Εργαστηριακή άσκηση #9

Θέμα: Πολυμορφισμός και Κληρονομικότητα

Η εργαστηριακή αυτή άσκηση αποσκοπεί στην εξοικείωση με τον πολυμορφισμό και την κληρονομικότητα. Θα κατασκευαστεί μία απλή βάση δεδομένων από σχήματα η οποία αρχικά θα υλοποιηθεί μέσω ArrayList. Για κάθε σχήμα (Shape) θα κρατηθούν τα παρακάτω στοιχεία.

- Η X-συντεταγμένη του σημείου αναφοράς (xCoord)
- Η Y-συντεταγμένη του σημείου αναφοράς (yCoord
- Το χρώμα (color)
- Το όνομα (id)

Το όνομα ενός σχήματος θα καθορίζεται αυτόματα από το σύστημα. Θα αποτελείται από το είδος του σχήματος ακολουθούμενο από τον αύξοντα αριθμό του (για τα σχήματα αυτού του είδους). Παραδείγματα ονομάτων είναι Rectagle-1, Circle-4, Triange-2, k.l.p. (Υποθέτουμε φυσικά ότι στην εφαρμογή έχουμε ορίσει τις κλάσεις για τα αντίστοιχα σχήματα.

Η κλάση Shape που υλοποιεί και περιγράφει τα βασικά χαρακτηριστικά του σχήματος θα επεκταθεί έτσι ώστε οι υποκλάσεις της να περιγράφουν συγκεκριμένους τύπους σχημάτων (παραλληλόγραμμα, κύκλους, κλπ).

Η βάση δεδομένων των σχημάτων, περιέχει απλές λειτουργίες που επιτρέπουν την προσθήκη σχημάτων στη βάση, την εκτύπωση όλων των σχημάτων που περιέχονται στη βάση, και την εκτύπωση στατιστικών στοιχείων (αριθμός σχημάτων, συνολικό εμβαδόν, συνολική περιφέρεια).

1. Χρησιμοποιώντας το BlueJ, δημιουργήστε το έργο shape_database.
2. Δημιουργήστε την κλάση Shape η οποία περιέχει τα πεδία xCoord, yCoord, color, και id. Όλα τα πεδία είναι private. Ο τύπος των πεδίων xCoord και yCoord είναι int ενώ τα πεδία color και id είναι String. Το εξ΄ ορισμού (default) χρώμα είναι το μαύρο (black) ενώ το εξ' ορισμού όνομα είναι κενό ("").
3. Να γράφουν κατασκευαστές για την κλάση Shape με την παρακάτω υπογραφή:

`Shape(int x, int y, String c)`	Κατασκευάζει σχήμα με σημείο αναφοράς το (x,y) και χρώμα c.
`Shape()`	Κατασκευάζει μαύρο σχήμα με σημείο αναφοράς το (0,0).

4. Να υλοποιηθούν οι παρακάτω μέθοδοι (με την προφανή σημασία):
   public void setColor(String newColor)
   public String getColor()
   public void setId(String newId)
   public String getId()
   public void setX(int newX)
   public void setY(int newY)
   public int getX()
   public int getY()
   public void moveTo(int newX, int newY)
5. Να υλοποιηθεί η μέθοδος **print()** η οποία τυπώνει στην έξοδο (σε ξεχωριστές γραμμές) τις συντεταγμένες του σημείου αναφοράς του σχήματος και το χρώμα του.
6. Να υλοποιηθούν οι παρακάτω μέθοδοι
   public double area()
   public double perimeter()
   οι οποίες επιστρέφουν την τιμή μηδέν. Οι μέθοδοι αυτοί θα επαναπροσδιορισθούν στις υποκλάσεις έτσι ώστε να επιστρέφουν το σωστό εμβαδόν και περίμετρο, αντίστοιχα[1].
7. Να κατασκευαστούν αντικείμενα της κλάσης Shape και να ελεγχθεί η σωστή λειτουργία των μεθόδων της κλάσης.
8. Να υλοποιηθεί η κλάση **Rectangle** η οποία επεκτείνει (extends) την κλάση Shape με τα πεδία **width, length**, και **rectangleCount**. Το πεδίο rectangleCount είναι στατικό και χρησιμοποιείται για να μετράει τον

[1] Η σωστή χρήση της java προστάζει να δηλωθούν οι μέθοδοι ως abstract έτσι ώστε να μην είναι αναγκαία η υλοποίηση τους από την κλάση shape αλλά μόνο ο προσδιορισμός της υπογραφής τους. Η δυνατότητα να δηλώνεται μία μέθοδος ως abstract θα εξεταστεί στην επόμενη εργαστηριακή άσκηση.

αριθμό των παραλληλόγραμμων που έχουν κατασκευαστεί. Η υλοποίηση του κατασκευαστή είναι η παρακάτω:

```
Rectangle(int len, int wid)
{
   super(0,0,"red");
   length=len;
   width=wid;
   rectangleCount++;
   setId("Rectangle-"+rectangleCount);
}
```

9. Να γραφεί ο εξ' ορισμού κατασκευαστής (χωρίς παραμέτρους) ο οποίος φτιάχνει ένα κόκκινο παραλληλόγραμμο διαστάσεων 1 X 1.

10. Να επαναπροσδιορισθούν οι μέθοδοι area() και perimeter() έτσι ώστε να επιστρέφουν το εμβαδόν και την περίμετρο του παραλληλόγραμμου, αντίστοιχα.

11. Να υλοποιηθεί η μέθοδος **print** της κλάσης Rectangle όπως παρακάτω:

```
public void print()
{
   System.out.println("\n---Rectange----");
   System.out.println("Id: "+ getId());
   System.out.println("Length: "+ length);
   System.out.println("Width: "+ width);
   super.print();
   System.out.println("Area: " + area());
   System.out.println("Perimeter: " + perimeter());
}
```

12. Να κατασκευαστούν αντικείμενα της κλάσης Rectangle και να ελεγχθεί η σωστή λειτουργία των μεθόδων της.

13. Να υλοποιηθεί η κλάση DatabaseOfShapes η οποία δημιουργεί μια βάση δεδομένων από σχήματα όπως παρακάτω:

```
import java.util.*;

public class DatabaseOfShapes
{
   private ArrayList db;

   DatabaseOfShapes()
   {
      db=new ArrayList();
   }

   public void add(Shape s)
   {
      db.add(s);
   }

   public void listAll()
   {
      System.out.println("\n===List of Shapes===");
      for (int i=0; i<db.size(); i++)
         ((Shape)db.get(i)).print();
   }
}//Database
```

14. Να δημιουργηθούν αντικείμενα του τύπου Rectangle, και κατόπιν να ελεγχθεί η σωστή λειτουργία των μεθόδων add() και listAll() της βάσης.

15. Να υλοποιηθεί η μέθοδος printStatistics όπως παρακάτω:

```
public void printStatistics()
{
   double totalArea=0;
   double totalPerimeter=0;

   System.out.println("\n===Database Statistics===");
```

```
    System.out.println("Number of shapes: " + db.size());

    for (int i=0; i<db.size(); i++)
       totalArea+= ((Shape)db.get(i)).area();
    System.out.println("Total Area: " + totalArea);

    for (int i=0; i<db.size(); i++)
       totalPerimeter+= ((Shape)db.get(i)).perimeter();
    System.out.println("Total perimeter: " + totalPerimeter);
  }
```

16. Να ελεγχθεί η σωστή λειτουργία της μεθόδου printStatistics.

17. Να υλοποιηθεί η κλάση **Circle** η οποία επεκτείνει (extends) την κλάση Shape με τα πεδία **radius** και **circleCount**. Το πεδίο circleCount είναι στατικό και χρησιμοποιείται για να μετράει τον αριθμό των κύκλων που έχουν κατασκευαστεί. Η κλάση Circle έχει μεθόδους (και λειτουργία) αντίστοιχες της κλάσης Rectangle.

18. Να κατασκευαστούν αντικείμενα της κλάσης Circle και να ελεγχθεί η σωστή λειτουργία των μεθόδων της κλάσης.

19. Να δημιουργηθούν αντικείμενα του τύπου Rectangle και Circle, και κατόπιν ένα αντικείμενο της κλάσης DatabaseOfShapes ετσι να ελεγχθεί η σωστή λειτουργία των μεθόδων add(), listAll() και printStatistics() της βάσης. Παρατηρήστε ότι **μετά την προσθήκη του νέου τύπου σχήματος (Circle) ο κώδικας της βάσης δεν χρειάζεται καμία αλλαγή!** Η επεκτασιμότητα αυτή οφείλεται στην κληρονομικότητα και τον πολυμορφισμό και αποτελεί το μεγάλο πλεονέκτημα του αντικειμενοστραφή προγραμματισμού.

20. Το πεδίο id της κλάσης Shape έχει δηλωθεί ως private και η πρόσβαση σε αυτό από τους κατασκευαστές των κλάσεων Rectangle και Circle γίνεται μέσω της μεθόδου setId(). Να αλλαχθεί ο κώδικας της Shape έτσι ώστε το πεδίο id να δηλωθεί ως protected. Να γίνουν οι απαιτούμενες αλλαγές στους κατασκευαστές των κλάσεων Rectangle και Circle έτσι ώστε η ανάθεση τιμών στο πεδίο να γίνεται με άμεση πρόσβαση σ' αυτό.