Appendix

```
1 package csfk;
2 /**
3  * @date xx
4  * Computer Science For Kids
5  * IDE - NetBeans
6  * Platform - PC
7  */
8 import java.awt.*;
9 import javax.swing.ImageIcon;
10 import javax.swing.JLabel;
11
12 public class CSFK extends EasyApp {
13
14    public static void main(String[] args) {
15       new CSFK();
16    }
17    Label Title = addLabel("Computer Science for Elementary School Students", 80, 80, 500, 50,
       this);
18    Button Teachers = addButton("For Teachers", 100, 170, 130, 60, this);
19    Button Students = addButton("For Students", 350, 170, 130, 60, this);
20    Button Exit = addButton("Exit", 100, 300, 130, 60, this);
21    Label CopyRight = addLabel("Copyright © Tbilisi 2013", 100, 370, 200, 50, this);
22    ImageIcon Icon1 = new ImageIcon(getClass().getResource("Pic1.jpg"));
23    JLabel Pic1 = addJLabel(Icon1, 350, 260, 111, 115, this);
24
25    public CSFK() {
26       setTitle("Computer Science for Kids");
27       Title.setForeground(Color.blue);
28       Title.setFont(new Font("Arial", 0, 20));
29       Teachers.setFont(new Font("Arial", 0, 15));
30       Students.setFont(new Font("Arial", 0, 15));
31       Exit.setFont(new Font("Arial", 0, 15));
32       setBounds(50, 50, 600, 450);
33       Teachers.setBackground(Color.CYAN);
34       Students.setBackground(Color.orange);
35       Exit.setBackground(Color.GRAY);
36    }
37
38    public void actions(Object source, String command) {
39       if (source == Teachers) {
40          new Teachers();
41       }
42       if (source == Students) {
43          new Students();
44       }
45       if (source == Exit) {
46          System.exit(0);
47       }
48    }
49 }
```

```
1 package csfk;
2
3 import java.awt.*;
4
5 public class Teachers extends EasyApp {
6
7    Label Title = addLabel("Teacher's Section", 50, 50, 400, 50, this);
8    Label HintA = addLabel("Activity A: Teacher should type numeric sequence, leaving one or
      two", 30, 110, 400, 25, this);
9    Label HintA1 = addLabel("boxes empty. Then type same sequence with answers.", 30, 130, 420,
      25, this);
10   Label HintB = addLabel("Activity B: Teacher should fill in all empty squares and put question",
      30, 205, 400, 25, this);
11   Label HintB1 = addLabel("mark where it is neccessary.", 30, 225, 400, 25, this);
12   Label HintB2 = addLabel("Then teacher should hit enter and then write numbers instead of
      question marks.", 30, 245, 400, 25, this);
13   Label HintC = addLabel("Activity C: Numbers for converting will be chosen randomly by
      computer and will be checked automatically.", 30, 305, 400, 25, this);
14   Label HintC1 = addLabel("and will be checked automatically.", 30, 330, 400, 25, this);
15   Button ActivityA = addButton("Activity A", 450, 100, 120, 80, this);
16   Button ActivityB = addButton("Activity B", 450, 200, 120, 80, this);
17   Button ActivityC = addButton("Activity C", 450, 300, 120, 80, this);
18   Button Close = addButton("Close", 110, 400, 50, 25, this);
19   Button Exit = addButton("Exit", 50, 400, 50, 25, this);
20
21   public Teachers() {
22      setTitle("Teacher's Section");
23      Title.setForeground(Color.red);
24      Title.setFont(new Font("Arial", 0, 30));
25      ActivityA.setFont(new Font("Arial", 0, 15));
26      ActivityB.setFont(new Font("Arial", 0, 15));
27      ActivityC.setFont(new Font("Arial", 0, 15));
28      ActivityA.setBackground(Color.YELLOW);
29      ActivityB.setBackground(Color.MAGENTA);
30      ActivityC.setBackground(Color.green);
31      Close.setBackground(Color.GRAY);
32      setBounds(50, 50, 600, 450);
33   }
34
35   public void actions(Object source, String command) {
36      if (source == ActivityA) {
37         new tActivityA();
38      }
39      if (source == ActivityB) {
40         new tActivityB();
41      }
42      if (source == ActivityC) {
43         outputString("Tasks will be created automaticaly! \nTry student's section!");
44      }
45      if (source == Close) {
46         dispose();
47      }
48      if (source == Exit) {
```

```
49              System.exit(0);
50          }
51      }
52  }
```

```
1 package csfk;

2
3 import java.awt.*;
4 import javax.swing.ImageIcon;
5 import javax.swing.JLabel;
6
7 public class Students extends EasyApp {
8
9    Label Title = addLabel("Student's Section", 50, 50, 400, 50, this);
10   Button ActivityA = addButton("Activity A", 50, 120, 120, 80, this);
11   Button ActivityB = addButton("Activity B", 190, 120, 120, 80, this);
12   Button ActivityC = addButton("Activity C", 330, 120, 120, 80, this);
13   Button Close = addButton("Close", 190, 260, 100, 50, this);
14   Button Exit = addButton("Exit", 310, 260, 100, 50, this);
15   ImageIcon Icon3 = new ImageIcon(getClass().getResource("Pic3.png"));
16   JLabel Pic3 = addJLabel(Icon3, 70, 200, 105, 142, this);
17
18   public Students() {
19      setTitle("Student's Section");
20      setBounds(100, 100, 500, 360);
21      Title.setForeground(Color.red);
22      Title.setFont(new Font("Arial", 0, 30));
23      ActivityA.setFont(new Font("Arial", 0, 15));
24      ActivityB.setFont(new Font("Arial", 0, 15));
25      ActivityC.setFont(new Font("Arial", 0, 15));
26      ActivityA.setBackground(Color.YELLOW);
27      ActivityB.setBackground(Color.MAGENTA);
28      ActivityC.setBackground(Color.green);
29      Close.setBackground(Color.GRAY);
30   }
31
32   public void actions(Object source, String command) {
33      if (source == ActivityA) {
34         new sActivityA();
35      }
36      if (source == ActivityB) {
37         new sActivityB();
38      }
39      if (source == ActivityC) {
40         new sActivityC();
41      }
42      if (source == Close) {
43         dispose();
44      }
45      if (source == Exit) {
46         System.exit(0);
47      }
48   }
49 }
```

```
1 package csfk;

 2
 3 import java.awt.*;
 4 import java.io.RandomAccessFile;
 5 import java.io.IOException;
 6
 7 public class tActivityA extends EasyApp {
 8
 9    Label Title = addLabel("Activity A - Numeric Sequence", 50, 50, 400, 50, this);
10    Label Question = addLabel("Type Sequence:", 50, 100, 400, 50, this);
11    TextField T1 = addTextField("", 50, 150, 50, 25, this);
12    TextField T2 = addTextField("", 110, 150, 50, 25, this);
13    TextField T3 = addTextField("", 170, 150, 50, 25, this);
14    TextField T4 = addTextField("", 230, 150, 50, 25, this);
15    TextField T5 = addTextField("", 290, 150, 50, 25, this);
16    TextField T6 = addTextField("", 350, 150, 50, 25, this);
17    TextField T7 = addTextField("", 410, 150, 50, 25, this);
18    TextField T8 = addTextField("", 470, 150, 50, 25, this);
19    Label Correct = addLabel("Type Correct Answers:", 50, 200, 200, 50, this);
20    TextField C1 = addTextField("", 50, 260, 50, 25, this);
21    TextField C2 = addTextField("", 110, 260, 50, 25, this);
22    TextField C3 = addTextField("", 170, 260, 50, 25, this);
23    TextField C4 = addTextField("", 230, 260, 50, 25, this);
24    TextField C5 = addTextField("", 290, 260, 50, 25, this);
25    TextField C6 = addTextField("", 350, 260, 50, 25, this);
26    TextField C7 = addTextField("", 410, 260, 50, 25, this);
27    TextField C8 = addTextField("", 470, 260, 50, 25, this);
28    Button OK = addButton("OK", 470, 310, 50, 25, this);
29    Button EditB = addButton("Edit", 600, 400, 50, 25, this);
30    Label TextList = addLabel("List of Current Activities: ", 600, 50, 200, 50, this);
31    Button Refresh = addButton("Refresh", 900, 70, 50, 25, this);
32    List List = addList("", 600, 120, 350, 250, this);
33    Button Delete = addButton("Delete", 670, 400, 50, 25, this);
34    Button Close = addButton("Close", 830, 400, 50, 25, this);
35    Button Exit = addButton("Exit", 900, 400, 50, 25, this);
36
37    public tActivityA() {
38       setTitle("Activity A");
39       Title.setForeground(Color.blue);
40       Title.setFont(new Font("Arial", 0, 20));
41       setBounds(50, 50, 1000, 470);
42
43       try {
44          RandomAccessFile tActivityA = new RandomAccessFile("ActivityA.txt", "rw");
45          if (tActivityA.length() == 0) {
46             tActivityA.writeBytes("List of current sequences:\n");
47             // If File is empty Java will write List Title on the first line
48          }
49       } catch (IOException e) {
50          e.getMessage();
51       }
52    }
53
54    public void actions(Object source, String command) {
```

```
55      if (source == OK) {
56         addActivity();
57      }
58      if (source == Refresh) {
59         Refresh();
60      }
61      if (source == EditB) {
62         EditB();
63      }
64      if (source == Delete) {
65         Delete();
66      }
67      if (source == Close) {
68         dispose();
69      }
70      if (source == Exit) {
71         System.exit(0);
72      }
73   }
74
75   //description of the methods
76   public void addActivity() {
77
78      try {
79         RandomAccessFile tActivityA = new RandomAccessFile("ActivityA.txt", "rw");
80
81         String pr1 = T1.getText();
82         String pr2 = T2.getText();
83         String pr3 = T3.getText();
84         String pr4 = T4.getText();
85         String pr5 = T5.getText();
86         String pr6 = T6.getText();
87         String pr7 = T7.getText();
88         String pr8 = T8.getText();
89         String problem = pr1 + " " + pr2 + " " + pr3 + " " + pr4 + " " + pr5 + " " + pr6 + " " + pr7 +
           " " + pr8;
90
91         String an1 = C1.getText();
92         String an2 = C2.getText();
93         String an3 = C3.getText();
94         String an4 = C4.getText();
95         String an5 = C5.getText();
96         String an6 = C6.getText();
97         String an7 = C7.getText();
98         String an8 = C8.getText();
99         String answer = an1 + " " + an2 + " " + an3 + " " + an4 + " " + an5 + " " + an6 + " " + an7
           + " " + an8 + " ";
100
101         if (pr1.equals("") || pr2.equals("") || pr3.equals("") || pr4.equals("") || pr5.equals("") ||
          pr6.equals("") || pr7.equals("") || pr8.equals("") || an1.equals("") || an2.equals("") ||
          an3.equals("") || an4.equals("") || an5.equals("") || an6.equals("") || an7.equals("") ||
          an8.equals("")) {
102            outputString("Error,\n" + "Please, fill all fields!");
103
104         } else {
```

```
105
106             tActivityA.seek(tActivityA.length());
107             tActivityA.writeBytes(problem + " - " + answer + "\n");
108             Clean();
109             tActivityA.close();
110          }
111
112       } catch (IOException e) {
113          e.getMessage();
114       }
115    }
116
117    void Clean() {
118       T1.setText("");
119       T2.setText("");
120       T3.setText("");
121       T4.setText("");
122       T5.setText("");
123       T6.setText("");
124       T7.setText("");
125       T8.setText("");
126       C1.setText("");
127       C2.setText("");
128       C3.setText("");
129       C4.setText("");
130       C5.setText("");
131       C6.setText("");
132       C7.setText("");
133       C8.setText("");
134    }
135    static String[][] ListA = new String[30][16];
136    static int k; //k equals the number of rows that are fill in the array
137
138    public static void createArray() {
139       try {
140          RandomAccessFile tActivityA = new RandomAccessFile("ActivityA.txt", "rw");
141          boolean flag = true;
142          tActivityA.readLine();
143
144          int x1;
145          int x2;
146          String number;
147          k = 0;
148          while (flag) {
149             x1 = 0;
150             String problem = tActivityA.readLine();
151             if (problem != null) {
152                for (int i = 0; i < 16; i++) {
153                   if (i != 8) {
154                      x2 = problem.indexOf(" ", x1);
155                   } else {
156                      x1 = x1 + 2;
157                      x2 = problem.indexOf(" ", x1);
158                   }
159
```

```
160                     number = problem.substring(x1, x2);
161                     ListA[k][i] = number;
162                     x1 = x2 + 1;
163                 }
164                 k++;
165             } else {
166                 flag = false;
167             }
168         }
169     } catch (IOException e) {
170         e.getMessage();
171     }
172 }
173
174 public void Refresh() {
175     List.removeAll();
176     createArray();
177     boolean flag = true;
178     List.add("Problem List: ");
179     List.add("");
180
181     for (int i = 0; i < 30; i++) {
182         String line = "" + (i + 1) + " | ";
183         for (int j = 0; j < 16; j++) {
184             if (ListA[i][j] != null) {
185                 if (j != 7) {
186                     line = line + ListA[i][j] + " ";
187                 } else {
188                     line = line + ListA[i][j] + " - ";
189                 }
190             } else {
191                 flag = false;
192                 break;
193             }
194         }
195         if (flag) {
196             List.add(line);
197         }
198     }
199 }
200
201 public void EditB() {
202     String line = List.getSelectedItem();
203     int x = line.indexOf("|");
204     int index = Integer.parseInt(line.substring(0, x - 1)) - 1;
205     T1.setText(ListA[index][0]);
206     T2.setText(ListA[index][1]);
207     T3.setText(ListA[index][2]);
208     T4.setText(ListA[index][3]);
209     T5.setText(ListA[index][4]);
210     T6.setText(ListA[index][5]);
211     T7.setText(ListA[index][6]);
212     T8.setText(ListA[index][7]);
213     C1.setText(ListA[index][8]);
214     C2.setText(ListA[index][9]);
```

```
215         C3.setText(ListA[index][10]);
216         C4.setText(ListA[index][11]);
217         C5.setText(ListA[index][12]);
218         C6.setText(ListA[index][13]);
219         C7.setText(ListA[index][14]);
220         C8.setText(ListA[index][15]);
221         Delete();
222
223     }
224
225     void Delete() {
226         String wordDel = List.getSelectedItem();   //word, we need to delete from list
227         int x = wordDel.indexOf("|");
228         int index = Integer.parseInt(wordDel.substring(0, x - 1)) - 1;
229         int wordlength = wordDel.length();
230         //delete word from array
231         for (int i = index; i < k - 1; i++) {
232             for (int j = 0; j < 16; j++) {
233                 ListA[i][j] = ListA[i + 1][j];
234             }
235         }
236         //last element fill manualy
237         for (int j = 0; j < 16; j++) {
238             ListA[k - 1][j] = null;
239         }
240         try {          // change file when a word is deleted
241             RandomAccessFile tActivityA = new RandomAccessFile("ActivityA.txt", "rw");
242             tActivityA.seek(0);
243             long length = tActivityA.length(); // file length before deleting a word
244             length = length - wordlength;       // file length after deleting a word
245             tActivityA.readLine();
246
247             for (int p = 0; p < k - 1; p++) {
248                 String line = "";
249                 for (int j = 0; j < 16; j++) {
250                     if (j != 7) {
251                         line = line + ListA[p][j] + " ";
252                     } else {
253                         line = line + ListA[p][j] + " - ";
254                     }
255                 }
256
257                 tActivityA.writeBytes(line + "\n");
258             }
259             tActivityA.setLength(length + 2);    // to cut useless records
260             tActivityA.writeBytes("\n");
261         } catch (IOException e) {
262             e.getMessage();
263         }
264     }
265 }
```

```
1 package csfk;

 2
 3 import java.awt.*;
 4
 5 public class sActivityA extends EasyApp {
 6
 7    Label Title = addLabel("Activity A - Numeric Sequence", 50, 50, 300, 50, this);
 8    Label CI = addLabel("", 300, 150, 250, 50, this);
 9    Label Hint = addLabel("Description of the Game: Think and fill all boxes.", 50, 90, 800, 50,
     this);
10    Button Start = addButton("Start Game", 50, 140, 100, 40, this);
11    Label TaskN = addLabel(taskNum, 50, 180, 600, 50, this);
12    Label Task = addLabel(task, 50, 220, 600, 50, this);
13    TextField C1 = addTextField("", 50, 290, 50, 25, this);
14    TextField C2 = addTextField("", 110, 290, 50, 25, this);
15    TextField C3 = addTextField("", 170, 290, 50, 25, this);
16    TextField C4 = addTextField("", 230, 290, 50, 25, this);
17    TextField C5 = addTextField("", 290, 290, 50, 25, this);
18    TextField C6 = addTextField("", 350, 290, 50, 25, this);
19    TextField C7 = addTextField("", 410, 290, 50, 25, this);
20    TextField C8 = addTextField("", 470, 290, 50, 25, this);
21    Button Check = addButton("Check", 470, 330, 50, 25, this);
22    Button Next = addButton("Next", 470, 370, 50, 25, this);
23    Button Close = addButton("Close", 50, 370, 50, 25, this);
24    Button Exit = addButton("Exit", 110, 370, 50, 25, this);
25
26    public sActivityA() {
27       setTitle("Activity A");
28       Title.setForeground(Color.blue);
29       Title.setFont(new Font("Arial", 0, 20));
30       setBounds(50, 50, 580, 440);
31
32       CI.setFont(new Font("Arial", 0, 25));
33
34       Task.setForeground(Color.blue);
35       Task.setFont(new Font("Arial", 0, 20));
36    }
37
38    public void actions(Object source, String command) {
39       if (source == Start) {   //This button should be pressed in order to start activity
40          Start();
41       }
42       if (source == Check) {
43          Check();
44       }
45       if (source == Next) {
46          Next();
47       }
48       if (source == Close) {
49          dispose();
50       }
51       if (source == Exit) {
52          System.exit(0);
53       }
```

```
54    }
55    static String[][] ListA = new String[30][16];
56    static int k; //k equals the number of elements in the array
57    static int count = 0;
58    static String taskNum = ""; // counts task numbers and displays on the screen
59    static String task = ""; // variable which contains task as a value and diplays on the screen
60
61    void CreateArray() {
62       tActivityA.createArray();
63       ListA = tActivityA.ListA;
64       k = tActivityA.k;  // Number of activities created by teacher in Teacher's Section A
65    }
66
67    void Start() { // The first method which beggins Student Section's functuality - Activity A
68       CreateArray();
69       if (k >= 0) {
70          count = 1;
71          task = ListA[0][0] + ", " + ListA[0][1] + ", " + ListA[0][2] + ", " + ListA[0][3] + ", " +
        ListA[0][4] + ", " + ListA[0][5] + ", " + ListA[0][6] + ", " + ListA[0][7];
72          taskNum = "Task N" + count;
73
74          dispose();
75          new sActivityA();
76       } else {
77          outputString("No activity!\n Please finish or start other activity!");
78       }
79    }
80
81    void Check() {
82       String c1 = C1.getText();  // these variables contain values written by student
83       String c2 = C2.getText();
84       String c3 = C3.getText();
85       String c4 = C4.getText();
86       String c5 = C5.getText();
87       String c6 = C6.getText();
88       String c7 = C7.getText();
89       String c8 = C8.getText();
90
91       if (c1.equals("") || c2.equals("") || c3.equals("") || c4.equals("") || c5.equals("") || c6.equals("")
        || c7.equals("") || c8.equals("")) {
92          outputString("Error,\n" + "Please, fill all fields!");
93       } else {
94
95          if (c1.equals(ListA[count - 1][8]) && c2.equals(ListA[count - 1][9]) &&
        c3.equals(ListA[count - 1][10]) && c4.equals(ListA[count - 1][11]) &&
        c5.equals(ListA[count - 1][12]) && c6.equals(ListA[count - 1][13]) &&
        c7.equals(ListA[count - 1][14]) && c8.equals(ListA[count - 1][15])) {
96             CI.setText("Correct! Click next...!");
97             CI.setForeground(Color.GREEN);
98          } else {
99             CI.setText("Incorrect! Try again...!");
100            CI.setForeground(Color.RED);
101         }
102      }
103
```

```
104     }
105
106     void Next() {
107         String ci = CI.getText();
108         if (ci.equals("Correct! Click next...!")) {
109
110             task = ListA[count][0] + ", " + ListA[count][1] + ", " + ListA[count][2] + ", " +
                ListA[count][3] + ", " + ListA[count][4] + ", " + ListA[count][5] + ", " + ListA[count][6] +
                ", " + ListA[count][7];
111             count = count + 1;
112             taskNum = "Task N" + count;
113             dispose();
114             if (count <= k) {
115                 new sActivityA();
116             } else {
117                 outputString("Game Over!\n Well Done!");
118             }
119         } else {
120             outputString("Please, try again!");
121         }
122     }
123 }
```

```
1  package csfk;
2
3  import java.awt.*;
4  import java.io.RandomAccessFile;
5  import java.io.IOException;
6
7  public class tActivityB extends EasyApp {
8
9     Label Title = addLabel("Activity B - Super Smart Squares", 50, 50, 400, 50, this);
10    TextField T1 = addTextField("", 50, 130, 50, 25, this);
11    TextField T2 = addTextField("", 110, 130, 50, 25, this);
12    TextField T3 = addTextField("", 170, 130, 50, 25, this);
13    TextField T4 = addTextField("", 230, 130, 50, 25, this);
14    TextField T5 = addTextField("", 290, 130, 50, 25, this);
15    TextField T6 = addTextField("", 50, 160, 50, 25, this);
16    TextField T7 = addTextField("", 110, 160, 50, 25, this);
17    TextField T8 = addTextField("", 170, 160, 50, 25, this);
18    TextField T9 = addTextField("", 230, 160, 50, 25, this);
19    TextField T10 = addTextField("", 290, 160, 50, 25, this);
20    TextField T11 = addTextField("", 50, 190, 50, 25, this);
21    TextField T12 = addTextField("", 110, 190, 50, 25, this);
22    TextField T13 = addTextField("", 170, 190, 50, 25, this);
23    TextField T14 = addTextField("", 230, 190, 50, 25, this);
24    TextField T15 = addTextField("", 290, 190, 50, 25, this);
25    TextField T16 = addTextField("", 50, 220, 50, 25, this);
26    TextField T17 = addTextField("", 110, 220, 50, 25, this);
27    TextField T18 = addTextField("", 170, 220, 50, 25, this);
28    TextField T19 = addTextField("", 230, 220, 50, 25, this);
29    TextField T20 = addTextField("", 290, 220, 50, 25, this);
30    TextField T21 = addTextField("", 50, 250, 50, 25, this);
31    TextField T22 = addTextField("", 110, 250, 50, 25, this);
32    TextField T23 = addTextField("", 170, 250, 50, 25, this);
33    TextField T24 = addTextField("", 230, 250, 50, 25, this);
34    TextField T25 = addTextField("", 290, 250, 50, 25, this);
35    Label ListName = addLabel("List of current Activities:", 400, 80, 200, 50, this);
36    List List = addList("", 400, 130, 350, 250, this);
37    Button CreateT = addButton("Create Task", 130, 300, 100, 25, this);
38    Button CreateA = addButton("Create Answers", 240, 300, 100, 25, this);
39    Button Refresh = addButton("Refresh", 700, 80, 50, 25, this);
40    Button Edit = addButton("Edit", 400, 400, 50, 25, this);
41    Button Delete = addButton("Delete", 460, 400, 50, 25, this);
42    Button Close = addButton("Close", 640, 400, 50, 25, this);
43    Button Exit = addButton("Exit", 700, 400, 50, 25, this);
44    Label note1 = addLabel("", 40, 350, 300, 25, this); //inform user when button "Edit" will be
      used
45    Label note2 = addLabel("", 40, 375, 300, 25, this);
46
47    public tActivityB() {
48       setTitle("Activity B");
49       Title.setForeground(Color.blue);
50       Title.setFont(new Font("Arial", 0, 20));
51       setBounds(50, 50, 800, 450);
52    }
53
```

```
54    public void actions(Object source, String command) {
55       if (source == CreateT) {
56          CreateT();
57       }
58       if (source == CreateA) {
59          CreateA();
60       }
61       if (source == Edit) {
62          Edit();
63       }
64       if (source == Delete) {
65          Delete();
66       }
67       if (source == Refresh) {
68          Refresh();
69       }
70       if (source == Close) {
71          dispose();
72       }
73       if (source == Exit) {
74          System.exit(0);
75       }
76    }
77    // variables declaration
78    String[][] arrayT = new String[5][5]; //for temporary tasks to create them
79    String[][] arrayA = new String[5][5]; // for temporary answers to create them
80    int order = 0;  //till task is created; after creating the task, order=1;
81
82    // methods section
83    public void CreateT() {
84       if (order == 1) {
85          outputString("Task is alrady created; \nPlease, create answers!");
86       } else {
87          try {
88             RandomAccessFile tActivityB = new RandomAccessFile("ActivityB.txt", "rw");
89             arrayT[0][0] = T1.getText();
90             arrayT[0][1] = T2.getText();
91             arrayT[0][2] = T3.getText();
92             arrayT[0][3] = T4.getText();
93             arrayT[0][4] = T5.getText();
94             arrayT[1][0] = T6.getText();
95             arrayT[1][1] = T7.getText();
96             arrayT[1][2] = T8.getText();
97             arrayT[1][3] = T9.getText();
98             arrayT[1][4] = T10.getText();
99             arrayT[2][0] = T11.getText();
100            arrayT[2][1] = T12.getText();
101            arrayT[2][2] = T13.getText();
102            arrayT[2][3] = T14.getText();
103            arrayT[2][4] = T15.getText();
104            arrayT[3][0] = T16.getText();
105            arrayT[3][1] = T17.getText();
106            arrayT[3][2] = T18.getText();
107            arrayT[3][3] = T19.getText();
108            arrayT[3][4] = T20.getText();
```

```
109             arrayT[4][0] = T21.getText();
110             arrayT[4][1] = T22.getText();
111             arrayT[4][2] = T23.getText();
112             arrayT[4][3] = T24.getText();
113             arrayT[4][4] = T25.getText();
114             boolean flag = true;
115             String problem = "";
116             outerloop:
117             for (int i = 0; i < 5; i++) {
118                 for (int j = 0; j < 5; j++) {
119                     if (arrayT[i][j].equals("")) {
120                         outputString("Error,\n" + "Please, fill all fields!");
121                         flag = false;
122                         break outerloop;
123                     }
124                 }
125             }
126 
127             if (flag) {
128                 for (int i = 0; i < 5; i++) {
129                     for (int j = 0; j < 5; j++) {
130                         problem = problem + arrayT[i][j] + " ";
131                     }
132                 }
133 
134                 tActivityB.seek(tActivityB.length());
135                 tActivityB.writeBytes("Problem: " + problem + "\n");
136                 tActivityB.close();
137                 order = 1;
138             }
139 
140         } catch (IOException e) {
141             e.getMessage();
142         }
143     }
144 }
145 
146 public void CreateA() {
147     if (order == 1) {  //if the condition is true, it means the Task is already created.
148         try {
149             RandomAccessFile tActivityB = new RandomAccessFile("ActivityB.txt", "rw");
150             arrayA[0][0] = T1.getText();
151             arrayA[0][1] = T2.getText();
152             arrayA[0][2] = T3.getText();
153             arrayA[0][3] = T4.getText();
154             arrayA[0][4] = T5.getText();
155             arrayA[1][0] = T6.getText();
156             arrayA[1][1] = T7.getText();
157             arrayA[1][2] = T8.getText();
158             arrayA[1][3] = T9.getText();
159             arrayA[1][4] = T10.getText();
160             arrayA[2][0] = T11.getText();
161             arrayA[2][1] = T12.getText();
162             arrayA[2][2] = T13.getText();
163             arrayA[2][3] = T14.getText();
```

```
164             arrayA[2][4] = T15.getText();
165             arrayA[3][0] = T16.getText();
166             arrayA[3][1] = T17.getText();
167             arrayA[3][2] = T18.getText();
168             arrayA[3][3] = T19.getText();
169             arrayA[3][4] = T20.getText();
170             arrayA[4][0] = T21.getText();
171             arrayA[4][1] = T22.getText();
172             arrayA[4][2] = T23.getText();
173             arrayA[4][3] = T24.getText();
174             arrayA[4][4] = T25.getText();
175             boolean flag = true;
176             String problem = "";
177             outerloop:
178             for (int i = 0; i < 5; i++) {
179                 for (int j = 0; j < 5; j++) {
180                     if (arrayA[i][j].equals("")) {
181                         outputString("Error,\n" + "Please, fill all fields!");
182                         flag = false;
183                         break outerloop;
184                     }
185                 }
186             }
187 
188             if (flag) {
189                 for (int i = 0; i < 5; i++) {
190                     for (int j = 0; j < 5; j++) {
191                         problem = problem + arrayA[i][j] + " ";
192                     }
193                 }
194 
195                 tActivityB.seek(tActivityB.length());
196                 tActivityB.writeBytes("Answers: " + problem + "\n");
197                 tActivityB.close();
198             }
199 
200         } catch (IOException e) {
201             e.getMessage();
202         }
203         order = 0;
204         note1.setText("");
205         note2.setText("");
206         Clean();
207     } else {
208         outputString("Please, create Task at first!");
209     }
210 }
211 int count;  // to count tasks/answers in List window
212 
213 public void Refresh() {
214     List.removeAll();
215     count = 0;
216     List.add("Problem List: ");
217     List.add("");
218 
```

```
219         try {
220             RandomAccessFile tActivityB = new RandomAccessFile("ActivityB.txt", "rw");
221             while (tActivityB.getFilePointer() != tActivityB.length()) {
222                 String problem = tActivityB.readLine();
223                 String answer = tActivityB.readLine();
224                 count++;
225                 List.add(count + " | " + problem);
226                 List.add(count + " | " + answer);
227 
228             }
229         } catch (IOException e) {
230             e.getMessage();
231         }
232     }
233 
234     void Clean() {
235         T1.setText("");
236         T2.setText("");
237         T3.setText("");
238         T4.setText("");
239         T5.setText("");
240         T6.setText("");
241         T7.setText("");
242         T8.setText("");
243         T9.setText("");
244         T10.setText("");
245         T11.setText("");
246         T12.setText("");
247         T13.setText("");
248         T14.setText("");
249         T15.setText("");
250         T16.setText("");
251         T17.setText("");
252         T18.setText("");
253         T19.setText("");
254         T20.setText("");
255         T21.setText("");
256         T22.setText("");
257         T23.setText("");
258         T24.setText("");
259         T25.setText("");
260     }
261     static String[][] paArray = new String[30][2];
262     static int countPA = 0;
263 
264     static void createArray() {
265         countPA = 0;
266         String line1 = "";
267         String line2 = "";
268         try {
269             RandomAccessFile tActivityB = new RandomAccessFile("ActivityB.txt", "rw");
270             tActivityB.seek(0);
271             while (tActivityB.length() != tActivityB.getFilePointer()) {
272                 line1 = tActivityB.readLine();
273                 line2 = tActivityB.readLine();
```

```
274             paArray[countPA][0] = line1.substring(9);
275             paArray[countPA][1] = line2.substring(9);
276             countPA++;
277          }
278       } catch (IOException e) {
279          e.getMessage();
280       }
281
282    }
283
284    void Delete() {
285       String item = List.getSelectedItem();
286       if (item.equals("") || item.equals("Problem List: ")) {
287          outputString("Select item!");
288       } else {
289          createArray();
290
291          try {
292             RandomAccessFile tActivityB = new RandomAccessFile("ActivityB.txt", "rw");
293             tActivityB.setLength(0);
294             for (int i = 0; i < countPA; i++) {
295                if (!paArray[i][0].equals(item.substring(13)) &&
    !paArray[i][1].equals(item.substring(13))) {
296                   tActivityB.writeBytes("Problem: " + paArray[i][0] + "\n");
297                   tActivityB.writeBytes("Answers: " + paArray[i][1] + "\n");
298                }
299             }
300          } catch (IOException e) {
301             e.getMessage();
302          }
303       }
304       Refresh();
305    }
306
307    void Edit() {
308       int index; // selected item is the problem or the answer? (0/1)
309       String item = List.getSelectedItem();
310       Delete();
311       if (item.substring(4, 11).equals("Problem")) {
312          index = 0; //selected item is a problem
313       } else {
314          index = 1; //selected item is an answer
315       }
316       //in the paArray Array which element to Edit
317       int itemN = Integer.parseInt(item.substring(0, item.indexOf(" "))) - 1;
318
319       int index1 = 0;
320       int index2 = paArray[itemN][index].indexOf(" ");
321       T1.setText(paArray[itemN][index].substring(0, index2));
322       index1 = index2;
323       index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
324       T2.setText(paArray[itemN][index].substring(index1 + 1, index2));
325       index1 = index2;
326       index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
327       T3.setText(paArray[itemN][index].substring(index1 + 1, index2));
```

```
328         index1 = index2;
329         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
330         T4.setText(paArray[itemN][index].substring(index1 + 1, index2));
331         index1 = index2;
332         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
333         T5.setText(paArray[itemN][index].substring(index1 + 1, index2));
334         index1 = index2;
335         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
336         T6.setText(paArray[itemN][index].substring(index1 + 1, index2));
337         index1 = index2;
338         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
339         T7.setText(paArray[itemN][index].substring(index1 + 1, index2));
340         index1 = index2;
341         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
342         T8.setText(paArray[itemN][index].substring(index1 + 1, index2));
343         index1 = index2;
344         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
345         T9.setText(paArray[itemN][index].substring(index1 + 1, index2));
346         index1 = index2;
347         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
348         T10.setText(paArray[itemN][index].substring(index1 + 1, index2));
349         index1 = index2;
350         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
351         T11.setText(paArray[itemN][index].substring(index1 + 1, index2));
352         index1 = index2;
353         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
354         T12.setText(paArray[itemN][index].substring(index1 + 1, index2));
355         index1 = index2;
356         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
357         T13.setText(paArray[itemN][index].substring(index1 + 1, index2));
358         index1 = index2;
359         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
360         T14.setText(paArray[itemN][index].substring(index1 + 1, index2));
361         index1 = index2;
362         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
363         T15.setText(paArray[itemN][index].substring(index1 + 1, index2));
364         index1 = index2;
365         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
366         T16.setText(paArray[itemN][index].substring(index1 + 1, index2));
367         index1 = index2;
368         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
369         T17.setText(paArray[itemN][index].substring(index1 + 1, index2));
370         index1 = index2;
371         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
372         T18.setText(paArray[itemN][index].substring(index1 + 1, index2));
373         index1 = index2;
374         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
375         T19.setText(paArray[itemN][index].substring(index1 + 1, index2));
376         index1 = index2;
377         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
378         T20.setText(paArray[itemN][index].substring(index1 + 1, index2));
379         index1 = index2;
380         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
381         T21.setText(paArray[itemN][index].substring(index1 + 1, index2));
382         index1 = index2;
```

```
383         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
384         T22.setText(paArray[itemN][index].substring(index1 + 1, index2));
385         index1 = index2;
386         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
387         T23.setText(paArray[itemN][index].substring(index1 + 1, index2));
388         index1 = index2;
389         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
390         T24.setText(paArray[itemN][index].substring(index1 + 1, index2));
391         index1 = index2;
392         index2 = paArray[itemN][index].indexOf(" ", index1 + 1);
393         T25.setText(paArray[itemN][index].substring(index1 + 1, index2));
394         note1.setText("*Note: 1-st create Task - use Create Task button");  //after Editing the record
            - note will be disappeared
395         note2.setText("then create Answer - use Create Answer button");
396
397     }
398 }
```

```
1 package csfk;

 2
 3 import java.awt.*;
 4
 5 public class sActivityB extends EasyApp {
 6
 7    Label Title = addLabel("Activity B - Super Smart Squares", 50, 40, 400, 50, this);
 8    TextField T1 = addTextField("", 50, 130, 50, 25, this);
 9    TextField T2 = addTextField("", 110, 130, 50, 25, this);
10     TextField T3 = addTextField("", 170, 130, 50, 25, this);
11     TextField T4 = addTextField("", 230, 130, 50, 25, this);
12     TextField T5 = addTextField("", 290, 130, 50, 25, this);
13     TextField T6 = addTextField("", 50, 160, 50, 25, this);
14     TextField T7 = addTextField("", 110, 160, 50, 25, this);
15     TextField T8 = addTextField("", 170, 160, 50, 25, this);
16     TextField T9 = addTextField("", 230, 160, 50, 25, this);
17     TextField T10 = addTextField("", 290, 160, 50, 25, this);
18     TextField T11 = addTextField("", 50, 190, 50, 25, this);
19     TextField T12 = addTextField("", 110, 190, 50, 25, this);
20     TextField T13 = addTextField("", 170, 190, 50, 25, this);
21     TextField T14 = addTextField("", 230, 190, 50, 25, this);
22     TextField T15 = addTextField("", 290, 190, 50, 25, this);
23     TextField T16 = addTextField("", 50, 220, 50, 25, this);
24     TextField T17 = addTextField("", 110, 220, 50, 25, this);
25     TextField T18 = addTextField("", 170, 220, 50, 25, this);
26     TextField T19 = addTextField("", 230, 220, 50, 25, this);
27     TextField T20 = addTextField("", 290, 220, 50, 25, this);
28     TextField T21 = addTextField("", 50, 250, 50, 25, this);
29     TextField T22 = addTextField("", 110, 250, 50, 25, this);
30     TextField T23 = addTextField("", 170, 250, 50, 25, this);
31     TextField T24 = addTextField("", 230, 250, 50, 25, this);
32     TextField T25 = addTextField("", 290, 250, 50, 25, this);
33     Label CI = addLabel("", 150, 75, 250, 50, this);
34     Button Start = addButton("Start", 50, 90, 50, 25, this);
35     Label taskNum = addLabel("", 50, 300, 50, 25, this);
36     Button Check = addButton("Check Answer", 130, 300, 100, 25, this);
37     Button Next = addButton("Next Task", 240, 300, 100, 25, this);
38     Button Close = addButton("Close", 400, 270, 50, 25, this);
39     Button Exit = addButton("Exit", 400, 300, 50, 25, this);
40
41     public sActivityB() {
42        setTitle("Activity B");
43        Title.setForeground(Color.blue);
44        Title.setFont(new Font("Arial", 0, 20));
45        setBounds(50, 50, 500, 370);
46        tActivityB.createArray(); //the program will use the array paArray in the methods: Start and
          Next
47        CI.setFont(new Font("Arial", 0, 25));
48     }
49
50     public void actions(Object source, String command) {
51        if (source == Check) {
52           Check();
53        }
```

```
54        if (source == Start) {
55           Start();
56        }
57        if (source == Next) {
58           Next();
59        }
60        if (source == Close) {
61           dispose();
62        }
63        if (source == Exit) {
64           System.exit(0);
65        }
66     }
67
68     public void taskDisplay(String taskLine) {
69        int index1 = 0;
70        int index2 = taskLine.indexOf(" ");
71        T1.setText(taskLine.substring(0, index2));
72        index1 = index2;
73        index2 = taskLine.indexOf(" ", index1 + 1);
74        T2.setText(taskLine.substring(index1 + 1, index2));
75        index1 = index2;
76        index2 = taskLine.indexOf(" ", index1 + 1);
77        T3.setText(taskLine.substring(index1 + 1, index2));
78        index1 = index2;
79        index2 = taskLine.indexOf(" ", index1 + 1);
80        T4.setText(taskLine.substring(index1 + 1, index2));
81        index1 = index2;
82        index2 = taskLine.indexOf(" ", index1 + 1);
83        T5.setText(taskLine.substring(index1 + 1, index2));
84        index1 = index2;
85        index2 = taskLine.indexOf(" ", index1 + 1);
86        T6.setText(taskLine.substring(index1 + 1, index2));
87        index1 = index2;
88        index2 = taskLine.indexOf(" ", index1 + 1);
89        T7.setText(taskLine.substring(index1 + 1, index2));
90        index1 = index2;
91        index2 = taskLine.indexOf(" ", index1 + 1);
92        T8.setText(taskLine.substring(index1 + 1, index2));
93        index1 = index2;
94        index2 = taskLine.indexOf(" ", index1 + 1);
95        T9.setText(taskLine.substring(index1 + 1, index2));
96        index1 = index2;
97        index2 = taskLine.indexOf(" ", index1 + 1);
98        T10.setText(taskLine.substring(index1 + 1, index2));
99        index1 = index2;
100       index2 = taskLine.indexOf(" ", index1 + 1);
101       T11.setText(taskLine.substring(index1 + 1, index2));
102       index1 = index2;
103       index2 = taskLine.indexOf(" ", index1 + 1);
104       T12.setText(taskLine.substring(index1 + 1, index2));
105       index1 = index2;
106       index2 = taskLine.indexOf(" ", index1 + 1);
107       T13.setText(taskLine.substring(index1 + 1, index2));
108       index1 = index2;
```

```
109         index2 = taskLine.indexOf(" ", index1 + 1);
110         T14.setText(taskLine.substring(index1 + 1, index2));
111         index1 = index2;
112         index2 = taskLine.indexOf(" ", index1 + 1);
113         T15.setText(taskLine.substring(index1 + 1, index2));
114         index1 = index2;
115         index2 = taskLine.indexOf(" ", index1 + 1);
116         T16.setText(taskLine.substring(index1 + 1, index2));
117         index1 = index2;
118         index2 = taskLine.indexOf(" ", index1 + 1);
119         T17.setText(taskLine.substring(index1 + 1, index2));
120         index1 = index2;
121         index2 = taskLine.indexOf(" ", index1 + 1);
122         T18.setText(taskLine.substring(index1 + 1, index2));
123         index1 = index2;
124         index2 = taskLine.indexOf(" ", index1 + 1);
125         T19.setText(taskLine.substring(index1 + 1, index2));
126         index1 = index2;
127         index2 = taskLine.indexOf(" ", index1 + 1);
128         T20.setText(taskLine.substring(index1 + 1, index2));
129         index1 = index2;
130         index2 = taskLine.indexOf(" ", index1 + 1);
131         T21.setText(taskLine.substring(index1 + 1, index2));
132         index1 = index2;
133         index2 = taskLine.indexOf(" ", index1 + 1);
134         T22.setText(taskLine.substring(index1 + 1, index2));
135         index1 = index2;
136         index2 = taskLine.indexOf(" ", index1 + 1);
137         T23.setText(taskLine.substring(index1 + 1, index2));
138         index1 = index2;
139         index2 = taskLine.indexOf(" ", index1 + 1);
140         T24.setText(taskLine.substring(index1 + 1, index2));
141         index1 = index2;
142         index2 = taskLine.indexOf(" ", index1 + 1);
143         T25.setText(taskLine.substring(index1 + 1, index2));
144     }
145     int count = 0; //to count tasks for student
146     String answer = "";
147     String taskLine = "";
148 
149     public void Start() { //The first method which beggins Student Section's functuality in ActivityB
150         if (tActivityB.countPA > 0) {
151             taskLine = tActivityB.paArray[0][0];
152             taskDisplay(taskLine);
153             taskNum.setText("Task N" + (count + 1));
154         } else {
155             outputString("Tasks are not created!");
156         }
157     }
158 
159     public void Check() {
160         String answerLine = tActivityB.paArray[count][1];
161         answer = T1.getText() + " " + T2.getText() + " " + T3.getText() + " " + T4.getText() + " " +
            T5.getText() + " " + T6.getText() + " " + T7.getText() + " " + T8.getText() + " " +
            T9.getText() + " " + T10.getText() + " " + T11.getText() + " " + T12.getText() + " " +
```

```
        T13.getText() + " " + T14.getText() + " " + T15.getText() + " " + T16.getText() + " " +
        T17.getText() + " " + T18.getText() + " " + T19.getText() + " " + T20.getText() + " " +
        T21.getText() + " " + T22.getText() + " " + T23.getText() + " " + T24.getText() + " " +
        T25.getText() + " ";
162     if (answer.equals(tActivityB.paArray[count][1])) {
163         CI.setText("Correct! Click next...!");
164         CI.setForeground(Color.GREEN);
165     } else {
166         CI.setText("Incorrect! Try again...!");
167         CI.setForeground(Color.RED);
168
169     }
170 }
171
172 public void Next() {
173
174     String ci = CI.getText();
175     CI.setText("");
176     if (ci.equals("Correct! Click next...!")) {
177         count++;
178         if (count < tActivityB.countPA) {
179             taskLine = tActivityB.paArray[count][0];
180             taskDisplay(taskLine);
181             taskNum.setText("Task N" + (count + 1));
182         } else {
183             outputString("Game Over!\n Well Done!");
184         }
185     } else {
186         outputString("Please, try again!");
187     }
188 }
189 }
```

```
1 package csfk;

 2
 3 import java.awt.*;
 4
 5 public class sActivityC extends EasyApp {
 6
 7    Label Title = addLabel("Converting numbers from decimal to binary system", 40, 50, 800, 50,
      this);
 8    Label Number = addLabel("", 170, 110, 50, 25, this);
 9    Button B1 = addButton("64", 50, 150, 50, 100, this);
10     Button B2 = addButton("32", 110, 150, 50, 100, this);
11     Button B3 = addButton("16", 170, 150, 50, 100, this);
12     Button B4 = addButton("8", 230, 150, 50, 100, this);
13     Button B5 = addButton("4", 290, 150, 50, 100, this);
14     Button B6 = addButton("2", 350, 150, 50, 100, this);
15     Button B7 = addButton("1", 410, 150, 50, 100, this);
16     Label L1 = addLabel("0", 70, 280, 50, 25, this);
17     Label L2 = addLabel("0", 130, 280, 50, 25, this);
18     Label L3 = addLabel("0", 190, 280, 50, 25, this);
19     Label L4 = addLabel("0", 250, 280, 50, 25, this);
20     Label L5 = addLabel("0", 310, 280, 50, 25, this);
21     Label L6 = addLabel("0", 370, 280, 50, 25, this);
22     Label L7 = addLabel("0", 430, 280, 50, 25, this);
23     Label CI = addLabel("", 350, 100, 250, 50, this);
24     Button Start = addButton("Start", 50, 110, 50, 25, this);
25     Button Check = addButton("Check", 50, 350, 50, 25, this);
26     Button Next = addButton("Next", 110, 350, 50, 25, this);
27     Button Close = addButton("Close", 350, 350, 50, 25, this);
28     Button Exit = addButton("Exit", 410, 350, 50, 25, this);
29
30     public sActivityC() {
31        setTitle("Activity C");
32        Title.setForeground(Color.blue);
33        Title.setFont(new Font("Arial", 0, 20));
34        setBounds(50, 50, 520, 420);
35        Number.setFont(new Font("Arial", 0, 20));
36        Number.setForeground(Color.getHSBColor(0.9f, 0.8f, 0.9f));
37        CI.setFont(new Font("Arial", 0, 20));
38        B1.setFont(new Font("Arial", 0, 20));
39        B2.setFont(new Font("Arial", 0, 20));
40        B3.setFont(new Font("Arial", 0, 20));
41        B4.setFont(new Font("Arial", 0, 20));
42        B5.setFont(new Font("Arial", 0, 20));
43        B6.setFont(new Font("Arial", 0, 20));
44        B7.setFont(new Font("Arial", 0, 20));
45     }
46
47     public void actions(Object source, String command) {
48        if (source == Start) {   //This button should be pressed in order to start activity
49           Start();
50        }
51        if (source == Check) {
52           Check();
53        }
```

```
54        if (source == Next) {
55           Next();
56        }
57        if (source == Close) {
58           dispose();
59        }
60        if (source == Exit) {
61           System.exit(0);
62        }
63        if (source == B1) {
64           B1();
65        }
66        if (source == B2) {
67           B2();
68        }
69        if (source == B3) {
70           B3();
71        }
72        if (source == B4) {
73           B4();
74        }
75        if (source == B5) {
76           B5();
77        }
78        if (source == B6) {
79           B6();
80        }
81        if (source == B7) {
82           B7();
83        }
84
85
86     }
87     //method implementation
88
89     void Start() {
90        int number = (int) (Math.random() * 100);
91        Number.setText(Integer.toString(number));
92     }
93
94     void B1() {
95        String b1 = L1.getText();
96        if (b1.equals("0")) {
97           L1.setText("1");
98        } else {
99           L1.setText("0");
100       }
101    }
102
103    void B2() {
104       String b2 = L2.getText();
105       if (b2.equals("0")) {
106          L2.setText("1");
107       } else {
108          L2.setText("0");
```

```
109         }
110     }
111
112     void B3() {
113         String b3 = L3.getText();
114         if (b3.equals("0")) {
115             L3.setText("1");
116         } else {
117             L3.setText("0");
118         }
119     }
120
121     void B4() {
122         String b4 = L4.getText();
123         if (b4.equals("0")) {
124             L4.setText("1");
125         } else {
126             L4.setText("0");
127         }
128     }
129
130     void B5() {
131         String b5 = L5.getText();
132         if (b5.equals("0")) {
133             L5.setText("1");
134         } else {
135             L5.setText("0");
136         }
137     }
138
139     void B6() {
140         String b6 = L6.getText();
141         if (b6.equals("0")) {
142             L6.setText("1");
143         } else {
144             L6.setText("0");
145         }
146     }
147
148     void B7() {
149         String b7 = L7.getText();
150         if (b7.equals("0")) {
151             L7.setText("1");
152         } else {
153             L7.setText("0");
154         }
155     }
156
157     void Check() {
158         int c1 = Integer.parseInt(L1.getText());
159         int c2 = Integer.parseInt(L2.getText());
160         int c3 = Integer.parseInt(L3.getText());
161         int c4 = Integer.parseInt(L4.getText());
162         int c5 = Integer.parseInt(L5.getText());
163         int c6 = Integer.parseInt(L6.getText());
```

```
164         int c7 = Integer.parseInt(L7.getText());
165
166         int answer = (int) (c1 * Math.pow(2, 6) + c2 * Math.pow(2, 5) + c3 * Math.pow(2, 4) + c4 *
            Math.pow(2, 3) + c5 * Math.pow(2, 2) + c6 * Math.pow(2, 1) + c7 * Math.pow(2, 0));
167
168         if (answer == Integer.parseInt(Number.getText())) {
169             CI.setForeground(Color.getHSBColor(0.3f, 1f, 0.7f));
170             CI.setText("Correct");
171         } else {
172             CI.setForeground(Color.getHSBColor(0f, 1f, 1f));
173             CI.setText("Incorrect");
174         }
175     }
176
177     void Next() {
178         String answer = CI.getText();
179         if (answer.equals("")) {
180             outputString("Error,\n" + "Please, try again!");
181         } else {
182             if (answer.equals("Correct")) {
183                 Start();
184                 CI.setText("");
185                 L1.setText("0");
186                 L2.setText("0");
187                 L3.setText("0");
188                 L4.setText("0");
189                 L5.setText("0");
190                 L6.setText("0");
191                 L7.setText("0");
192             } else {
193                 outputString("Error,\n" + "Please, try again!");
194             }
195         }
196     }
197 }
```