "Math.exp( 2.5 )           = " + Math.exp( 2.5 ) );
    System.out.println( "Math.log( 1.234567 )      = " + Math.log( 1.234567 ) );
    System.out.println( "Math.log( 9.876543 )      = " + Math.log( 9.876543 ) );

    System.out.println( "Math.pow( 4, 6 )          = " + Math.pow( 4, 6 ) );
    System.out.println( "Math.pow( 8, .5 )         = " + Math.pow( 8, .5 ) );
    System.out.println( "Math.sin( 1.0 )           = " + Math.sin( 1.0 ) );
    System.out.println( "Math.sqrt( 49.0 )         = " + Math.sqrt( 49.0 ) );
    System.out.println( "Math.tan( 1.0 )           = " + Math.tan( 1.0 ) );
  }
}
```

Αν εκτελέσουμε το πρόγραμμα θα πάρουμε τα αποτελέσματα:



Τυχαίοι αριθμοί

Η Math.random() επιστρέφει ένα τυχαίο αριθμό τύπου double στο διάστημα: 0.0 <= X < 1.0. Κάθε φορά που εκτελείται η random() παράγεται διαφορετικός τυχαίος αριθμός.

Τυχαίοι ακέραιοι σε διάστημα

Για να πάρουμε ένα τυχαίο αριθμό σε κάποιο διάστημα χρησιμοποιούμε την σχέση:

```
n = a + (int) ( Math.random() * b );
```

Όπου: n = τυχαίος ακέραιος, a = μετατόπιση, b = πολλαπλασιαστής

Π.χ για τυχαίους στο διάστημα [1, 6]: n = 1 + (int) (Math.random() * 6);

```
class RandomTest {
  public static void main(String[] args) {
        int x;
         x = 1 + (int) ( Math.random() * 6 );
         System.out.println("x= "+x);
  }
}
```

Η κλάση Random

Ένας άλλος τρόπος για να πάρουμε ακέραιους μέσα σε κάποιο διάστημα είναι να χρησιμοποιήσουμε την nextInt(), με ή χωρίς παράμετρο διαστήματος, της κλάσης Random. Στην περίπτωση αυτή γράφουμε πρώτα την εντολή (βιβλιοθήκη κλάσης): **import java.util.Random;** για να χρησιμοποιήσουμε τις μεθόδους της κλάσης Random. Μετά δημιουργούμε αντικείμενο του τύπου Random και καλούμε την **nextInt()** για να πάρουμε τυχαίους ακέραιους αριθμούς, διαφορετικών διαστημάτων.

```
import java.util.Random;
class RandomTest {
 public static void main(String[] args) {
    Random rnd = new Random();
    int x;

   //ακέραιοι από τα – 2 δις μέχρι το + 2δις
    x=rnd.nextInt();
    System.out.println("x= "+x);

    //ακέραιοι μέχρι το 100
    x=rnd.nextInt(100);
    System.out.println("x= "+x);
 }
 }
```