ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΕΦΗΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

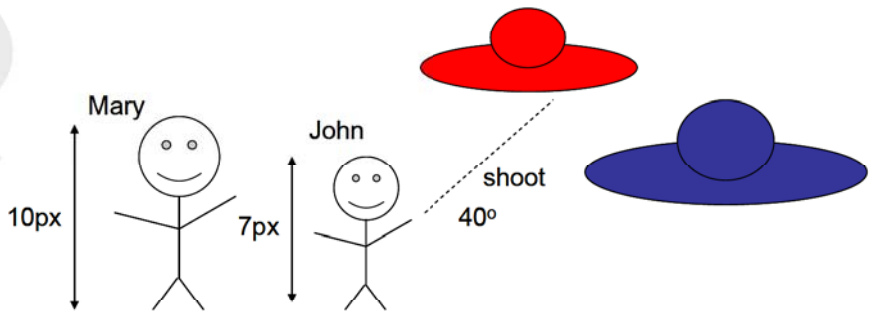
Δημήτρης Κ. Ιακωβίδης

Στόχος αυτής της ενότητας είναι η εισαγωγή στις βασικές έννοιες του αντικειμενοστρεφούς προγραμματισμού, όπως είναι οι κλάσεις, τα αντικείμενα, οι μεταβλητές υπόστασης (ή στιγμιότυπου) και οι μέθοδοι.

3.1 Αντικειμενοστρεφής λογική

Στον αντικειμενοστρεφή προγραμματισμό ένα πρόγραμμα δομείται ως σύνολο αυτόνομων δομικών μονάδων. Κάθε μονάδα αναπαριστά μια οντότητα, η οποία μπορεί να λάβει διαφορετικές υποστάσεις στον κόσμο μας. Τις μονάδες αυτές επιθυμούμε να μπορούμε να τις χρησιμοποιούμε σε διαφορετικά λογισμικά χωρίς να χρειάζεται να επεμβαίνουμε στον κώδικά τους.

Παράδειγμα. Έστω ότι επιθυμούμε να κατασκευάσουμε ένα παιχνίδι με ανθρωπάκια και ουφάκια, όπου τα ανθρωπάκια έχουν τη δυνατότητα να πυροβολούν τα ουφάκια (Σχ.1). Σκεπτόμαστε: σε πόσα κομμάτια τα οποία θα μπορούσαμε να ξαναχρησιμοποιήσουμε μπορούμε να σπάσουμε το πρόβλημα; Για παράδειγμα, ένα ανθρωπάκι είναι μια αυτόνομη οντότητα που θα μπορούσαμε να χρησιμοποιήσουμε και σε άλλο παιχνίδι. Αντίστοιχα και τα ουφάκια μπορούν να θεωρηθούν ως επαναχρησιμοποιήσιμες αυτόνομες οντότητες. Κάθε οντότητα, προγραμματιστικά αναπαριστάται από μια **κλάση** (class). Έτσι, το παιχνίδι μας θα αποτελείται συνολικά από τρεις κλάσεις: α) μια κλάση που θα περιγράφει τι είναι ανθρωπάκι, β) μια κλάση που θα περιγράφει τι είναι ουφάκι, και γ) μια κλάση που θα περιγράφει το παιχνίδι μέσα στο οποίο θα παίζουν τα ανθρωπάκια και τα ουφάκια, η οποία θα πρέπει να είναι εκτελέσιμη.



Σχήμα 1. Παράδειγμα παιχνιδιού με ανθρωπάκια και ουφάκια.

Κάθε ανθρωπάκι διαθέτει ένα σύνολο από **ιδιότητες** (properties), π.χ. έχει το δικό του όνομα, το δικό του ύψος, και μπορεί να εκτελεί **ενέργειες** π.χ. να πυροβολεί (shoot) υπό κάποια γωνία (angle). Επομένως ένα ανθρωπάκι μπορεί να λάβει διαφορετικές **υποστάσεις** (instances) ή, με άλλα λόγια, τα διαφορετικά ανθρωπάκια αποτελούν διαφορετικά **στιγμιότυπα** της έννοιας άνθρωπος π.χ. μπορεί να ονομάζεται Γιάννης (John) και να έχει ύψος (height) 7 pixels, ενώ ένα άλλο ανθρωπάκι μπορεί να ονομάζεται (name) Μαρία (Mary) και να έχει ύψος 10 pixels. Ωστόσο, αξίζει να σημειωθεί όλα τα ανθρωπάκια έχουν δύο μάτια, δηλαδή το πλήθος των ματιών είναι μια ιδιότητα κοινή για όλα τα ανθρωπάκια. Ομοίως ένα ουφάκι μπορεί να έχει διαφορετικό μέγεθος (size) από ένα άλλο, όπως επίσης μπορεί να έχει και διαφορετικό χρώμα (color) από ένα άλλο.

Οι παραπάνω ιδιότητες υλοποιούνται ως **μεταβλητές** (variables) μέσα σε μια κλάση, και επειδή μπορούν να έχουν διαφορετικές τιμές για διαφορετικές υποστάσεις, ονομάζονται **μεταβλητές υπόστασης** ή **στιγμιότυπου** (instance variables). Οι ενέργειες που μπορεί να εκτελέσει μια οντότητα, όπως είναι ένα ανθρωπάκι, υλοποιούνται ως **συναρτήσεις** (functions), δηλαδή ως υπο-προγράμματα της κλάσης μας. Επειδή μια συνάρτηση περιγράφει ουσιαστικά τη μέθοδο με την οποία υλοποιείται μια ενέργεια, στο πλαίσιο του αντικειμενοστρεφούς προγραμματισμού **μια συνάρτηση ονομάζεται μέθοδος** (method) της κλάσης. Στα Σχ. 2, 3 και 4 παρουσιάζονται αντίστοιχα οι τρεις κλάσεις που περιγράφουν το παιχνίδι με τα ανθρωπάκια και τα ουφάκια του Σχ.1.

```
1    class Human
2    {
3            // Instance variables
4            String name;
5            int height;
6
7            // Static variable
8            static int numberOfEyes;
9
10           // Method
11           void shoot(int angle)
12           {
13               System.out.println("I am " + name + " and I
14               shoot at " + angle + " degrees");
15           }
16
17           // Static method
18           static void whatAmI()
19           {
```

```
20              System.out.println("I am a human.");
21          }
22  }
```

Σχήμα 2. Κλάση Java που περιγράφει τι είναι "ανθρωπάκι".

```
1   class Ufo
2   {
3           // Instance variables
4           String color;
5           int size;
6   }
```

Σχήμα 3. Κλάση Java που περιγράφει τι είναι "ουφάκι".

```
1   class Game
2   {
3           public static void main(String args[])
4           {
5               // John is represented by variable human1
6               Human human1;
7
8               // John needs to be constructed using "new"
9               human1 = new Human();
10              human1.name = "John";
11              human1.height = 7;
12
13              // Mary is represented by variable human2
14              Human human2 = new Human();
15              human2.name = "Mary";
16              human2.height = 10;
17
18              // Ufos needs to be constructed as well
19              Ufo ufo1 = new Ufo();
20              ufo1.color = "red";
21              ufo1.size = 10;
22
23              Ufo ufo2 = new Ufo();
24              ufo2.color = "blue";
25              ufo2.size = 20;
26
27              // John shoots at 40 degrees
28              human1.shoot(40);
29
30              // Static methods are called using class name
31              Human.whatAmI();
32          }
33  }
```

Σχήμα 4. Κλάση Java που περιγράφει το παιχνίδι.

Στις μεθόδους (συναρτήσεις) whatAmI και shoot του Σχ.2, η λέξη **void** σημαίνει ότι δεν επιστρέφουν κάποια τιμή και μπορούμε να τις καλούμε σα να είναι εντολές (Σχ.4, γραμμές 28, 31). Η μεταβλητή **angle** της μεθόδου shoot (Σχ.2, γραμμή 11), χρησιμοποιείται ως **παράμετρος** για να προσδιορίζουμε τη γωνία κατά την οποία θα πυροβολεί ένα ανθρωπάκι, π.χ., στη γραμμή 28 του Σχ.4 το ανθρωπάκι human1 πυροβολεί υπό γωνία 40 μοιρών.

Η κλάση Game περιλαμβάνει μια μέθοδο **main**, γεγονός που την κάνει **εκτελέσιμη**, π.χ., από τη γραμμή εντολών (command line). Μια μέθοδος main μπορεί να υπάρξει σε οποιαδήποτε κλάση. Για παράδειγμα, θα μπορούσε και η κλάση Human να έχει μέθοδο main προκειμένου μέσα σε αυτή να δοκιμάζαμε αν οι μέθοδοί της λειτουργούν σωστά (χρήση ως **δοκιμαστική μέθοδος**). Σημειώνεται πως οι κλάσεις Human και Ufo δε μπορούν να εκτελεσθούν γιατί δεν περιλαμβάνουν μέθοδο main. Περιγράφουν τι ορίζεται ως ανθρωπάκι και τι ορίζεται ως ουφάκι, και μπορούν να χρησιμοποιηθούν μόνο για την παραγωγή διαφορετικών υποστάσεων ανθρώπων και ούφο, αντίστοιχα.

Στην κλάση Game δημιουργούνται υποστάσεις της κλάσης Human. Αυτές είναι η **human1** (ο Γιάννης) και η **human2** (η Μαρία), και ονομάζονται **αντικείμενα** (objects) της κλάσης Human. Αντίστοιχα, οι υποστάσεις της κλάσης Ufo, που είναι οι ufo1 και ufo2, ονομάζονται αντικείμενα της κλάσης Ufo. Ένα αντικείμενο δεν είναι τίποτε περισσότερο από μια μεταβλητή. Μια μεταβλητή τέτοιου τύπου ονομάζεται **μεταβλητή τύπου κλάσης** (class type). Η **δημιουργία των αντικειμένων** στη Java γίνεται με τη χρήση της λέξης-κλειδί **new**. Για να αναφερθούμε στις μεταβλητές υπόστασης κάθε αντικειμένου, χρησιμοποιούμε το σύμβολο της **τελείας** (dot), με τον ίδιο τρόπο που θα χρησιμοποιούσαμε τη λέξη "**του**". Δηλαδή, ο συμβολισμός human1.name σημαίνει "η μεταβλητή name **του** αντικειμένου human1", και η έκφραση human1.name = "John", σημαίνει πως ανατίθεται η τιμή "John" στη μεταβλητή name του human1.

3.2 Μέθοδοι

Η έννοια της συνάρτησης είναι εμπνευσμένη από τα μαθηματικά, ωστόσο, δεν απαιτεί γνώσεις μαθηματικών. Στα μαθηματικά μια συνάρτηση μπορεί να είναι της μορφής:

$$f(x, y) = 2x + 4y$$

όπου x, y είναι οι μεταβλητές εισόδου της συνάρτησης, οι οποίες πρέπει να είναι γνωστές για να υπολογιστεί η τιμή (έξοδος) της συνάρτησης f(x, y).

Η μαθηματική συνάρτηση f(x, y) μπορεί να υλοποιηθεί ως μέθοδος στη γλώσσα Java όπως εικονίζεται στο Σχ.5. Ο ορισμός της μεθόδου πραγματοποιείται στις γραμμές 4-9. Η πρόταση της γραμμής 4 ονομάζεται η **επικεφαλίδα** (header) της μεθόδου, και το σύνολο των γραμμών 5-9 που ακολουθούν εντός αγκυλών, ονομάζεται **σώμα** (body) της μεθόδου. Η μέθοδος f του Σχ.5 υπολογίζει μια τιμή την οποία **επιστρέφει** (returns) στο κυρίως πρόγραμμα (στη μεταβλητή output).

3.2.1. Κατασκευή επικεφαλίδας μεθόδου:

- **Αριστερά** του ονόματός της δηλώνεται **ο τύπος** της τιμής που επιθυμούμε να επιστρέφει η μέθοδος (π.χ., στο Σχ.5 ο τύπος αυτός είναι double). Ο τύπος αυτός ονομάζεται και **επιστρεφόμενος τύπος** (returned type).
- Η μέθοδος μπορεί να **επιστρέφει μόνο μια τιμή** ή **καμία τιμή**. Για να μην επιστρέφει καμία τιμή χρησιμοποιείται ο τύπος **void**.
- **Δεξιά** του ονόματός της μέσα σε παρενθέσεις δηλώνονται **οι μεταβλητές εισόδου** (παράμετροι) της μεθόδου, χωρίς περιορισμό ως προς το πλήθος τους.

3.2.2. Κατασκευή σώματος μεθόδου:

- Αν η μέθοδος δεν επιστρέφει τιμή (είναι δηλαδή τύπου void) τότε στο σώμα της περιλαμβάνεται ένα σύνολο προτάσεων που υλοποιούν την επιθυμητή ενέργεια, όπως στη μέθοδο shoot της κλάσης Human (Σχ.2, γραμμή 11).
- Αν η μέθοδος επιστρέφει τιμή τότε προτείνονται τα εξής βήματα, όπως στη μέθοδο f της κλάσης Test (Σχ.5, γραμμές 6-8):
    - **Βήμα 1**. **Δηλώνουμε μια μεταβλητή** του ίδιου τύπου με τον επιστρεφόμενο.
    - **Βήμα 2**. Σημειώνουμε την έκφραση **return** ακολουθούμενη από το **όνομα της μεταβλητής** που δηλώθηκε στο βήμα 1.

- **Βήμα 3.** Ανάμεσα στις παραπάνω εκφράσεις **υλοποιούμε τον υπολογισμό** της επιστρεφόμενης τιμής.

Προσοχή! Η μέθοδος f καλείται στη γραμμή 18 του Σχ.5, όμως για να κληθεί **πρέπει πρώτα να έχει δημιουργηθεί ένα αντικείμενο** της κλάσης (π.χ., της Test) μέσα στην οποία ορίστηκε η μέθοδος. Αυτό διότι η μέθοδος f είναι **μέλος** (member) της κλάσης Test.

```
1    class Test
2    {
3          // Example method or function
4          double f(double x, double y)
5          {
6                double value = 0;        // step 1
7                value = 2 * x + 4 * y;   // step 3
8                return value;            // step 2
9          }
10
11         // Example main method
12         public static void main(String args[])
13         {
14               // Create a new Test object
15               Test object = new Test();
16
17               // Call (use) method f of the object
18               double output = object.f(1, 2);
19               System.out.println("f = " + output);
20         }
21   }
```

Σχήμα 5. Παράδειγμα μεθόδου στη γλώσσα Java.

3.3 Στατικές μεταβλητές και στατικές μέθοδοι

Μια **στατική** (static) μεταβλητή ή **μεταβλητή κλάσης** (class variable) είναι κοινή για όλα τα αντικείμενα της κλάσης μας. Αυτό σημαίνει ότι αν η τιμή της αλλάξει για ένα αντικείμενο της κλάσης μας, θα αλλάξει ομοίως για όλα τα αντικείμενα της κλάσης μας. Για παράδειγμα, αν μέσα στην κλάση Game θέσουμε το πλήθος των ματιών του human1 ίσο με ένα, δηλαδή,

human1.numberOfEyes = 1;

τότε όλα τα ανθρωπάκια (ακόμα και ο human2) θα έχουν ένα μάτι, δηλαδή θα γίνουν κύκλωπες! Για το λόγο αυτό, δε χρειάζεται να δημιουργούμε αντικείμενα για

να θέτουμε τιμές στις στατικές μεταβλητές. Αρκεί να τους δίνουμε τιμές χρησιμοποιώντας απευθείας το όνομα της κλάσης π.χ.

Human.numberOfEyes = 1;

Στην ίδια λογική των στατικών μεταβλητών μπορούν να δηλωθούν και **στατικές μέθοδοι** (static methods), όπως είναι η μέθοδος whatAmI στο Σχ.2. Μια τέτοια μέθοδος είναι κοινή για όλα τα αντικείμενα της κλάσης, υπό την έννοια ότι δε διαφέρει το αποτέλεσμά της είτε κληθεί από το αντικείμενο human1, είτε κληθεί από το αντικείμενο human2. Για το λόγο αυτό δε χρειάζεται να δημιουργούμε αντικείμενα για να καλούμε στατικές μεθόδους. Η κλήση τους γίνεται χρησιμοποιώντας απλά το όνομα της κλάσης, όπως, π.χ., στο τέλος του Σχ.4.

Όταν κατασκευάζουμε μια μέθοδο θα πρέπει να έχουμε πάντα υπόψη μας τους εξής κανόνες:

1. **Αν δεν εξαρτάται από μεταβλητές υπόστασης, πρέπει να ορίζεται ως στατική.**
2. **Μια στατική μέθοδος μπορεί να φιλοξενεί στατικές μεταβλητές,** διότι και αυτές είναι κοινές για όλα τα αντικείμενα της κλάσης.

3.4 Java API

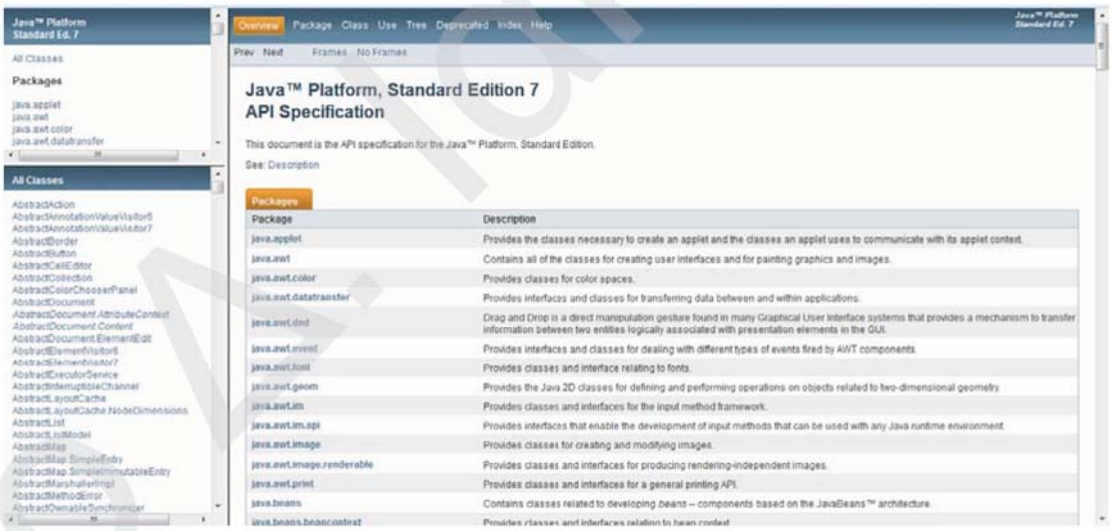
Το σύνολο των κλάσεων που παρέχει η Java προς δημόσια χρήση, και τα περιεχομενά τους, όπως είναι οι μεταβλητές υπόστασης και οι μέθοδοι, συνιστούν τη διεπαφή προγραμματισμού εφαρμογών (Application Programming Interface, API) της Java. Μπορείτε να εξερευνήσετε τα περιεχόμενά της τεκμηρίωσης του Java API στο διαδίκτυο μέσω της ηλεκτρονικής διεύθυνσης: http://docs.oracle.com/javase/7/docs/api/ (Σχ.6).

Όπως προαναφέρθηκε, στη Java οι κλάσεις οργανώνονται σε βιβλιοθήκες που ονομάζονται **πακέτα** (packages). Οι **κλάσεις** (classes) περιέχουν **ιδιότητες** (properties) που υλοποιούνται ως **μεταβλητές υπόστασης** (instance variables), **μεθόδους** (methods). Στην τεκμηρίωση του Java API που εικονίζεται στο Σχ.6 τα πακέτα επιλέγονται άνω αριστερά, οι κλάσεις του επιλεγμένου πακέτου επιλέγονται κάτω αριστερά, και στο κέντρο εμφανίζονται τα περιεχόμενα της επιλεγμένης κλάσης.

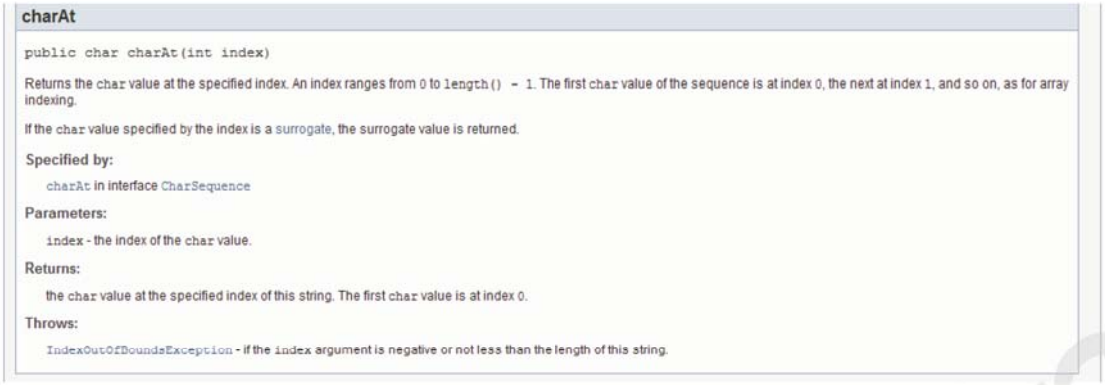
Για παράδειγμα, στο Σχ.7 εικονίζεται η μέθοδος **charAt** της κλάσης **String**. Μπορεί κανείς να παρατηρήσει ότι στην πρώτη γραμμή δίνεται ο τρόπος με τον οποίο ορίζεται η μέθοδος μέσα στην κλάση. Αυτό είναι χρήσιμο για να γνωρίζουμε πως να τη χρησιμοποιήσουμε. Παρακάτω από τη γραμμή ορισμού της μεθόδου, δίνονται βοηθητικές πληροφορίες για αυτή. Έτσι, για παράδειγμα, από τον ορισμό της **charAt** στο Java API καταλαβαίνουμε ότι:

- Επειδή δεν είναι στατική πρέπει να υπάρχει κάποιο αντικείμενο τύπου **String** για να τη χρησιμοποιήσουμε.
- Επιστρέφει μια τιμή τύπου char.
- Δέχεται ως όρισμα μια ακέραια τιμή με την οποία προσδιορίζουμε τη θέση ενός χαρακτήρα μέσα στη μεταβλητή τύπου String.

Τελικά, αν έχουμε ένα αλφαριθμητικό, π.χ., **String s = "abcdefg"** η χρήση της charAt έχει ως αποτέλεσμα να επιστρέψει έναν χαρακτήρα που βρίσκεται σε συγκεκριμένη θέση που προσδιορίζουμε ως όρισμα, π.χ., η κλήση **char c = s.charAt(3)** θα έχει ως αποτέλεσμα να αποθηκευτεί ο χαρακτήρας **'d'** στη μεταβλητή c.



Σχήμα 6. Στιγμιότυπο της τεκμηρίωσης του Java API.



charAt

`public char charAt(int index)`

Returns the `char` value at the specified index. An index ranges from `0` to `length() - 1`. The first `char` value of the sequence is at index `0`, the next at index `1`, and so on, as for array indexing.

If the `char` value specified by the index is a surrogate, the surrogate value is returned.

Specified by:

`charAt` in interface `CharSequence`

Parameters:

`index` - the index of the `char` value.

Returns:

the `char` value at the specified index of this string. The first `char` value is at index `0`.

Throws:

`IndexOutOfBoundsException` - if the `index` argument is negative or not less than the length of this string.

Σχήμα 7. Στιγμιότυπο της τεκμηρίωσης του Java API που παρουσιάζει τη μέθοδο charAt της κλάσης String.

3.5 Περιβάλλουσες κλάσεις

Η Java διαθέτει κάποιες **κλάσεις που αντιστοιχούν σε πρωταρχικούς τύπους**, π.χ., υπάρχει η κλάση **Integer** που αντιστοιχεί σε ακέραιους τύπους **int**. Αντίστοιχα υπάρχουν οι κλάσεις **Character**, **Float**, **Double** κλπ που αντιστοιχούν στους πρωταρχικούς τύπους **char**, **float**, **double** κλπ. Οι κλάσεις αυτές ουσιαστικά διαθέτουν μια μεταβλητή υπόστασης αυτού του τύπου. Υπό αυτή την έννοια **περιβάλλουν** (wrap) έναν πρωταρχικό τύπο, και για το λόγο αυτό ονομάζονται **περιβάλλουσες** (wrapper) κλάσεις.

Ο λόγος ύπαρξης αυτών των κλάσεων είναι ότι παρέχουν χρήσιμα εργαλεία υπό τη μορφή μεθόδων, σταθερών τιμών κλπ που βοηθούν στο χειρισμό των αντίστοιχων πρωταρχικών τύπων. Έτσι, η στατική μέθοδος **parseInt** της κλάσης **Integer** χρησιμοποιείται για τη μετατροπή ενός αλφαριθμητικού, π.χ., "123" σε ακέραιο 123, ενώ οι (στατικές) σταθερές **MAX_VALUE** και **MIN_VALUE** περιέχουν τη μέγιστη και την ελάχιστη τιμή ενός ακεραίου τύπου int. Στο Σχ.7 παρουσιάζεται ένα παράδειγμα που δείχνει πως θα μπορούσαμε να πολλαπλασιάσουμε έναν αριθμό που έχουμε σε αλφαριθμητική (και όχι σε αριθμητική) μορφή με το 1000.

```
1    class TestInteger
2    {
3            public static void main(String args[])
4            {
5                String s = "123";
6                int i = Integer.parseInt(s);
7                System.out.println(i * 1000);
8            }
```

```
9       }
```

Σχήμα 7. Παράδειγμα χρήσης της περιβάλλουσας κλάσης Integer. Εκτυπώνει το γινόμενο της μεταβλητής i με το 1000.

Επειδή χρησιμοποιούνται συχνά, η ανάθεση μιας τιμής σε αντικείμενα περιβαλλουσών κλάσεων γίνεται απλά όπως γίνεται και η ανάθεση τιμών σε αλφαριθμητικά, δηλαδή χρησιμοποιώντας μόνο το σύμβολο της ισότητας, π.χ.,

Integer x = 5;

Αυτό ονομάζεται **αυτόματη συσκευασία** (automatic boxing). Το αντίθετο, επίσης επιτρέπεται και ονομάζεται **αυτόματη αποσυσκευασία** (automatic unboxing), π.χ., η τιμή της παραπάνω μεταβλητής x να ανατεθεί σε ακέραια μεταβλητή πρωταρχικού τύπου:

int y = x;

Ασκήσεις

1. Να υλοποιηθούν τα παραπάνω παραδείγματα.

2. Βάσει των παραπάνω παραδειγμάτων να τροποποιήσετε την κλάση Human ώστε να μπορούν να μιλάνε. Θα λένε ότι δίδεται ως όρισμα σε μια νέα μέθοδο say. Να κατασκευάσετε ένα δικό σας παιχνίδι Game στο οποίο θα υπάρχουν δύο κύκλωπες οι οποίοι θα συνομιλούν.

3. Να κατασκευαστεί μια κλάση Cat που αναπαριστά μια γάτα. Η γάτα θα έχει όνομα και θα μπορεί να εκτελέσει δύο ενέργειες: α) Θα μπορεί να λέει «νιάου» όταν πεινάει, και β) θα μπορεί να λέει «πουρ» όταν θέλει χάδια.

4. Να κατασκευαστεί πρόγραμμα που να χρησιμοποιεί την κλάση Cat για να κατασκευάσει δύο γάτες. Οι γάτες θα πρέπει να έχουν διαφορετικό όνομα, η μία να λέει «νιάου» και η άλλη να λέει «πουρ».

5. Χρησιμοποιώντας την τεκμηρίωση του Java API να κατασκευαστεί πρόγραμμα που να μετατρέπει ένα αλφαριθμητικό ώστε τα γράμματά του να είναι όλα κεφαλαία, π.χ., αν το αλφαριθμητικό είναι “Test” να εκτυπώνει “TEST”.

6. Χρησιμοποιώντας την τεκμηρίωση του Java API να κατασκευαστεί πρόγραμμα που να εκτυπώνει στην οθόνη τους χαρακτήρες ενός αλφαριθμητικού τον έναν κάτω από τον άλλο, π.χ., αν το αλφαριθμητικό είναι “Test” να εκτυπώνει:

T

e

s

t

7. Χρησιμοποιώντας την τεκμηρίωση του Java API να μελετήσετε τη στατική μέθοδο sqrt της κλάσης Math για τον υπολογισμό της τετραγωνικής ρίζας ενός θετικού δεκαδικού αριθμού.

8. Να υλοποιήσετε το παράδειγμα του Σχ.7. Να προσθέσετε μια πρόταση που να ελέγχει αν η τιμή της μεταβλητής i είναι μικρότερη από το ένα χιλιοστό του μέγιστου ακεραίου, και στην περίπτωση αυτή να εκτυπώνει το αλφαριθμητικό “μικρότερη”, διαφορετικά να εκτυπώνει το αλφαριθμητικό “μεγαλύτερη”.

9. Να τροποποιήσετε το παράδειγμα του Σχ.7 ώστε να εκτελείται και για δεκαδικούς αριθμούς τύπου double. Υπόδειξη: μελετήστε αν υπάρχει στο Java API μέθοδος αντίστοιχη της parseInt για double.