ΚΑΤΑΣΚΕΥΑΖΟΝΤΑΣ ΚΛΑΣΕΙΣ Ι

Δημήτρης Κ. Ιακωβίδης

Στόχος αυτής της ενότητας είναι η κατασκευή κλάσεων που αναπαριστούν διάφορες οντότητες, εφοδιασμένες όχι μόνο με μεταβλητές υπόστασης αλλά και με χρήσιμες μεθόδους. Εξηγούνται οι έννοιες των μεθόδων δημιουργών, και της τυπικής μεθόδου toString.

4.1 Κατασκευή νέας κλάσης

Έστω ότι επιθυμούμε να κατασκευάσουμε μια κλάση που να αναπαριστά ένα μιγαδικό αριθμό (complex number). Τι είναι όμως ένας μιγαδικός αριθμός; Από τα μαθηματικά του λυκείου γνωρίζουμε πως ένας μιγαδικός είναι ένας αριθμός της μορφής:

$$z = a+bi,$$

όπου τα a και b είναι πραγματικοί αριθμοί που ονομάζονται πραγματικό (real part) και φανταστικό μέρος (imaginary part) του μιγαδικού αντίστοιχα, και το i είναι ένα σύμβολο, π.χ., z1 = 3+2i.

Για να κατασκευάσουμε μια τέτοια κλάση θα πρέπει να σκεφτούμε τι χαρακτηρίζει έναν μιγαδικό αριθμό, ή με άλλα λόγια, τι κάνει έναν μιγαδικό αριθμό να διαφέρει από έναν άλλο. Προφανώς η απάντηση είναι τα a και b. Αρκεί λοιπόν να κατασκευάσουμε μια κλάση Complex με δύο μόνο μεταβλητές υπόστασης a και b, όπως εικονίζεται στο Σχ.1. Για να τη δοκιμάσουμε προσθέσουμε και μια μέθοδο main μέσα στην οποία δημιουργούμε ένα αντικείμενο της κλάσης Complex που αναπαριστά ένα συγκεκριμένο μιγαδικό αριθμό π.χ. z1 = 3+2i.

```
1    class Complex
2    {
3          // Instance variables
4          double a = 0.0, b = 0.0;
5
6          public static void main(String args[])
7          {
8                // Create a new Complex object
9                Complex z1 = new Complex();
10               // Set values to a and b
11               z1.a = 3;
12               z1.b = 2; // z1 = 3+2i
```

```
13
14                  // Print the Complex object
15                  System.out.println("z1 = " + z1.a + "+" + z1.b
16                  + "i");
17          }
18  }
```

Σχήμα 1. Κλάση Java που αναπαριστά ένα μιγαδικό αριθμό (απλή εκδοχή).

4.2 Μέθοδοι δημιουργοί

Για την ευκολότερη αρχικοποίηση των μεταβλητών υπόστασης τη στιγμή που δημιουργούμε ένα αντικείμενο, υπάρχει η έννοια των **μεθόδων δημιουργών** (constructors) (Σχ.2). Οι μέθοδοι αυτές:

- Επιτρέπουν την αρχικοποίηση των μεταβλητών υπόστασης τη στιγμή που κάνουμε new, γιατί καλούνται αμέσως μετά τη λέξη new.
- Κατασκευάζονται μέσα στην κλάση μας ως ανεξάρτητες μέθοδοι.
- Έχουν το ίδιο όνομα με την κλάση στην οποία βρίσκονται και δεν επιστρέφουν τίποτε, ούτε δηλώνονται ως void.

Αν δεν υπάρχει μέθοδος δημιουργός στην κλάση μας τότε η Java κατασκευάζει αυτόματα μια μέθοδο δημιουργό άνευ ορισμάτων όπως είναι η Complex() του Σχ.2. **Αν υπάρχει μέθοδος δημιουργός** τότε η Java δεν κατασκευάζει τη μέθοδο δημιουργό άνευ ορισμάτων αυτόματα, και αν επιθυμούμε να τη χρησιμοποιήσουμε θα πρέπει να την κατασκευάσουμε εμείς.

4.3 Η μέθοδος toString

Προκειμένου μπορούμε να εκτυπώνουμε ευκολότερα τα αντικείμενα της κλάσης μας στην κονσόλα, η Java μας προσφέρει τη δυνατότητα να κατασκευάζουμε μια συγκεκριμένη μέθοδο (που αναμένεται να υπάρχει, χωρίς αυτό να είναι υποχρεωτικό). Η μέθοδος αυτή είναι η toString και θα πρέπει να δηλώνεται πάντα όπως ακριβώς εικονίζεται στο Σχ.2. Η μέθοδος αυτή μας επιστρέφει το αντικείμενο της κλάσης μας σε αλφαριθμητική μορφή (String), όπως τουλάχιστον φανταζόμαστε ότι αυτό πρέπει να επιστρέφεται. Έτσι, κατασκευάζουμε την toString της Complex για να επιστρέφει ένα μιγαδικό στη μορφή που μάθαμε στα μαθηματικά, δηλαδή:

z1 = 3.0 + 2.0i

όπου τα μηδενικά υποδεικνύουν ότι οι παραπάνω αριθμοί είναι δεκαδικοί.

Για να κατασκευάσουμε την toString ακολουθούμε την ίδια διαδικασία που ακολουθείται για οποιαδήποτε μέθοδο επιστρέφει τιμή. Δηλαδή:

1. Βλέπουμε ότι επιστρέφει τιμή τύπου String; Τότε αρχικά δηλώνουμε μια τέτοια μεταβλητή π.χ. s (και αν θέλουμε την αρχικοποιούμε).
2. Υπολογίζουμε την τιμή της s.
3. Επιστρέφουμε την τιμή της s με τη χρήση της πρότασης return.

Η toString καλείται όπως και κάθε άλλη μέθοδος που επιστρέφει τιμή, π.χ.

String test = z1.toString();

Μέσα στη System.out.println() καλούμε z1.toString() γιατί όλη αυτή η έκφραση αποτιμάται σε μια αλφαριθμητική τιμή (δηλαδή το αποτέλεσμά της μπορεί να αποθηκευτεί σε μια μεταβλητή τύπου String). Ωστόσο η Java, μας διευκολύνει επιτρέποντάς μας να μπορούμε να γράψουμε την έκφραση System.out.println(z1); και αυτόματα να καλείται η αντίστοιχη toString, όπως εικονίζεται στο Σχ.2.

```
1    class Complex
2    {
3          // Instance variables
4          double a = 0.0, b = 0.0;
5
6          // Constructor method with two arguments
7          Complex(double alfa, double beta)
8          {
9                a = alfa;
10               b = beta;
11         }
12
13         // Constructor method with no arguments
14         Complex()
15         {
16         }
17
18         // A standard method to return the object as a String
19         public String toString()
20         {
21               String s;
22               s = a + "+" + b + "i";
23               return s;
24         }
25
26         public static void main(String args[])
```

```
27          {
28                // Calling of two argument constructor
29                Complex z1 = new Complex(3,2); // z1 = 3+2i
30
31                System.out.println(z1);
32                /*
33                   Calling of toString
34                   This here is equivalent to
35                   System.out.println(z1.toString());
36                */
37
38
39                // Similarly for z2
40                Complex z2 = new Complex(5,2); // z2 = 5+2i
41                System.out.println(z2);
42          }
43    }
```

Σχήμα 2. Κλάση Java που αναπαριστά ένα μιγαδικό αριθμό (με μεθόδους δημιουργούς και toString).

4.4 Η λέξη κλειδί this ως αντικείμενο

Τι θα συνέβαινε αν στη μέθοδο δημιουργό του Σχ.2, η οποία λαμβάνει δύο ορίσματα, ονομάζαμε τις παραμέτρους a και b, αντί για alfa και beta αντίστοιχα; Για να αποφευχθεί η διένεξη ανάμεσα στα ονόματα των τοπικών μεταβλητών a, b της μεθόδου δημιουργού και τα ονόματα a, b των μεταβλητών στιγμιότυπου της κλάσης μας χρησιμοποιείται η λέξη κλειδί this, όπως εικονίζεται στο Σχ.3.

Η λέξη κλειδί **this** περιγράφει **το τρέχον αντικείμενο της κλάσης μας**, και η έκφαση **this.a** αναφέρεται στη **μεταβλητή υπόστασης a** της κλάσης μας. Η λέξη this εννοείται ότι υπάρχει πάντα μπροστά από τις μεταβλητές υπόστασης και τις κλήσεις των μεθόδων της κλάσης μας, άλλά δεν την εμφανίζουμε παρά μόνο αν αυτό είναι απαραίτητο.

```
1     class Complex
2     {
3           // Instance variables
4           double a = 0.0, b = 0.0;
5
6           // Constructor method with two arguments
7           Complex(double a, double b)
8           {
9                 this.a = a;
10                this.b = b;
11          }
```

```
12    }
```

Σχήμα 3. Η μέθοδος δημιουργός της Complex, χρησιμοποιώντας τη λέξη κλειδί this για την επίλυση της διένεξης ανάμεσα στα ονόματα των μεταβλητών.

4.5 Παραδείγματα γνωστών κλάσεων με μεθόδους δημιουργούς

Αν και σε ένα αντικείμενο τύπου String μπορούμε για ευκολία να δίνουμε τιμές απευθείας, χρησιμοποιώντας το σύμβολο της ανάθεσης τιμής, π.χ.,

String s = "test";

κανονικά, επειδή πρόκειται για αντικείμενο μιας κλάσης, χρειάζεται επίσης κλήση της έκφρασης new, ως εξής:

String s = new String("test");

Δηλαδή, με τον τρόπο αυτό καλείται η μέθοδος δημιουργός της κλάσης String που δέχεται ένα αλφαριθμητικό όρισμα.

Ομοίως, και για τη δημιουργία ενός αντικειμένου περιβάλλουσας κλάσης, ενώ μπορούμε να γράψουμε, π.χ.,

Integer x = 5;

κανονικά μπορεί να γίνει και ως εξής:

Integer x = new Integer(5);

καλώντας έτσι τη μέθοδο δημιουργό της κλάσης Integer που δέχεται ένα μόνο ακέραιο όρισμα.

4.6 Είσοδος από το πληκτρολόγιο

Ως ένα ακόμα παράδειγμα κλάσης από το Java API με μέθοδο δημιουργό δίνεται η κλάση **Scanner**, η οποία βρίσκεται στο πακέτο **java.util**. Αν και μπορεί να χρησιμοποιηθεί γενικά για ανάγνωση αρχείων, στο παράδειγμα που ακολουθεί η κλάση Scanner χρησιμοποιείται για τη λήψη εισόδου από το χρήστη μέσω πληκτρολογίου.

Για είσοδο από το χρήστη πρέπει να ακολουθήσουμε **δύο βήματα**:

- Να δημιουργήσουμε ένα αντικείμενο της κλάσης Scanner
- Να καλέσουμε μια μέθοδο της Scanner που έχει το πρόθεμα next, π.χ. nextInt αν επιθυμούμε η είσοδος του χρήστη να είναι μια ακέραια τιμή.

Στο παράδειγμα του Σχ.4 παρουσιάζεται η χρήση της κλάσης Scanner σε μια μέθοδο main για τη λήψη τιμών από το χρήστη για τις μεταβλητές a και b ενός μιγαδικού (εννοείται πως η κλάση Complex του Σχ.2 είναι υλοποιημένη και βρίσκεται στον ίδιο φάκελο με την κλάση Test).

```
1    import java.util.Scanner;
2
3    class Test
4    {
5          public static void main(String args[])
6          {
7                Complex z;
8                Scanner scan = new Scanner(System.in);
9                System.out.print("Give a=");
10               double a = scan.nextDouble();
11               System.out.print("Give b=");
12               double b = scan.nextDouble();
13
14               z = new Complex(a, b);
15               System.out.println("You gave: " + z);
16         }
17   }
```

Σχήμα 4. Είσοδος ενός μιγαδικού αριθμού από το χρήστη.

Ασκήσεις

1. Να υλοποιηθούν τα παραπάνω παραδείγματα.

2. Να προστεθεί στην κλάση Complex νέα μέθοδος print η οποία θα εκτυπώνει στην κονσόλα το τρέχον αντικείμενο της κλάσης μας. Να κληθεί από τη main. Ποιο είναι το πλεονέκτημα αν χρησιμοποιήσουμε την toString αντί της print;

3. Να προστεθεί στην κλάση Complex νέα μέθοδος που να υπολογίζει το μέτρο ενός μιγαδικού αριθμού, δεδομένου ότι αυτό υπολογίζεται ως m = Math.sqrt(a*a+b*b);

4. Να προστεθεί στην κλάση Complex νέα μέθοδος add η οποία να προσθέτει στον τρέχοντα μιγαδικό, έναν νέο μιγαδικό που θα δίδεται ως όρισμα της add. Η add θα πρέπει π.χ. να καλείται ως: z1.add(z2);

5. Να υλοποιηθεί στην κλάση Complex μια νέα μέθοδος που να ζητά τις τιμές του πραγματικού και φανταστικού μέρους του μιγαδικού αριθμού από το χρήστη μέσω πληκτρολογίου.

6. Να κατασκευάσετε μια νέα κλάση 2DVector που αναπαριστά ένα ευθύγραμμο τμήμα στο επίπεδο, μεταξύ σημείων (x1, y1) και (x2, y2), όπου οι μεταβλητές x1, y1, x2, y2 είναι ακέραιες.

α) Να προστεθούν μέθοδοι δημιουργοί.

β) Να προστεθεί μια μέθοδος που να υπολογίζει το μήκος του ευθυγράμμου τμήματος.

7. Μέσα σε μια μέθοδο main να κατασκευαστεί ένα παιχνίδι που θα ζητά από το χρήστη να μαντέψει έναν ακέραιο αριθμό. Αν ο αριθμός που θα εισάγει ο χρήστης από το πληκτρολόγιο είναι ο σωστός τότε θα πρέπει να εκτυπώνεται ένα μήνυμα στην οθόνη, π.χ., "You won!", διαφορετικά να ζητείται νέος αριθμός έως ότου ο χρήστης βρει το σωστό. Σε κάθε προσπάθεια ο χρήστης θα πρέπει να καθοδηγείται από το πρόγραμμα αναφορικά με το αν ο αριθμός που έδωσε είναι μεγαλύτερος ή μικρότερος από το σωστό.

Υπόδειξη: Αρχικά θεωρείστε έναν σταθερό αριθμό τον οποίο θα εισάγετε με το χέρι σε μια μεταβλητή του κώδικα του προγράμματος. Αφού το καταφέρετε αυτό χρησιμοποιείστε τη μέθοδο Math.random για την παραγωγή ενός ψευδοτυχαίου αριθμού τον οποίο θα πρέπει να μετατρέψετε σε ακέραιο.

8. Να κατασκευαστεί το παιχνίδι της άσκησης 7 ως ολοκληρωμένη κλάση στην οποία ο προς εύρεση αριθμός θα αναπαριστάται ως μεταβλητή υπόστασης. Θα πρέπει να έχει μέθοδο δημιουργό, ξεχωριστή μέθοδο για την ερώτηση και λήψη εισόδου από το χρήστη, και μια μέθοδο toString που θα επιστρέφει τον προς

εύρεση αριθμό σε περίπτωση που ζητηθεί. Να δοθεί παράδειγμα εκτέλεσης του παιχνιδιού μέσω μιας δοκιμαστικής μεθόδου main.