ΜΕΘΟΔΟΛΟΓΙΕΣ ΑΝΑΠΤΥΞΗΣ ΕΦΑΡΜΟΓΩΝ

Εργασία 2

ΠΑΝΤΕΛΕΗΜΩΝ ΠΡΩΙΟΣ
ice18390023
6ο Εξάμηνο
ice18390023@uniwa.gr

MAE04 15:00-17:00

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
UNIVERSITY OF WEST ATTICA

Υπεύθυνοι καθηγητές

ΝΙΚΗΤΑΣ ΚΑΡΑΝΙΚΟΛΑΣ
ΓΕΩΡΓΙΟΣ ΠΡΕΖΕΡΑΚΟΣ
ΠΑΥΛΙΔΗΣ ΘΕΟΔΟΣΙΟΣ

Τμήμα Μηχανικών και Πληροφορικής Υπολογιστών
17 Ιουνίου 2021

Περιεχόμενα

Κατάλογος σχημάτων

Κώδικες

1 Factorial

Το πρόγραμμα Factorial (κώδικας 1.1) δέχεται arguments, τα οποία πρέπει να είναι ακέραιος αριθμός. Στην περίπτωση που δεν είναι ακέραιος αριθμός, τότε εμφανίζει μήνυμα λάθους στο stderr και το προσπερνά. Για κάθε ένα από τους αριθμούς αυτούς, βρίσκει το παραγοντικό, το εμφανίζει στην οθόνη και παράλληλα το εκτυπώνει στο αρχείο results.txt. Επίσης, εφόσον είναι δεδομένο πως δέχεται μη αρνητικό ακέραιο, ο έλεγχος αυτός παραλήφθηκε. Ακόμα, επειδή χρησιμοποιούμαι τον τύπο long, ο μέγιστος αριθμός, δεν πρέπει να είναι πάνω από 20.

Σχήμα 1.1: Ενδεικτικό τρέξιμο του προγράμματος Factorial

```
1  import java.io.FileNotFoundException;
2  import java.io.PrintWriter;
3
4  /**
5   * Η κλάση factorial βρίσκει τα παραγοντικά
6   * όλων των arguments και τα εκτυπώνει στο αρχείο
7   * results.txt. Προσοχή, πρέπει το αποτέλεσμα να έχει
8   * μέγεθος μικρότερο του μεγέθους του long, οπότε ο μέγιστος
9   * αριθμός ως argument πρέπει να μην είναι μεγαλύτερος του 20.
10  *
11  * @author padelis
12  *
13  */
14 public class Factorial {
15
16     /**
17      * Η main ελέγχει αν υπάρχει έστω και ένα agument,
18      * αν δεν υπάρχει τότε εκτυπώνει μήνυμα και τερματίζει.
19      * Αν δεν τερματίσει θα συνεχίσει παράγοντας το αρχείο
```

```
20   * results.txt με τα αποτελέσματα και παράλληλα θα τα
21   * εκτυπώση και στην οθόνη.
22   *
23   * @param integer numbers
24   */
25  public static void main(String[] args) {
26
27      if (args.length == 0) {
28          System.out.println("There are no arguments");
29          System.exit(0);
30      }
31
32      int num;
33      long result;
34
35      // i/o
36      PrintWriter outStream = null;
37
38      try {
39          outStream = new PrintWriter("results.txt");
40
41          // if number above 20 will overflow
42          for(String tmp : args) {
43              // check if number
44              try {
45                  num = Integer.parseInt(tmp);
46                  // call factor
47                  result = factor(num);
48                  // print stdout and file
49                  outStream.printf("%-5s: %d\n",tmp+"!",result);
50                  System.out.printf("%-5s: %d\n",tmp+"!",result);
51              }
52              catch(NumberFormatException e) {
53                  // It doesn't print to the file
54                  System.out.println("This is not an integer: "
55                          + e.getMessage());
56              }
57          }
58
59          outStream.close();
60      }catch(FileNotFoundException e) {
61          System.out.println("File \"results.txt\" not found ");
62      }
63
```

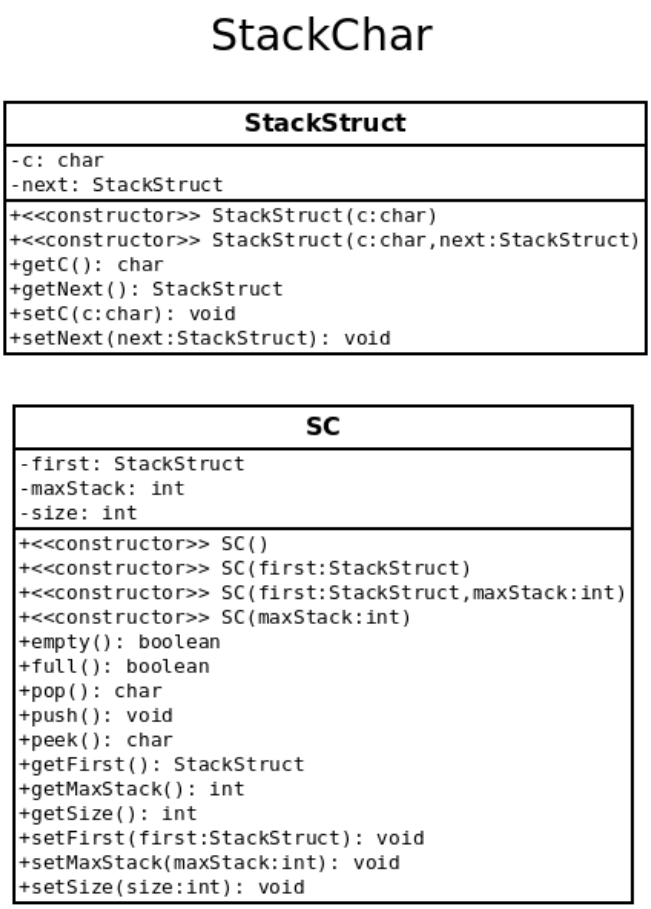
```
64    }
65
66    /**
67     *
68     * Βρίσκει το παραγοντικό του n
69     *
70     * @param n είναι ο αριθμός του factor
71     * @return επιστρέφει το παραγοντικό του n
72     */
73
74    public static long factor(int n) {
75
76        if(n == 1 || n == 0)
77            return 1;
78
79        return (long)(factor(n-1) * n);
80    }
81
82 }
```

Κώδικας 1.1: Κώδικας Factorial

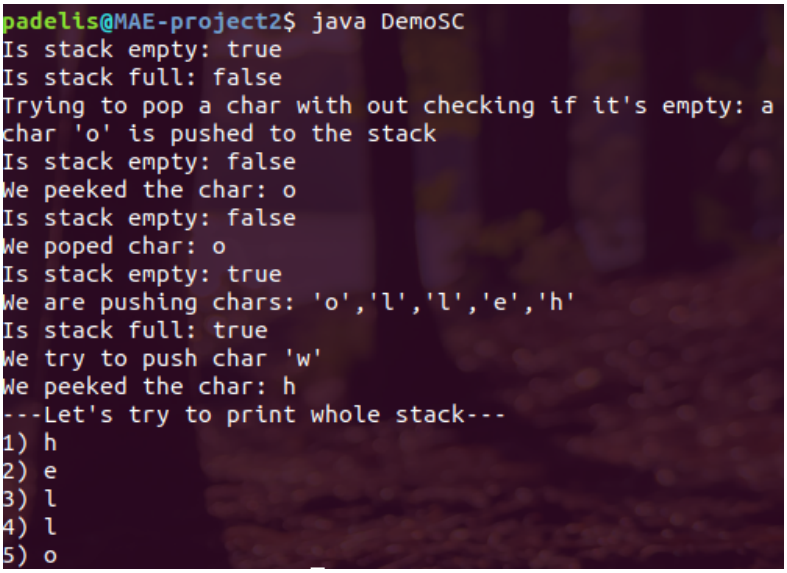
2 SC

Η κλάση StackStruct (κώδικας 2.1) είναι μία δομή δεδομένων, όπου κρατάει τον χαρακτήρα και γνωρίζει το επόμενο struct. Το πρόγραμμα SC (κώδικας 2.2) υλοποιεί μεθόδους χειρισμού του struct της μορφής FILO. Ποίο συγκεκριμένα υλοποιεί τις μεθόδους:

- empty, επιστρέφει true αν υπάρχει τουλάχιστον ένας χαρακτήρας
- full, επιστρέφει true αν το μέγεθος είναι ίδιο με το μέγιστο μέγεθος
- pop, επιστρέφει τον χαρακτήρα που βρίσκεται στην επιφάνεια και τον διαγράφει
- peek, επιστρέφει τον χαρακτήρα που βρίσκεται στην επιφάνεια χωρίς να τον διαγράψει
- push, τοποθετεί στην επιφάνεια τον χαρακτήρα που δόθηκε ως παράμετρος



Σχήμα 2.1: UML class diagram του Stack for Characters



Σχήμα 2.2: Ενδεικτικό τρέξιμο του προγράμματος Stack for Characters

```java
/**
 * Το StackStruct είναι μία δομή η οποία κρατάει την επόμενη
 * παρόμοια δομή με αυτήν την κλάση και το περιεχόμενο του,
 * δηλαδή τον χαρακτήρα.
 *
 * @author padelis
 *
 */
public class StackStruct {
    private char c;
    private StackStruct next;

    /**
     * Ο constructor αρχικοποιεί τον χαρακτήρα c με αυτόν
     * που δόθηκε και το next με null.
     *
     * @param c ο χαρακτήρας
     */
    public StackStruct(char c) {
        this.c = c;
        this.next = null;
    }

    /**
     * Ο constructor αρχικοποιεί τον χαρακτήρα c με αυτόν
```

```
     * που δόθηκε και το next με αυτό που δόθηκε.
     *
     * @param c ο χαρακτήρας
     * @param next η επόμενη δομή
     */
    public StackStruct(char c, StackStruct next) {
        this.c = c;
        this.next = next;
    }

    public char getC() {
        return c;
    }
    public StackStruct getNext() {
        return next;
    }
    public void setC(char c) {
        this.c = c;
    }
    public void setNext(StackStruct next) {
        this.next = next;
    }
}
```

Κώδικας 2.1: Κώδικας StackStruct

```
/**
 * Η κλάση SC (StackChar) είναι μια υλοποίηση νοοτροπίας FILO, όπου
 * έχει τα χαρακτηριστικά,
 * first: όπου είναι το τελευταίο στοιχείο που δόθηκε,
 * maxStack: το μέγιστο μέγεθος του stack και
 * size: το παρόν μέγεθος του stack.
 *
 * @author padelis
 *
 */
public class SC {

    private StackStruct first;
    private int maxStack;
    private int size;

    /**
     * Ο constructor αρχικοποιεί το first με null, το maxStack με 5 και
     * το size με 0.
```

```
20      */
21     public SC() {
22         first = null;
23         maxStack = 5;
24         size = 0;
25     }
26 
27     /**
28      * Καλεί το SC() και τοποθετεί ένα υπάρχον StackStruct όπου το next μπορεί να
29      * δείχνει κάπου αλλού με το πρόβλημα πως πρέπει με τα setters
30      * να αλλάξουν τα size και maxStack.
31      *
32      * @param first το πρώτο στοιχείο
33      */
34     public SC(StackStruct first) {
35         this();
36         this.first = first;
37     }
38 
39     /**
40      * Καλεί το SC() και τοποθετεί ένα υπάρχον StackStruct όπου το next μπορεί να
41      * δείχνει κάπου αλλού με το πρόβλημα πως πρέπει με το setter
42      * να αλλάξει το size.
43      *
44      * @param first το πρώτο στοιχείο
45      * @param maxStack το μέγιστο μέγεθος του stack
46      */
47     public SC(StackStruct first, int maxStack) {
48         this();
49         this.first = first;
50         this.maxStack = maxStack;
51     }
52 
53     /**
54      * Καλεί το SC() και οριοθετεί το μέγιστο μέγεθος του stack.
55      *
56      * @param maxStack οριοθετεί το μέγιστο μέγεθος του stack
57      */
58     public SC(int maxStack) {
59         this();
60         this.maxStack = maxStack;
61     }
62 
63     /**
```

```
64   * Ελέγχει αν το stack είναι άδειο.
65   *
66   * @return true αν είναι άδειο αλλιώς false
67   */
68  public boolean empty() {
69      return first == null ? true: false;
70  }
71
72  /**
73   * Ελέγχει αν το stack έχει φτάσει στο μέγιστο μέγεθος
74   *
75   * @return true αν το size είναι ίσο με το maxStack αλλιώς false
76   */
77  public boolean full() {
78      return size == maxStack ? true: false;
79  }
80
81  /**
82   * Επιστρέφει τον χαρακτήρα που μπήκε τελευταίος και τον διαγράφει.
83   * Αν το stack είναι empty τότε επιστρέφει τον χαρακτήρα 'a' για αυτό
84   * πρέπει να γίνεται έλεγχος αν είναι άδειο.
85   *
86   * @return ο χαρακτήρας της επιφάνειας
87   */
88  public char pop() {
89
90      if(empty())
91          return 'a';
92
93      char c = first.getC();
94
95      first = first.getNext();
96
97      this.size--;
98
99      return c;
100 }
101
102 /**
103  * Τοποθετεί ένα χαρακτήρα στην επιφάνεια εφόσον δεν έχει
104  * γεμίσει το stack
105  *
106  * @param c ο χαρακτήρας που θα τοποθετηθεί στην επιφάνεια
107  */
```

```
108  public void push(char c) {
109
110      if(size < maxStack) {
111          StackStruct tmp = new StackStruct(c, first);
112          this.first = tmp;
113          this.size++;
114      }
115  }
116
117  /**
118   * Επιστρέφει τον χαρακτήρα της επιφάνειας αλλά δεν τον διαγράφει.
119   *
120   * @return ο χαρακτήρας της επιφάνειας
121   */
122  public char peek() {
123
124      if (empty())
125          return 'a';
126
127      return first.getC();
128  }
129
130  public StackStruct getFirst() {
131      return first;
132  }
133
134  public int getMaxStack() {
135      return maxStack;
136  }
137
138  public int getSize() {
139      return size;
140  }
141
142  public void setFirst(StackStruct first) {
143      this.first = first;
144  }
145
146  public void setMaxStack(int maxStack) {
147      this.maxStack = maxStack;
148  }
149
150  public void setSize(int size) {
151      this.size = size;
```

```
152   }
153
154 }
```

Κώδικας 2.2: Κώδικας SC

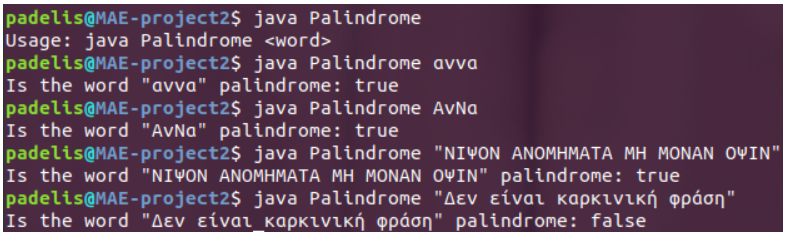
```
1
2 public class DemoSC {
3
4   public static void main(String[] args) {
5     SC test = new SC();
6
7     System.out.println("Is stack empty: " + test.empty());
8
9     System.out.println("Is stack full: " +test.full());
10
11    System.out.println("Trying to pop a char "
12          + "with out checking if it's empty: " + test.pop());
13
14    if(test.empty()) {
15      System.out.println("char 'o' is pushed to the stack");
16      test.push('o');
17    }
18
19    System.out.println("Is stack empty: " + test.empty());
20
21    System.out.println("We peeked the char: " + test.peek());
22
23    System.out.println("Is stack empty: " + test.empty());
24
25    System.out.println("We poped char: " + test.pop());
26
27    System.out.println("Is stack empty: " + test.empty());
28
29    System.out.println("We are pushing chars: 'o','l','l','e','h'");
30    test.push('o');
31
32    test.push('l');
33
34    test.push('l');
35
36    test.push('e');
37
38    test.push('h');
39
```

```
40      System.out.println("Is stack full: " + test.full());
41
42      System.out.println("We try to push char 'w'");
43      test.push('w');
44
45      System.out.println("We peeked the char: " + test.peek());
46
47      int counter = 1;
48
49      System.out.println("---Let's try to print whole stack---");
50      while(!test.empty()) {
51          System.out.println(counter + ") " + test.pop());
52          counter++;
53      }
54
55  }
56
57 }
```

Κώδικας 2.3: Κώδικας DemoSC

3 Palindrome

Το πρόγραμμα Palindrome (κώδικας 3.1), δέχεται μια παράμετρο και επιστρέφει αν είναι παλίνδρομή ή καρκινική λέξη/φράση παραλείποντας τα white spaces που μπορεί να υπάρχουν. Επίσης, χρησιμοποιεί την δομή της άσκησης StackChars (κώδικας 2.2). Ποίο συγκεκριμένα, αν ο αριθμός της λέξης/φράσης είναι περιττός, τότε το μεσαίο γράμμα παραλείπεται, σε κάθε περίπτωση όμως, δημιουργούνται 2 StackChars με μέγιστο μέγεθος το μισό μέγεθος της λέξης φράσης που δόθηκε (αφού υποστεί μερικές αλλαγές όπως lower case και διαγραφή των white spaces). Έπειτα, τοποθετούνται οι μισοί χαρακτήρες από την αρχή έως την μέση στο ένα StackChar και οι υπόλοιποι χαρακτήρες από το τέλος έως μέση στο άλλο StackChar. Τέλος, ελέγχεται σε μία δομή επανάληψης με την χρήση της μεθόδου pop, αν οι χαρακτήρες είναι ίδιοι.



Σχήμα 3.1: Ενδεικτικό τρέξιμο του προγράμματος palindrome

```java
/**
 *
 * Η κλάση palindrome ελέγχει αν μία συμβολοσειρά μπορεί
 * να διαβαστεί το ίδιο πρός τα πίσω όπως και πρός τα εμπρός.
 *
 * @author padelis
 *
 */
public class Palindrome {

    /**
     * Η main ελέγχει το argument που δώθηκε αν ικανοποιεί παλίνδρομο
     * και εκτυπώνει κατάλληλο μήνυμα.
     * Αν η συμβολοσειρά είναι περιττή, τότε ο μεσαίος χαρακτήρας
     * απλά παραλείπετε.
     *
     * @param args η συβολοσειρά πρός έλεγχο
     */
    public static void main(String[] args) {

        // check args
        if (args.length != 1) {
```

```
23          System.out.println("Usage: java Palindrome <word>");
24          System.exit(0);
25      }
26      // το len πρέπει να είναι τουλάχιστον 2
27      if (args[0].length() < 2) {
28          System.out.println("Argument must be at least a word, "
29                  + "not a single character");
30          System.exit(0);
31      }
32
33      int i = 0;
34      String tmp = args[0].toLowerCase();
35      tmp = tmp.replaceAll("\\s+", "");
36
37      int wordLen = tmp.length();
38
39      SC stack1 = new SC(wordLen/2);
40      SC stack2 = new SC(wordLen/2);
41
42
43      // Δημιουργία των stack
44      for(; i < wordLen/2; i++) {
45          stack1.push(tmp.charAt(i));
46          stack2.push(tmp.charAt(wordLen-i-1));
47      }
48
49      System.out.println("Is the word \"" + args[0]
50              + "\" palindrome: " + checkPal(stack1, stack2));
51
52
53  }
54
55  /**
56   *
57   * @param s1 το ένα StackChar που θέλουμε να ελέγξουμε
58   * @param s2 το άλλο StackChar που θέλουμε να ελέγξουμε
59   * @return επιστρέφη true αν πληροί τις προϋπόθεσης αλλιώς false
60   */
61  public static boolean checkPal(SC s1, SC s2) {
62
63      boolean result = true;
64
65      while(!s1.empty() && !s2.empty()) {
66          if(s1.pop() != s2.pop()) {
```

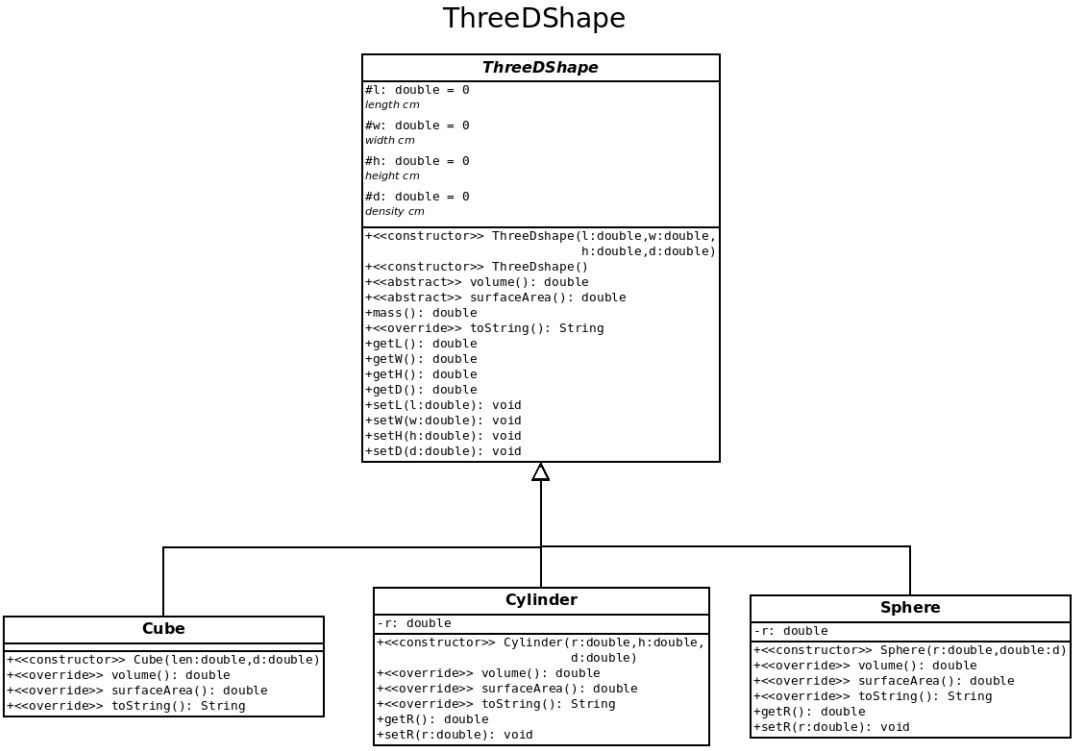
```
67              result = false;
68              break;
69          }
70      }
71
72      // Αν βγήκε από την while και κάποιο από τα δύο είναι
73      // γεμάτο τότε επιστρέφει false
74      if(!s1.empty() ^ !s2.empty()) // XOR
75          result = false;
76
77      return result;
78  }
79
80 }
```

Κώδικας 3.1: Κώδικας Palindrome

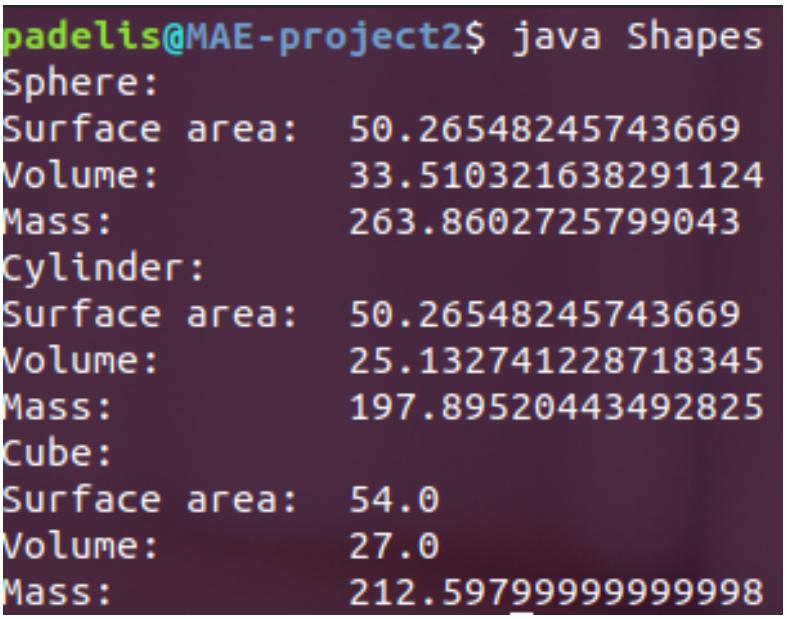
4 ThreeDShape

Ο κλάση ThreeDShape (κώδικας 4.1), είναι μια abstract κλάση με 2 abstract μεθόδους και μια υλοποιήσημη μέθοδο. Επίσης, κάνει override την μέθοδο toString από την κλάση object. Οι υπόλοιπες κλάσης υλοποιούν τις 2 abstract μεθόδους, όπως και κάνουν override ξανά την μέθοδο toString και η διαφορά είναι πως η κλάση Sphere (κώδικας 4.3) και Cylinder (κώδικας 4.4) έχουν ένα παραπάνω χαρακτηριστικό όπου είναι η ακτίνα. Επίσης, ως υλικό χρησιμοποιείται ο σίδηρος όπου έχει πυκνότητα $7.874\ g/cm^3$.

Κάποια σχόλια στους κώδικες λόγο προβλήματος ασυμβατότητας με τον compiler του LaTeXγια τα ελληνικά τυπώνονται λάθος και είναι προτιμότερο τα σχόλια να αναγνωρισθούν από τους πηγαίους κώδικες.



Σχήμα 4.1: UML class diagram των σχημάτων



Σχήμα 4.2: Ενδεικτικό τρέξιμο του προγράμματος shapes

```
1
2  abstract public class ThreeDShape {
3      protected double l; // length
4      protected double w; // width
5      protected double h; // height
6      protected double d; // density
7
8      public ThreeDShape() {
9          this.l = 0;
10         this.w = 0;
11         this.h = 0;
12         this.d = 0;
13     }
14
15     public ThreeDShape(double l, double w, double h, double d) {
16         this.l = l;
17         this.w = w;
18         this.h = h;
19         this.d = d;
20     }
21
22     /**
23      * Υπολογίζει τον όγκο.
24      * @return επιστρέφει τον όγκο σε cm*cm*cm
```

```
25      */
26     abstract public double volume();
27
28     /**
29      * Υπολογίζει την εξωτερική επιφάνεια.
30      * @return επιστρέφει την εξωτερική επιφάνεια σε cm*cm*cm
31      */
32     abstract public double surfaceArea();
33
34     /**
35      * Υπολογίζει την μάζα του αντικειμένου.
36      * @return επιστρέφει την μάζα σε g/cm*cm*cm
37      */
38     public double mass() {
39         return d * this.volume(); // m = d*v
40     }
41
42     @Override
43     public String toString() {
44         return "Surface area:  " + surfaceArea()
45                 + "\nVolume:        " + volume()
46                 + "\nMass:          " + mass();
47     }
48
49     public double getL() {
50         return l;
51     }
52
53     public double getW() {
54         return w;
55     }
56
57     public double getH() {
58         return h;
59     }
60
61     public double getD() {
62         return d;
63     }
64
65     public void setL(double l) {
66         this.l = l;
67     }
68
```

```
69    public void setW(double w) {
70        this.w = w;
71    }
72
73    public void setH(double h) {
74        this.h = h;
75    }
76
77    public void setD(double d) {
78        this.d = d;
79    }
80
81 }
```

Κώδικας 4.1: Κώδικας ThreeDShape

```
1  /**
2   * Η κλάση cube κύβος() παριστάνει έναν κύβο
3   * και κληρονομεί την κλάση ThreeDShape.
4   *
5   * @author padelis
6   *
7   */
8  public class Cube extends ThreeDShape{
9
10     /**
11      *
12      * @param len το μέγεθος της πλευράς σε cm
13      * @param d η πυκνότητα του υλικού
14      */
15     public Cube(double len, double d) {
16         super(len, len, len, d);
17     }
18
19     /**
20      * Υπολογίζει τον όγκο.
21      * @return επιστρέφει τον όγκο σε cm*cm*cm
22      */
23     public double volume() {
24         return l*w*h; // l*l*l
25     }
26
27     /**
28      * Υπολογίζει την εξωτερική επιφάνεια.
29      * @return επιστρέφει την εξωτερική επιφάνεια σε cm*cm*cm
```

```
30     */
31    public double surfaceArea() {
32        return 2*(l*w+l*h+h*w);
33    }
34
35    @Override
36    public String toString() {
37        return "Cube:\n" + super.toString();
38    }
39
40 }
```

Κώδικας 4.2: Κώδικας Cube

```
1  /**
2   * Η κλάση Sphere παριστάνει μία σφέρα
3   * και κληρονομεί την κλάση ThreeDShape.
4   *
5   * @author padelis
6   *
7   */
8  public class Sphere extends ThreeDShape{
9
10    private double r;
11
12    /**
13     *
14     * @param r radiusακτίνα/
15     * @param d densityπυκνότητα/ υλικού
16     */
17    public Sphere (double r, double d) {
18        super(2*r, 2*r, 2*r, d);
19        this.r = r;
20    }
21
22    /**
23     * Υπολογίζει τον όγκο.
24     * @return επιστρέφει τον όγκο σε cm*cm*cm
25     */
26    public double volume() {
27        return (4*Math.PI*r*r*r)/3;
28    }
29
30    /**
31     * Υπολογίζει την εξωτερική επιφάνεια.
```

```
32      * @return επιστρέφει την εξωτερική επιφάνεια σε cm*cm*cm
33      */
34     public double surfaceArea() {
35         return 4*Math.PI*r*r;
36     }
37 
38     @Override
39     public String toString() {
40         return "Sphere:\n" + super.toString();
41     }
42 
43     public double getR() {
44         return r;
45     }
46 
47     public void setR(double r) {
48         this.r = r;
49     }
50 
51 }
```

Κώδικας 4.3: Κώδικας Sphere

```
1  /**
2  *
3  * Η συνάρτηση Cylinder παριστάνει έναν κύλινδρο
4  * και κληρονομεί την κλάση ThreeDShape.
5  *
6  * @author padelis
7  *
8  */
9  public class Cylinder extends ThreeDShape{
10 
11     private double r;
12 
13     /**
14      * @param r radiusακτίνα/
15      * @param h heightύψος/ κυλίνδρου
16      * @param d densityπυκνότητα/ υλικού
17      */
18     public Cylinder (double r, double h, double d) {
19         super(2*r, 2*r, h, d);
20         this.r = r;
21     }
22 
```

```
23    /**
24     * Υπολογίζει τον όγκο.
25     * @return επιστρέφει τον όγκο σε cm*cm*cm
26     */
27    public double volume() {
28        return Math.PI*r*r*h;
29    }
30
31    /**
32     * Υπολογίζει την εξωτερική επιφάνεια.
33     * @return επιστρέφει την εξωτερική επιφάνεια σε cm*cm*cm
34     */
35    public double surfaceArea() {
36        return 2*Math.PI*r*r+2*Math.PI*r*h;
37    }
38
39    @Override
40    public String toString() {
41        return "Cylinder:\n" + super.toString();
42    }
43
44    public double getR() {
45        return r;
46    }
47
48    public void setR(double r) {
49        this.r = r;
50    }
51
52 }
```

Κώδικας 4.4: Κώδικας Cylinder

```
1
2  public class Shapes {
3
4      public static void main(String[] args) {
5
6          Sphere sphere = new Sphere(2.0, 7.874);
7          Cylinder cylinder = new Cylinder(2.0, 2, 7.874);
8          Cube cube = new Cube(3.0, 7.874);
9
10         System.out.println(sphere.toString());
11
12         System.out.println(cylinder.toString());
```

```
13
14          System.out.println(cube.toString());
15
16      }
17
18  }
```

Κώδικας 4.5: Κώδικας Shapes

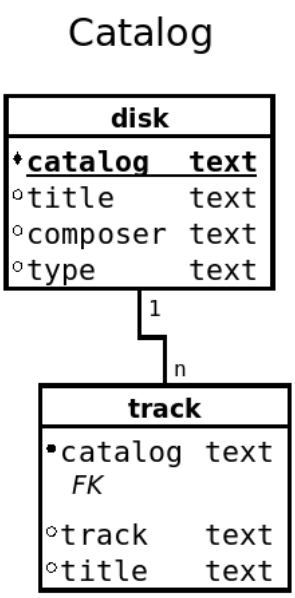
5 CDcatalog

5.1 Δημιουργία βάσης δεδομένων catalog.db

Αρχικά, δημιουργήσαμε τους πίνακες disk και track στην βάση δεδομένων όπως στην εικόνα 5.1.

```
CREATE TABLE disk(
 catalog text,
 title text UNIQUE,
 composer text,
 type text,
 PRIMARY KEY (catalog));

CREATE TABLE track(
 catalog text,
 track text,
 title text,
 FOREIGN KEY (catalog) REFERENCES disk(catalog));
```



Σχήμα 5.1: Συσχέτιση βάσης δεδομένων

Έπειτα, τοποθετούμαι τα δεδομένα όπως είναι στο παράδειγμα της άσκησης με αποτέλεσμα να έχουμε τις εγγραφές της εικόνας 5.2.

```
sqlite> select * from disk;
+---------+----------------------------------------+----------------------+-------------+
| catalog |                 title                  |       composer       |    type     |
+---------+----------------------------------------+----------------------+-------------+
| LP14    | EXILE ON MAIN ST.                      |  THE ROLLING STONES  | ROCK & ROLL |
| LP23    | COLTRANE LIVE AT THE VILLAGE VANGUARD  | JOHN COLTRANE        | JAZZ        |
| CD27    | STRING QUARTETS OPP. 131 & 132         | LUDWIG VAN BEETHOVEN | CLASSICAL   |
| LP35    | RETURN TO FOREVER                      | CHICK COREA          | JAZZ        |
| CD5     | THE BEST OF BLUE NOTE                  | VARIOUS              | BLUES       |
+---------+----------------------------------------+----------------------+-------------+
sqlite> select * from track;
+---------+---------+-----------------------------------------+
| catalog |  track  |                  title                  |
+---------+---------+-----------------------------------------+
| LP14    | D1_S1_1 | ROCKS OFF                               |
| LP14    | D1_S1_2 | RIP THIS JOINT                          |
| LP14    | S2_1    | SWEET VIRGIINIA                         |
| LP23    | S1_1    | SPIRITUAL                               |
| LP23    | S1_2    | SOFTLY AS IN A MORNING                  |
| LP35    | S1_1    | RETURN TO FOREVER                       |
| LP35    | S1_2    | CRYSTAL SILENCE                         |
| CD27    | S1_1    | ADAGIO MA NON TROPPO E MOLTO ESPRESSIVO |
| CD5     | D1_1    | SUMMERTIME                              |
+---------+---------+-----------------------------------------+
```

Σχήμα 5.2: Οι εγγραφές στους πίνακες disk και track

5.2 Κλάση διαχείρισης βάσης δεδομένων

Η κλάση CDcatalog (κώδικας 5.1), συνδέεται στην βάση δεδομένων και μπορεί να κάνει τα εξής:

- search, να αναζητήσει κάτι βάση τον τίτλο του δίσκου, τον τίτλο του κομματιού, τον συνθέτη ή τον τύπο και εμφανίζει τα αποτελέσματα σε μορφή table
- update, ενημερώνει υπάρχουσες εγγραφές των πινάκων
- insert, δημιουργεί μια νέα εγγραφή
- insertFromCsv, εισάγει πολλές εγγραφές από κάποιο αρχείο της μορφής csv
- exportToCsv, εξάγει όλες τις εγγραφές ενός πίνακα σε κάποιο αρχείο

```
1  import java.io.BufferedReader;
2  import java.io.FileNotFoundException;
3  import java.io.FileReader;
4  import java.io.IOException;
5  import java.io.PrintWriter;
6  import java.sql.Connection;
7  import java.sql.DriverManager;
8  import java.sql.ResultSet;
9  import java.sql.SQLException;
10 import java.sql.Statement;
11
12 /**
13  * Η κλάη CDcatalog διαχειρίζεται μία βάση δεδομένων με 2 πίνακες.
14  *
15  * @author padelis
```

```
16  *
17  */
18  public class CDcatalog {
19      private String dbName;
20      private Connection conn;
21      private boolean connected;
22
23      /**
24       * Αρχικοποιεί ως cbName το 'catalog.db'
25       * και προσπαθεί να συνδέθει στην βάση.
26       */
27      public CDcatalog() {
28          this.dbName= "catalog.db";
29          connectToDb();
30      }
31
32      /**
33       * Τοποθετή το όνομα dbName και προσπαθεί να συνδέθει στην βάση
34       * @param dbName όνομα αρχείου βάσης
35       */
36      public CDcatalog(String dbName) {
37          this.dbName= dbName;
38          connectToDb();
39      }
40
41      /**
42       * Προσπαθεί να συνδεθεί στην βάση και αν επιτευχθεί αλλάζει η τιμή
43       * στην μεταβλητή connected σε true.
44       */
45      public void connectToDb() {
46          try {
47              conn = DriverManager.getConnection("jdbc:sqlite:" + dbName);
48              connected = true;
49          } catch (SQLException e) {
50              System.out.println(e.toString());
51              connected = false;
52          }
53      }
54
55      /**
56       * Αναζητά στην βάση δεδομένων βάση των arguments που δώθηκαν.
57       * Αν κάποιοα/ argmuent είναι κενό τότε κάνει match με όλα.
58       * Αν βρεθούν δεδομένα τότε εκτυπώνονται σε μορφή πίνακα.
59       *
```

```
60      * @param disk ο τίτλος τους δίσκου
61      * @param track ο τίτλος του κομματίου
62      * @param comp ο συνθέτης
63      * @param type ο τύπος
64      */
65     public void search(String disk, String track, String comp, String type) {
66
67         Statement stm;
68
69         try {
70             stm = conn.createStatement();
71
72             stm.setQueryTimeout(60);
73
74             // Υπάρχει για λόγους μορφοποίησης
75             String fields[] = {"catalog","disk title","track title","composer","type","track"};
76
77             // Υπάρχει για λόγους μορφοποίησης
78             int size[] = {fields[0].length(), fields[1].length(), fields[2].length(),
79                     fields[3].length(), fields[4].length(), fields[5].length()};
80
81             // Υπάρχει για λόγους μορφοποίησης
82             ResultSet rs = stm.executeQuery("SELECT MAX(LENGTH(disk.catalog)), MAX(
        LENGTH(disk.title)),"
83                     +"MAX(LENGTH(track.title)), MAX(LENGTH(composer)), MAX(LENGTH(type))
        ,"
84                     +"MAX(LENGTH(track)) FROM disk "
85                     +"INNER JOIN track ON disk.catalog = track.catalog "
86                     +"WHERE disk.title LIKE '%" + disk + "%' AND "
87                     +"track.title LIKE '%" + track + "%' AND "
88                     +"composer LIKE '%" + comp + "%' AND "
89                     +"type LIKE '%" + type + "%';");
90
91             int i;
92
93             // Υπάρχει για λόγους μορφοποίησης
94             for(i = 0; i < size.length; i++) {
95                 if(size[i] < rs.getInt(i+1))
96                     size[i] = rs.getInt(i+1);
97             }
98
99             rs = stm.executeQuery("SELECT disk.catalog, disk.title, track.title, composer, type,
        track FROM disk "
100                    +"INNER JOIN track ON disk.catalog = track.catalog "
```

```
101             +"WHERE disk.title LIKE '%" + disk + "%' AND "
102             +"track.title LIKE '%" + track + "%' AND "
103             +"composer LIKE '%" + comp + "%' AND "
104             +"type LIKE '%" + type + "%';");
105
106         String n = "+";
107
108         int j;
109
110         // Υπάρχει για λόγους μορφοποίησης
111         for(i = 0; i < fields.length; i++) {
112             for(j = 0; j < size[i]; j++)
113                 n += '-';
114             n += '+';
115         }
116
117         System.out.println(n);
118         System.out.printf("|%-" + size[0] + "s", fields[0]);
119         System.out.printf("|%-" + size[1] + "s", fields[1]);
120         System.out.printf("|%-" + size[2] + "s", fields[2]);
121         System.out.printf("|%-" + size[3] + "s", fields[3]);
122         System.out.printf("|%-" + size[4] + "s", fields[4]);
123         System.out.printf("|%-" + size[5] + "s|\n", fields[5]);
124
125         System.out.println(n);
126
127         while(rs.next()) {
128             System.out.printf("|%-" + size[0] + "s", rs.getString(1));
129             System.out.printf("|%-" + size[1] + "s", rs.getString(2));
130             System.out.printf("|%-" + size[2] + "s", rs.getString(3));
131             System.out.printf("|%-" + size[3] + "s", rs.getString(4));
132             System.out.printf("|%-" + size[4] + "s", rs.getString(5));
133             System.out.printf("|%-" + size[5] + "s|\n", rs.getString(6));
134         }
135
136         System.out.println(n);
137
138     } catch (SQLException e) {
139         System.err.println("SQLException: " + e.toString());
140     }
141 }
142
143 /**
144  * Ανανεώνη μια ήδη υπάρχουσα εγγραφή βάση των arg, στον πίνακα που επιλέχθηκε.
```

```
145      *
146      * @param arg, αν επιλεχθή ο πίνακας disk, τότε θα πρέπει να είναι ο
147      * τίτλος, ο συνθέτης, ο τύπος και ο τίτλος που αναζητούμε. Αν πάλι
148      * επιλέχθηκε ο πίνακας track τότε θα πρέπει να είναι ο τίτλος, το κομμάτι
149      * και ο τίτλος αναζήτησης.
150      * @param table, μπορεί να πάρει 1 για τον πίνακα disk και 2 για τον track
151      * @return επιστρέφει true αν έγινε κάποια αλλαγή και false αν δεν έγινε αλλαγή
152      * ή υπήρξε error
153      */
154     public boolean update(String arg[], int table) {
155
156         Statement stm;
157         boolean result;
158
159         try {
160             stm = conn.createStatement();
161
162             stm.setQueryTimeout(60);
163
164             String q = null;
165
166             if(table == 1 && arg.length == 4)
167                 q = "update disk set title = '" + arg[0]
168                         + "', composer = '" + arg[1]
169                         + "', type = '" + arg[2]
170                         + "' where title LIKE '" + arg[3] + "';";
171             else if (table == 2 && arg.length == 3)
172                 q = "update track set title = '" + arg[0]
173                         + "', track = '" + arg[1]
174                         + "' where title LIKE '" + arg[2] + "';";
175
176             result = stm.executeUpdate(q) > 0 ? true : false;
177
178         } catch (SQLException e) {
179             System.err.println("Unsuccesfull update");
180             result = false;
181         }
182         return result;
183     }
184
185     /**
186      * Δημιουργεί μια νέα εγγραφή βάση των arg, στον πίνακα που επιλέχθηκε.
187      *
188      * @param arg, αν επιλεχθή ο πίνακας disk, τότε θα πρέπει να είναι ο
```

```
189     * κατάλογος, ο τίτλος, ο συνθέτης και ο τύπος. Αν πάλι
190     * επιλέχθηκε ο πίνακας track τότε θα πρέπει να είναι ο κατάλογος,
191     * το κομμάτι και ο τίτλος.
192     *
193     * @param table, μπορεί να πάρει 1 για τον πίνακα disk και 2 για τον track
194     * @return επιστρέφει true αν έγινε η νέα εγγραφή αλλιώς false
195     */
196    public boolean insert(String arg[], int table) {
197
198        Statement stm;
199        boolean result;
200
201        try {
202            stm = conn.createStatement();
203
204            stm.setQueryTimeout(60);
205
206            String q = null;
207
208            if(table == 1 && arg.length == 4)
209                q = "INSERT INTO disk (catalog, title, composer, type) VALUES('"
210                        + arg[0] + "','"
211                        + arg[1] + "','"
212                        + arg[2] + "','"
213                        + arg[3] + "');";
214            else if (table == 2 && arg.length == 3)
215                q = "INSERT INTO track (catalog, track, title) VALUES('"
216                        + arg[0] + "','"
217                        + arg[1] + "','"
218                        + arg[2] + "');";
219
220            result = stm.executeUpdate(q) > 0 ? true : false;
221
222        } catch (SQLException e) {
223            System.err.println("Unsuccesfull insert");
224            result = false;
225        }
226        return result;
227    }
228
229    /**
230     * Κάνει εισαγωγή εγγραφών από ένα csv αρχείο
231     *
232     * @param file όνομα csv αρχείου
```

```
233     * @param table 1 για τον πίνακα disk και 2 για τον πίνακα track
234     */
235    public void insertFromCsv(String file, int table) {
236
237        try {
238            BufferedReader inStream = new BufferedReader(new FileReader(file));
239            String line;
240            String row[];
241
242            while( (line = inStream.readLine()) != null) {
243                row = line.split(",");
244                this.insert(row, table);
245            }
246
247            inStream.close();
248        }catch (FileNotFoundException e) {
249            System.out.println("File "+file+" not found");
250        }catch(IOException e) {
251            e.toString();
252        }
253    }
254
255    /**
256     * Παράγει ένα αρχείο με τον πίνακα που επιλέχθηκε να γίνει export.
257     * @param file το αρχείο output μορφής csv
258     * @param table 1 για τον πίνακα disk και 2 για τον πίνακα track
259     */
260    public void exportToCsv(String file, int table) {
261
262        String q = null;
263        String line = null;
264        Statement stm;
265
266        if(table == 1)
267            q = "SELECT catalog, title, composer, type FROM disk;";
268        else if (table == 2)
269            q = "SELECT catalog, track, title FROM track;";
270
271        try {
272
273            PrintWriter pw = new PrintWriter(file);
274
275            stm = conn.createStatement();
276
```

```
277         stm.setQueryTimeout(60);
278
279         ResultSet rs = stm.executeQuery(q);
280
281         while(rs.next()) {
282             if(table == 1)
283                 line = rs.getString(1) + ","
284                         + rs.getString(2) + ","
285                         + rs.getString(3) + ","
286                         + rs.getString(4);
287             else if (table == 2)
288                 line = rs.getString(1) + ","
289                         + rs.getString(2) + ","
290                         + rs.getString(3);
291             pw.println(line);
292         }
293
294         pw.close();
295     }catch(SQLException e) {
296         System.err.println("SQLException:" + e.toString());
297     }catch(FileNotFoundException e) {
298         System.err.println("File not found");
299     }
300 }
301
302
303 public String getDbName() {
304     return dbName;
305 }
306
307 public Connection getConn() {
308     return conn;
309 }
310
311 public boolean isConnected() {
312     return connected;
313 }
314
315 public void setDbName(String dbName) {
316     this.dbName = dbName;
317 }
318
319 public void setConn(Connection conn) {
320     this.conn = conn;
```

```
321     }
322
323     public void setConnected(boolean connected) {
324         this.connected = connected;
325     }
326
327 }
```

Κώδικας 5.1: Κώδικας CDcatalog

5.3 Ενδεικτικό τρέξιμο

Το πρόγραμμα CDcatalogTest (κώδικας 5.2), δοκιμάζει την κλάση CDcatalog. Παρατηρήστε (εικόνα 5.3) πως την δεύτερη φορά δεν θα γίνει η εγγραφή στον πίνακα disk επειδή το primary key υπάρχει. Ενώ οι εγγραφές track θα γίνουν χωρίς κανένα πρόβλημα και θα είναι duplicates. Στην εικόνα 5.4 βλέπουμε τα αρχεία που έγιναν export την δεύτερη φορά σε μορφή csv.

```
padelis@MAE-project2$ java -cp ".:sqlite-jdbc-3.27.2.1.jar" CDcatalogTest
+-------+-----------------------------------+-----------+-------------+----+-----+
|catalog|disk title                         |track title|composer     |type|track|
+-------+-----------------------------------+-----------+-------------+----+-----+
|LP23   |COLTRANE LIVE AT THE VILLAGE VANGUARD|SPIRITUAL  |JOHN COLTRANE|JAZZ|S1_1 |
+-------+-----------------------------------+-----------+-------------+----+-----+
Disk: true
Track: true
+-------+----------+-----------+------------+-------+-----+
|catalog|disk title|track title|composer    |type   |track|
+-------+----------+-----------+------------+-------+-----+
|LP15   |new title |Track 1    |new composer|unknown|S1_1 |
|LP15   |new title |Track 2    |new composer|unknown|S1_2 |
|LP15   |new title |Track 3    |new composer|unknown|S1_3 |
|LP15   |new title |Track 4    |new composer|unknown|S1_4 |
|LP15   |new title |Track 5    |new composer|unknown|S1_5 |
+-------+----------+-----------+------------+-------+-----+
padelis@MAE-project2$ java -cp ".:sqlite-jdbc-3.27.2.1.jar" CDcatalogTest catalog.sql
+-------+-----------------------------------+-----------+-------------+----+-----+
|catalog|disk title                         |track title|composer     |type|track|
+-------+-----------------------------------+-----------+-------------+----+-----+
|LP23   |COLTRANE LIVE AT THE VILLAGE VANGUARD|SPIRITUAL  |JOHN COLTRANE|JAZZ|S1_1 |
+-------+-----------------------------------+-----------+-------------+----+-----+
Unsuccesfull insert
Disk: false
Track: true
+-------+----------+-----------+------------+-------+-----+
|catalog|disk title|track title|composer    |type   |track|
+-------+----------+-----------+------------+-------+-----+
|LP15   |new title |Track 1    |new composer|unknown|S1_1 |
|LP15   |new title |Track 2    |new composer|unknown|S1_2 |
|LP15   |new title |Track 3    |new composer|unknown|S1_3 |
|LP15   |new title |Track 4    |new composer|unknown|S1_4 |
|LP15   |new title |Track 5    |new composer|unknown|S1_5 |
|LP15   |new title |Track 1    |new composer|unknown|S1_1 |
|LP15   |new title |Track 2    |new composer|unknown|S1_2 |
|LP15   |new title |Track 3    |new composer|unknown|S1_3 |
|LP15   |new title |Track 4    |new composer|unknown|S1_4 |
|LP15   |new title |Track 5    |new composer|unknown|S1_5 |
+-------+----------+-----------+------------+-------+-----+
```

Σχήμα 5.3: Ενδεικτικό τρέξιμο του CDcatalogTest

```
padelis@MAE-project2$ cat exportDisk.csv
LP14,EXILE ON MAIN ST., THE ROLLING STONES,ROCK & ROLL
LP23,COLTRANE LIVE AT THE VILLAGE VANGUARD,JOHN COLTRANE,JAZZ
CD27,STRING QUARTETS OPP. 131 & 132,LUDWIG VAN BEETHOVEN,CLASSICAL
LP35,RETURN TO FOREVER,CHICK COREA,JAZZ
CD5,THE BEST OF BLUE NOTE,VARIOUS,BLUES
LP15,new title,new composer,unknown
padelis@MAE-project2$ cat exportTrack.csv
LP14,D1_S1_1,ROCKS OFF
LP14,D1_S1_2,RIP THIS JOINT
LP14,S2_1,SWEET VIRGIINIA
LP23,S1_1,SPIRITUAL
LP23,S1_2,SOFTLY AS IN A MORNING
LP35,S1_1,RETURN TO FOREVER
LP35,S1_2,CRYSTAL SILENCE
CD27,S1_1,ADAGIO MA NON TROPPO E MOLTO ESPRESSIVO
CD5,D1_1,SUMMERTIME
LP15,S1_1,Track 1
LP15,S1_2,Track 2
LP15,S1_3,Track 3
LP15,S1_4,Track 4
LP15,S1_5,Track 5
LP15,S1_1,Track 1
LP15,S1_2,Track 2
LP15,S1_3,Track 3
LP15,S1_4,Track 4
LP15,S1_5,Track 5
```

Σχήμα 5.4: Τα αρχεία export

```
1
2  public class CDcatalogTest {
3
4    public static void main(String[] args) {
5
6      CDcatalog catalog = new CDcatalog("catalog.db");
7      catalog.search("co","sp","jo","JAZZ");
8      String arg1[] ={"LP15", "new title", "new composer", "unknown"};
9      System.out.println("Disk: " + catalog.insert(arg1, 1));
10     String arg2[] = {"LP15","S1_1","Track 1"};
11     System.out.println("Track: " + catalog.insert(arg2, 2));
12     catalog.insertFromCsv("new_tracks.csv", 2);
13     catalog.search("","","","unknown");
14     catalog.exportToCsv("exportDisk.csv", 1);
15     catalog.exportToCsv("exportTrack.csv", 2);
16   }
17
18 }
```

Κώδικας 5.2: Κώδικας CDcatalogTest