МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Уфимский Государственный Авиационный Технический Университет

Кафедра АСУ

Расчетно-графическая работа по дисциплине
«Объектно-ориентированное моделирование и
программирование»

«Депозитный калькулятор»

Выполнил:
ст. гр ПИ-209з
Марков Н.Г
Латыпов Р.А
Сагитов А.Э.
Сираев И.Р.

Проверил:
Старцев Г.В.

Уфа, 2019 г.

Содержание

1. Описание прототипа программного продукта

Для создания депозитного калькулятора была выбрана среда выполнения Eclipce;
Для работы с визуальными формами было установлено официальное расширение WindowBuilder, которое добавляет визуальную форму в которой можно редактировать Swing компоненты.

Были определены входные и выходные данные для депозитного калькулятора:

- Сумма вклада
- Дата открытия
- Процентная ставка по вкладу
- Срок вклада
- Капитализация вклада
- Периодичность капитализации

Из выходных данных требовались:

- Итоговый депозит
- Отдельно количество процентов
- Опционально таблица с периодами выплат

После анализа входных и выходных данных была собрана визуальная форма в WindowBuilder. Форма состоит из полей ввода для информации, чекбокса для выбора капитализации и ком-

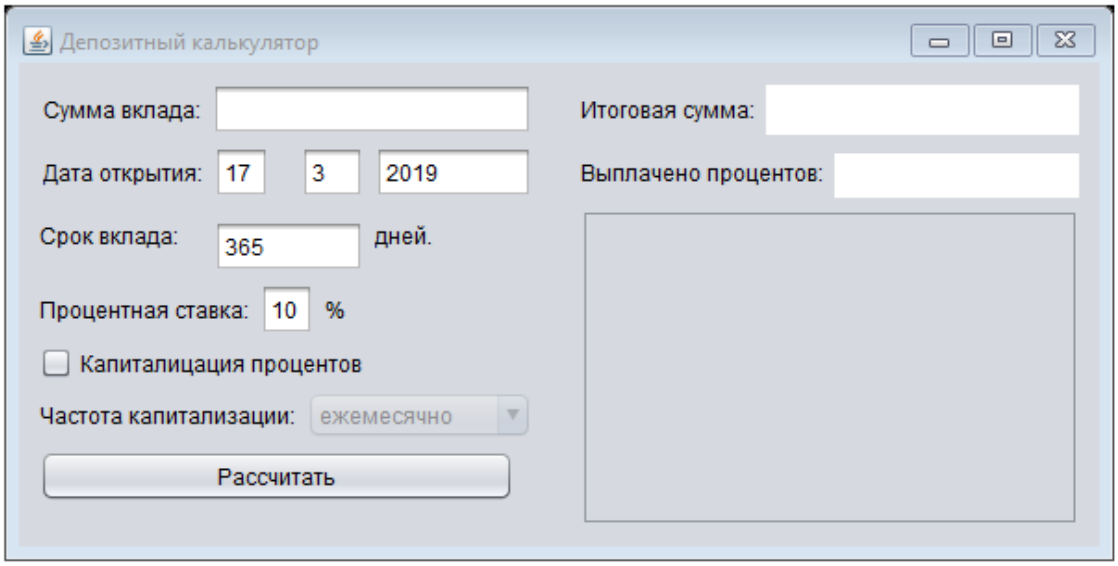

Рис. 1. Итоговый вид формы

бобокса для выбора частоты капитализации. Для того, что бы без выбора капитализации нельзя было сменить его частоту был использован следующий код:

Листинг 1. Нажатие на флажок

```
//ОБРАБОТКА ФЛАЖКА
ch_capital.addActionListener(new ActionListener() {
	public void actionPerformed(ActionEvent arg0) {
		if (ch_capital.isSelected()) //без капитализации вывод недоступен (т.к ничего не меняет)
		{c_type_proc.setEnabled(true);}else
		{c_type_proc.setEnabled(false);}
	}
});
```

Так же форма содержит место для скроллящейся таблицы.

2. Программный код

Проект состоит из 4х файлов:

- Deposit.java - основной класс, получает входные данные
- Proc.java - содержит энумератор для комбобокса
- Test.java - содержит юнит-тесты
- wind.java - основной класс UI и обработки данных

Deposit.java содержит методы

- public void calculate() - публичный метод, обрабатывает данные
- public void setDate(int date) - проверяет входит ли день в месяц, в случае ошибки (например 29е число в феврале) генерирует исключение.
- private void dayInPeriod() - считает время от от начала вклада до забора
- private double howDay() - возвращает кол-во дней в текущем месяце
- private double mathMoney() - подсчитывает проценты по формуле
- private void addCapital(double capMoney, int i) - добавляет проценты к общей сумме и вносит строку в таблицу
- private void simpleInterest() - подсчитывает простые проценты
- private void hardInterest() - подсчитывает сложные проценты

Wind.java содержит методы:

- private void initComp() - создание GUI
- private void create_event() - обработка событий
  - Нажатие на флажок - блокирует комбобокс
  - Нажатие на кнопку расчёта - проверяет данные из текстбоксов, в случае успеха передает их созданному экземпляру класса deposit;

Полный исходный код всех файлов приведен в приложении. Так же весь исходный код доступен в репозитории github: https://github.com/fauls/DCalc

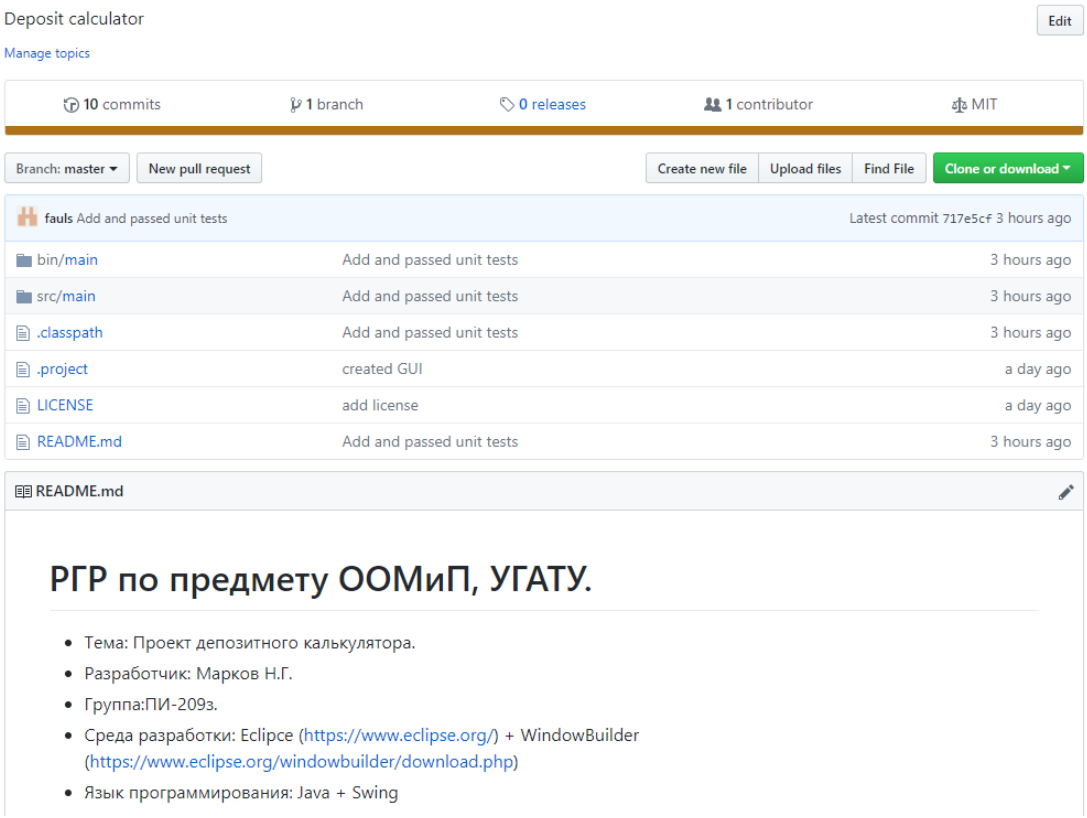

Рис. 2. Страница репозитория

3. Описание тестирования работы программы.

Для тестирования использовались JUnit5 тесты, содержащиеся в файле Test.java. В ходе тестирование создавался экземпляр класса deposit, которому передавались стартовые значения и вызывался основной метод calculate.После этого тестовой функцией assertEquals проводилось сравнение выхода программы и эталонного значения, полученного с различных онлайн-депозитных калькуляторов.

Всего было написано 8 тестовых кейсов с различными условиями, полный код юнит-тестов можно найти в приложении.

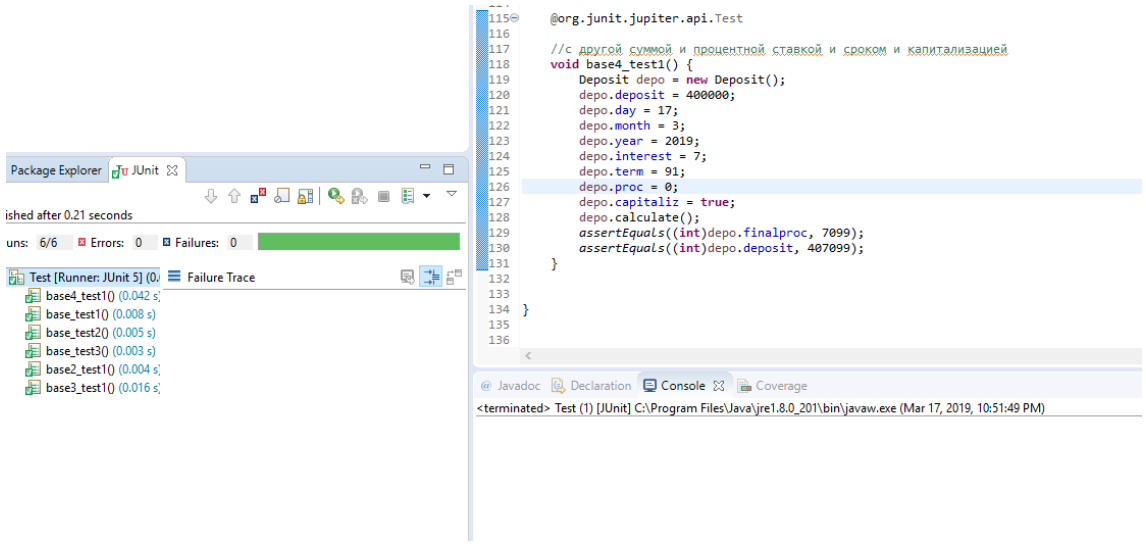

Рис. 3. Пример пройденного unit-тестирования

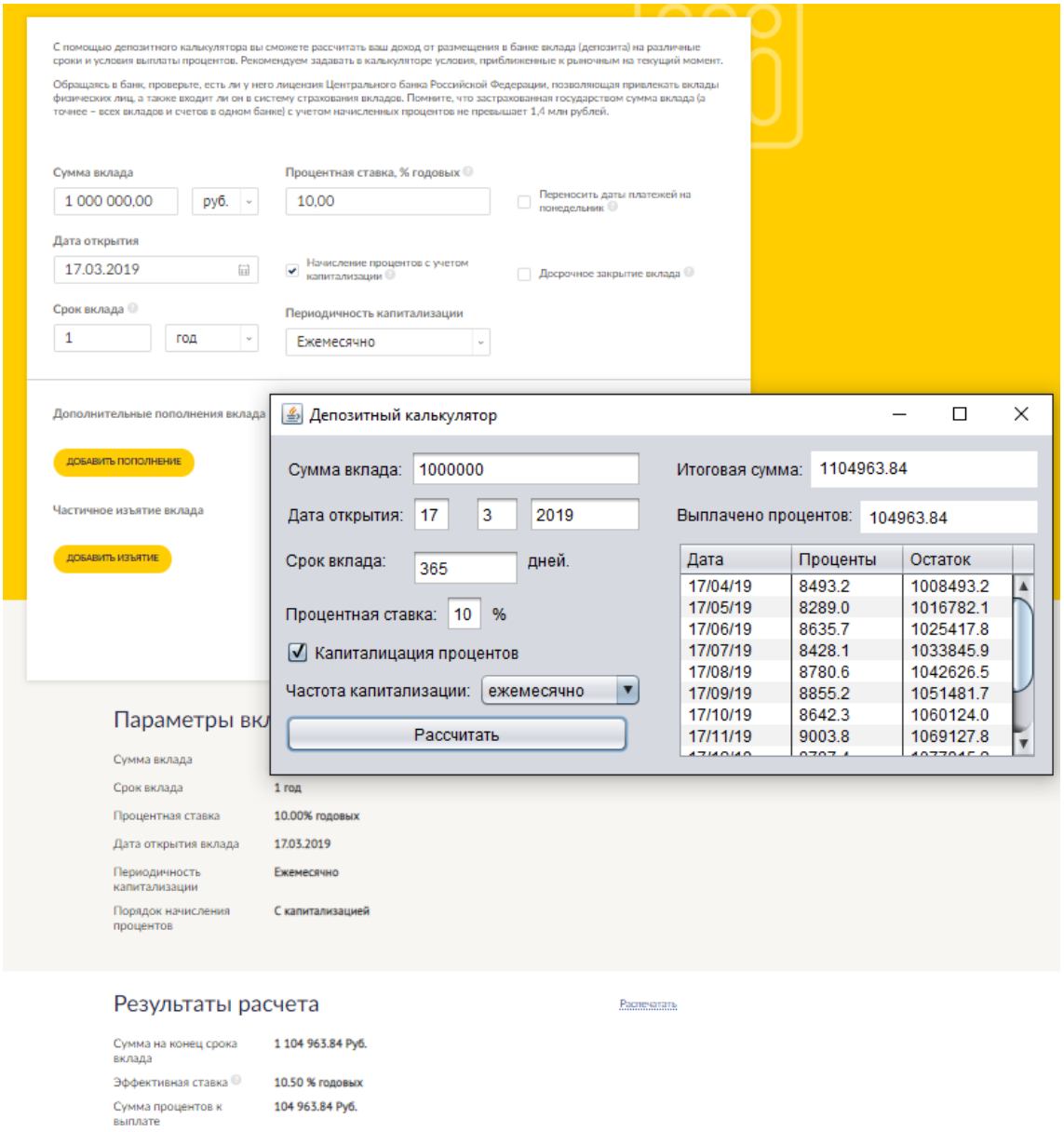

Рис. 4. Сравнение с онлайн-калькулятором

4. Приложение

4.1. Deposit.java - основной класс

Листинг 2. Основной класс

```
1  package main;
2
3  import java.time.LocalDate;
4  import java.time.format.DateTimeFormatter;
5  import java.time.temporal.ChronoUnit;
6
7  import javax.swing.JOptionPane;
8
9  //
10 //основной класс для работы с депозитом
11 //
12 public class Deposit {
13         double deposit; //сумма
14         int year;
15         int month;
16         int day;
17         int term; //срок
18         double interest; //процентная ставка
19         boolean capitaliz; //капитализация
```

```
20          int proc =0; //0 - капитализация, 1 - через 3 месяца, 2 - ежегодно
21          String[][] table; //таблица с датой, процентами и остатком
22
23          private LocalDate _startdate; //начальная дата
24          private LocalDate _stopdate; //конечная
25          private int _periods; //кол-во месяцев (проценты добавляются помесячно)
26          double finalproc=0; //итоговые проценты
27
28          public void setDate(int date)//проверка дня
29          {
30                  LocalDate startdate = LocalDate.of(year, month, 2);
31                  if (date<1 || date>startdate.lengthOfMonth())
32                  {throw new NumberFormatException();}else
33                  {day = date;}
34          }
35
36          public void calculate()
37          {
38                  dayInPeriod();//подсчёт срока
39
40                  if (capitaliz)
41                  {hardInterest();}
42                  else
43                  {simpleInterest();}//если не капитализация, то это простые проценты
44          }
45
46          private void dayInPeriod()//считаем время от вложения вклада до забора
47          {
48                  _startdate = LocalDate.of(year, month, day); //конвертируем дату из формата ввода в удобный
49                  _stopdate = _startdate;
50                  _stopdate = _stopdate.plusDays(term+1);//начальная дата + срок = финальная дата +1 т.к финальная не
                        учитывается
51                  _periods = (int) ChronoUnit.MONTHS.between(_startdate, _stopdate);//кол-во полных месяцев в периоде
52          }
53
54          private double howDay()//сколько дней в месяце
55          {
56                  return _startdate.lengthOfMonth();
57          }
58
59          private double mathMoney()//подсчёт по формуле
60          {
61                  return deposit*((interest/100)*(howDay()/_startdate.lengthOfYear())); //подневный расчет
62          }
63
64          private void addCapital(double capMoney, int i)//добавить сумму к финальной, внести строку в таблицу,
                сдвинуть дату
65          {
66                  finalproc += capMoney; //финальные проценты - сумма всех процентов
67                  switch(proc)//добавление месяцев, в зависимости от частоты вывода в таблицу
68                  {
69                  case 0: _startdate = _startdate.plusMonths(1); break;//добавление месяца
70                  case 1: _startdate = _startdate.plusMonths(3); break;
71                  case 2: _startdate = _startdate.plusYears(1); break;
72                  }
73                  table[i][0]=_startdate.format(DateTimeFormatter.ofPattern("d/MM/YY"));//форматированный вывод даты в
                        первую колонку
74                  table[i][1]=String.format("%.1f",capMoney); //формат до 2х знаков после запятой
75                  table[i][2]=String.format("%.1f",deposit);
76          }
77
78          private void simpleInterest()//простые проценты, для них выдача/невыдача денег не важна, так что proc
                игнорируется
79          {
80
81                  table = new String[_periods][3]; //инициализация таблицы
82
83                  for(int i=0; i<_periods;i++) //добавление процентов каждый период
84                  {addCapital(mathMoney(), i); //считаем
85                  }
86                  deposit+=finalproc; //к итогу добавляем проценты
87          }
88
89          private void hardInterest()//сложные проценты
90          {
91                  table = new String[_periods][3]; //инициализация таблицы
92                  double capMoney = 0; //накопление процентов за период
93                  int clock = 0; //для отсчёта периодов
94                  int period =0; //для ровного вывода строк
```

```
95              for(int i=0; i<_periods;i++) //добавление процентов каждый период
96              {
97                      capMoney+=mathMoney(); //накапливать за период
98                      clock++; //отсчёт периодов
99
100                     switch(proc)//проведение капитализации в зависимости от выбранного варианта
101                     {
102                     case 0: if (clock==1)  {deposit +=capMoney; addCapital(capMoney, period); period++; capMoney
                            =0; clock=0;} break;
103                     case 1: if (clock==3)  {deposit +=capMoney; addCapital(capMoney, period); period++; capMoney
                            =0; clock=0;} break;
104                     case 2: if (clock==12)  {deposit +=capMoney; addCapital(capMoney, period); period++; capMoney
                            =0; clock=0;} break;
105                     }
106
107             }
108     }
109
110 }
```

4.2. Test.java - unit-тесты

Листинг 3. UNIT-тесты

```
1  /**
2   *
3   */
4  package main;
5
6  //import static org.junit.jupiter.api.Assertions.*;
7  //import org.junit.Test;
8  //import org.junit.After;
9  //import org.junit.Before;
10
11 import static org.junit.jupiter.api.Assertions.assertEquals;
12
13 //import org.junit.After;
14 //import org.junit.Before;
15 /**
16  * @author stalk
17  *
18  */
19 class Test {
20
21         @org.junit.jupiter.api.Test
22
23         //первый тест без капитализации, с первым калькулятором
24         //вводе известен
25         //ожидается вывод 100 229 руб в проценты
26         //1 100 229 всего
27         //ожидаемые числа взяты как мода (самое часто встречающееся значение) из онлайн-калькуляторов
28         void base_test1() {
29                 Deposit depo = new Deposit();
30         depo.deposit = 1000000;
31         depo.day = 17;
32         depo.month = 3;
33         depo.year = 2019;
34         depo.interest = 10;
35         depo.term = 365;
36         depo.proc = 0;
37         depo.capitaliz = false;
38                 depo.calculate();
39                 assertEquals((int)depo.finalproc, 100229);//преобразование до int из за погрешностей
40                 assertEquals((int)depo.deposit, 1100229);
41         }
42
43         @org.junit.jupiter.api.Test
44
45         //аналогично, только теперь с капитализацией
46         void base_test2() {
47                 Deposit depo = new Deposit();
48         depo.deposit = 1000000;
49         depo.day = 17;
50         depo.month = 3;
51         depo.year = 2019;
52         depo.interest = 10;
```

```
53          depo.term = 365;
54          depo.proc = 0;
55          depo.capitaliz = true;
56                 depo.calculate();
57                 assertEquals((int)depo.finalproc, 104963);
58                 assertEquals((int)depo.deposit, 1104963);
59          }
60
61          @org.junit.jupiter.api.Test
62
63          //аналогично, только теперь с капитализацией по периодам
64          void base_test3() {
65                 Deposit depo = new Deposit();
66          depo.deposit = 1000000;
67          depo.day = 17;
68          depo.month = 3;
69          depo.year = 2019;
70          depo.interest = 10;
71          depo.term = 365;
72          depo.proc = 1;
73          depo.capitaliz = true;
74                 depo.calculate();
75                 assertEquals((int)depo.finalproc, 104107);
76                 assertEquals((int)depo.deposit, 1104107);
77          }
78
79          @org.junit.jupiter.api.Test
80
81          //с другой суммой и процентной ставкой
82          void base2_test1() {
83                 Deposit depo = new Deposit();
84          depo.deposit = 2000000;
85          depo.day = 17;
86          depo.month = 3;
87          depo.year = 2019;
88          depo.interest = 8;
89          depo.term = 365;
90          depo.proc = 0;
91          depo.capitaliz = false;
92                 depo.calculate();
93                 assertEquals((int)depo.finalproc, 160366);
94                 assertEquals((int)depo.deposit, 2160366);
95          }
96
97          @org.junit.jupiter.api.Test
98
99          //с другой суммой и процентной ставкой и сроком
100         void base3_test1() {
101                Deposit depo = new Deposit();
102         depo.deposit = 500000;
103         depo.day = 17;
104         depo.month = 3;
105         depo.year = 2019;
106         depo.interest = 5;
107         depo.term = 60;
108         depo.proc = 0;
109         depo.capitaliz = false;
110                depo.calculate();
111                assertEquals((int)depo.finalproc, 4178);
112                assertEquals((int)depo.deposit, 504178);
113         }
114
115         @org.junit.jupiter.api.Test
116
117         //с другой суммой и процентной ставкой и сроком и капитализацией
118         void base4_test1() {
119                Deposit depo = new Deposit();
120         depo.deposit = 400000;
121         depo.day = 17;
122         depo.month = 3;
123         depo.year = 2019;
124         depo.interest = 7;
125         depo.term = 91;
126         depo.proc = 0;
127         depo.capitaliz = true;
128                depo.calculate();
129                assertEquals((int)depo.finalproc, 7099);
130                assertEquals((int)depo.deposit, 407099);
131         }
```

```
132
133         @org.junit.jupiter.api.Test
134
135         //с другой суммой и процентной ставкой и сроком
136                 void base5_test1() {
137                         Deposit depo = new Deposit();
138                 depo.deposit = 300000;
139                 depo.day = 20;
140                 depo.month = 3;
141                 depo.year = 2019;
142                 depo.interest = 8;
143                 depo.term = 365;
144                 depo.proc = 0;
145                 depo.capitaliz = false;
146                         depo.calculate();
147                         assertEquals((int)depo.finalproc, 24054);
148                         assertEquals((int)depo.deposit, 324054);
149                 }
150
151         @org.junit.jupiter.api.Test
152
153         //тест без капитализации, ожидается вывод проценты - 8 018, всего - 108 018
154         void base6_test1() {
155                 Deposit depo = new Deposit();
156         depo.deposit = 100000;
157         depo.day = 20;
158         depo.month = 3;
159         depo.year = 2019;
160         depo.interest = 8;
161         depo.term = 365;
162         depo.proc = 0;
163         depo.capitaliz = false;
164                 depo.calculate();
165                 assertEquals((int)depo.finalproc, 8018);
166                 assertEquals((int)depo.deposit, 108018);
167         }
168
169 }
```

4.3. Proc.java - энумератор

Листинг 4. ENUM для текстбокса

```
1  package main;
2  //энумератор для комбобокса
3  public enum Proc {
4          ежемесячно,
5          ежеквартально,
6          ежегодно
7  }
```

4.4. wind.java - GUI и ввод

Листинг 5. GUI и обработка ввода

```
1  package main;
2
3  import java.awt.EventQueue;
4
5  import javax.swing.JFrame;
6  import javax.swing.JPanel;
7  import javax.swing.border.EmptyBorder;
8  import javax.swing.GroupLayout;
9  import javax.swing.GroupLayout.Alignment;
10 import javax.swing.JLabel;
11 import javax.swing.JOptionPane;
12 import javax.swing.JTextField;
13 import javax.swing.LayoutStyle.ComponentPlacement;
14 import javax.swing.UIManager;
15 import javax.swing.JComboBox;
16 import javax.swing.DefaultComboBoxModel;
17
18 import java.time.LocalDate;
```

```
19 import java.time.Month;
20 import javax.swing.JCheckBox;
21 import javax.swing.JButton;
22 import javax.swing.JTextPane;
23 import java.awt.event.ActionListener;
24 import java.awt.event.ActionEvent;
25 import javax.swing.JTable;
26 import javax.swing.table.DefaultTableModel;
27 import javax.swing.JScrollPane;
28
29 public class wind extends JFrame {
30
31         private JPanel contentPane;
32         private JTextField t_sum;
33         private JTextField t_day;
34         private JTextField t_year;
35         private JTextField t_month;
36         private JLabel l_time_vklad;
37         private JTextField t_time_vklad;
38         private JLabel label;
39         private JLabel l_proc;
40         private JTextField t_proc;
41         private JLabel label_2;
42         private JLabel label_3;
43         private JLabel label_4;
44         private JButton b_go;
45         private JCheckBox ch_capital;
46         private JTable table;
47         private JTextPane t_prok_out;
48         private JTextPane t_ost_vklad;
49         private JComboBox c_type_proc;
50
51         /**
52          * Launch the application.
53          */
54         public static void main(String[] args) {
55                 try {
56                         UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
57                 } catch (Throwable e) {
58                         e.printStackTrace();
59                 }
60                 EventQueue.invokeLater(new Runnable() {
61                         public void run() {
62                                 try {
63                                         wind frame = new wind();
64                                         frame.setVisible(true);
65                                 } catch (Exception e) {
66                                         e.printStackTrace();
67                                 }
68                         }
69                 });
70         }
71
72         /**
73          * Create the frame.
74          */
75         public wind() {//главная функция
76                 initComp();//создание компонентов
77                 create_event();//обработка событий
78         }
79
80         //
81         //ОБРАБОТКА СОБЫТИЙ
82         //
83         private void create_event()
84         {
85
86                 //ОБРАБОТКА ФЛАЖКА
87                 ch_capital.addActionListener(new ActionListener() {
88                         public void actionPerformed(ActionEvent arg0) {
89                                 if (ch_capital.isSelected()) //без капитализации вывод недоступен (т.к ничего не меняет
                                        )
90                                 {c_type_proc.setEnabled(true);}else
91                                 {c_type_proc.setEnabled(false);}
92                         }
93                 });
94
95                 //ОБРАБОТКА КНОПКИ
96                 b_go.addActionListener(new ActionListener() {
```

```
97
98                      public void actionPerformed(ActionEvent arg0) {//нажатие на кнопку "Рассчитать"
99                              Deposit deposit = new Deposit(); //создаём новый класс депозита
100                             //ввод значений и проверка
101
102                             //
103                             //ОБРАБОТКА ВВОДА
104                             //
105
106                             try{//сумма вклада, переменная deposit
107                                     deposit.deposit = Integer.parseInt(t_sum.getText());
108                                   } catch (NumberFormatException e) {
109                                         JOptionPane.showMessageDialog(null, "Неверно указана сумма вклада");
110                                   }
111
112                             try{//год
113                                     int year = Integer.parseInt(t_year.getText());
114
115                                     if (year<0) {throw new NumberFormatException();}else
116                                     {deposit.year = year;}
117
118                                   } catch (NumberFormatException e) {
119                                         JOptionPane.showMessageDialog(null, "Введите год числом больше 0");
120                                   }
121
122                             try{//месяц
123                                     int month = Integer.parseInt(t_month.getText());
124
125                                     if (month<1 || month>12) {throw new NumberFormatException();}else
126                                     {deposit.month = month;}
127
128                                   } catch (NumberFormatException e) {
129                                         JOptionPane.showMessageDialog(null, "Введите месяц числом от 1 до 12");
130                                   }
131
132                             try{//день
133                                     int day = Integer.parseInt(t_day.getText());
134                                     deposit.setDate(day);//сеттер
135                                   } catch (NumberFormatException e) {
136                                         JOptionPane.showMessageDialog(null, "Введите корректный день");
137                                         deposit.day=1;//по-умолчанию
138                                   }
139
140
141                             try{//срок вклада
142                                     int term = Integer.parseInt(t_time_vklad.getText());
143
144                                     if (term<0) {throw new NumberFormatException();}else
145                                     {deposit.term = term;}
146
147                                   } catch (NumberFormatException e) {
148                                         JOptionPane.showMessageDialog(null, "Введите срок числом больше 0");
149                                   }
150                             try{//процентная ставка
151                                     int interest = Integer.parseInt(t_proc.getText());
152
153                                     if (interest<0) {throw new NumberFormatException();}else
154                                     {deposit.interest = interest;}
155
156                                   } catch (NumberFormatException e) {
157                                         JOptionPane.showMessageDialog(null, "Введите число без знака процента");
158                                   }
159
160                             deposit.capitaliz = ch_capital.isSelected(); //капитализация
161                             deposit.proc = c_type_proc.getSelectedIndex();//капитализация, 0 - ежемесячно,1 -
                                    ежеквартально, 2 - ежегодно
162
163                             //
164                             //ВЫЧИСЛЕНИЯ
165                             //
166                             deposit.calculate();//вычисления
167
168                             //
169                             //ВЫВОД ИНФОРМАЦИИ
170                             //
171                             //текстбоксы
172                             t_prok_out.setText(String.format("%.2f",deposit.finalproc));//форматный вывод
173                             t_ost_vklad.setText(String.format("%.2f", deposit.deposit));
174                             //таблица
```

```
                                        table.setModel(new DefaultTableModel(//инициализация таблицы
                                deposit.table,//двухмерным массивом из deposit
                                new String[] {
                                        "Дата", "Проценты", "Остаток" //и заголовком
                                }
                        ));
                        deposit = null;//очистка класса, на всякий случай
                        }

                });
        }


//
//ИНИЦИАЛИЗАЦИЯ
//(по большей части код из конструктора)

        private void initComp()
        {
                setTitle("\u0414\u0435\u043F\u043E\u0437\u0438\u0442\u043D\u044B\u0439␣\u043A\u0430\u043B\u044C\u043A
                      \u0443\u043B\u044F\u0442\u043E\u0440");
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setBounds(100, 100, 599, 300);
                contentPane = new JPanel();
                contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
                setContentPane(contentPane);

                JLabel l_sum = new JLabel("\u0421\u0443\u043C\u043C\u0430␣\u0432\u043A\u043B\u0430\u0434\u0430:");

                t_sum = new JTextField();
                t_sum.setToolTipText("\u0412\u0432\u0435\u0434\u0438\u0442\u0435␣\u0441\u0443\u043C\u043C\u0443");
                t_sum.setColumns(10);

                JLabel l_date = new JLabel("\u0414\u0430\u0442\u0430␣\u043E\u0442\u043A\u0440\u044B\u0442\u0438\u044F
                      :");

                //представление даты переменными
        LocalDate date = LocalDate.now();

        int year = date.getYear();
        Month month = date.getMonth();
        int day = date.getDayOfMonth();

        //ввод текущей даты в текстбоксы
                t_day = new JTextField();
                t_day.setText(""+day);
                t_day.setColumns(10);

                t_year = new JTextField();
                t_year.setText(""+year);
                t_year.setColumns(10);

                t_month = new JTextField();
                t_month.setText(""+(month.ordinal()+1));
                t_month.setColumns(10);

                //автогенерация

                //Hic sunt dracones
                l_time_vklad = new JLabel("\u0421\u0440\u043E\u043A␣\u0432\u043A\u043B\u0430\u0434\u0430:");

                t_time_vklad = new JTextField();
                t_time_vklad.setToolTipText("\u0412␣\u0434\u043D\u044F\u0445");
                t_time_vklad.setText("365");
                t_time_vklad.setColumns(10);

                label = new JLabel("\u0434\u043D\u0435\u0439.");

                l_proc = new JLabel("\u041F\u0440\u043E\u0446\u0435\u043D\u0442\u043D\u0430\u044F␣\u0441\u0442\u0430\
                      u0432\u043A\u0430:");

                t_proc = new JTextField();
                t_proc.setToolTipText("\u0412␣\u043F\u0440\u043E\u0446\u0435\u043D\u0442\u0430\u0445");
                t_proc.setText("10");
                t_proc.setColumns(10);

                label_2 = new JLabel("%");

```

```
250             ch_capital = new JCheckBox("\u041A\u0430\u043F\u0438\u0442\u0430\u043B\u0438\u0446\u0430\u0446\u0438\
                    u044F␣\u043F\u0440\u043E\u0446\u0435\u043D\u0442\u043E\u0432");
251
252             ch_capital.setToolTipText("\u041F\u0440\u043E\u0446\u0435\u043D\u0442\u044B␣\u0441␣\u0443\u0441\u0442\
                    u0430\u043D\u043E\u0432\u043B\u0435\u043D\u043D\u043E\u0439␣\u043F\u0435\u0440\u0435\u043E\
                    u0434\u0438\u0447\u043D\u043E\u0441\u0442\u044C\u044E␣\u0434\u043E\u0431\u0430\u0432\u043B\u044F
                    \u044E\u0442\u0441\u044F␣\u043A␣\u0432\u043A\u043B\u0430\u0434\u0443");
253
254             JLabel label_1 = new JLabel("Частота␣капитализации:");
255
256             c_type_proc = new JComboBox();
257             c_type_proc.setToolTipText("\u0412\u044B\u043F\u043B\u0430\u0442\u0430␣\u0438\u043C\u0435\u0435\u0442
                    ␣\u0437\u043D\u0430\u0447\u0435\u043D\u0438\u0435␣\u0442\u043E\u043B\u044C\u043A\u043E␣\u043F\
                    u0440\u0438␣\u043A\u0430\u043F\u0438\u0442\u0430\u043B\u0438\u0437\u0430\u0446\u0438\u0438␣\
                    u043F\u0440\u043E\u0446\u0435\u043D\u0442\u043E\u0432");
258             c_type_proc.setEnabled(false);
259             c_type_proc.setModel(new DefaultComboBoxModel(Proc.values()));
260
261             b_go = new JButton("\u0420\u0430\u0441\u0441\u0447\u0438\u0442\u0430\u0442\u044C");
262
263
264             label_3 = new JLabel("Итоговая␣сумма:");
265
266             label_4 = new JLabel("\u0412\u044B\u043F\u043B\u0430\u0447\u0435\u043D\u043E␣\u043F\u0440\u043E\u0446
                    \u0435\u043D\u0442\u043E\u0432:");
267
268             t_ost_vklad = new JTextPane();
269             t_ost_vklad.setEditable(false);
270
271             t_prok_out = new JTextPane();
272             t_prok_out.setEditable(false);
273
274             JScrollPane scrollPane = new JScrollPane();
275
276             GroupLayout gl_contentPane = new GroupLayout(contentPane);
277             gl_contentPane.setHorizontalGroup(
278                    gl_contentPane.createParallelGroup(Alignment.LEADING)
279                            .addGroup(gl_contentPane.createSequentialGroup()
280                                    .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING)
281                                            .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING, false)
282                                                    .addGroup(gl_contentPane.createSequentialGroup()
283                                                            .addContainerGap()
284                                                            .addComponent(l_time_vklad, GroupLayout.PREFERRED_SIZE,
                                                                    94, GroupLayout.PREFERRED_SIZE))
285                                                    .addGroup(gl_contentPane.createSequentialGroup()
286                                                            .addGap(8)
287                                                            .addGroup(gl_contentPane.createParallelGroup(Alignment.
                                                                    LEADING)
288                                                                    .addGroup(gl_contentPane.createSequentialGroup()
289                                                                            .addComponent(l_sum)
290                                                                            .addPreferredGap(ComponentPlacement.RELATED)
291                                                                            .addComponent(t_sum, GroupLayout.
                                                                                    DEFAULT_SIZE, 151, Short.MAX_VALUE))
292                                                                    .addGroup(gl_contentPane.createSequentialGroup()
293                                                                            .addComponent(l_date)
294                                                                            .addPreferredGap(ComponentPlacement.RELATED)
295                                                                            .addGroup(gl_contentPane.createParallelGroup
                                                                                    (Alignment.LEADING)
296                                                                                    .addGroup(gl_contentPane.
                                                                                            createSequentialGroup()
297                                                                                            .addComponent(t_day,
                                                                                                    GroupLayout.
                                                                                                    PREFERRED_SIZE, 30,
                                                                                                    GroupLayout.
                                                                                                    PREFERRED_SIZE)
298                                                                                            .addPreferredGap(
                                                                                                    ComponentPlacement.
                                                                                                    RELATED, GroupLayout.
                                                                                                    DEFAULT_SIZE, Short.
                                                                                                    MAX_VALUE)
299                                                                                            .addComponent(t_month,
                                                                                                    GroupLayout.
                                                                                                    PREFERRED_SIZE, 34,
                                                                                                    GroupLayout.
                                                                                                    PREFERRED_SIZE))
300                                                                                    .addComponent(t_time_vklad, 0, 0,
                                                                                            Short.MAX_VALUE))
301                                                                    .addPreferredGap(ComponentPlacement.RELATED)
```

```
                                                                .addGroup(gl_contentPane.createParallelGroup
                                                                      (Alignment.LEADING)
                                                                        .addComponent(label)
                                                                        .addComponent(t_year, GroupLayout.
                                                                              DEFAULT_SIZE, 74, Short.
                                                                              MAX_VALUE)))))
                                                .addGroup(gl_contentPane.createSequentialGroup()
                                                        .addContainerGap()
                                                        .addComponent(l_proc)
                                                        .addPreferredGap(ComponentPlacement.RELATED)
                                                        .addComponent(t_proc, GroupLayout.PREFERRED_SIZE, 29,
                                                              GroupLayout.PREFERRED_SIZE)
                                                        .addPreferredGap(ComponentPlacement.RELATED)
                                                        .addComponent(label_2))
                                                .addGroup(gl_contentPane.createSequentialGroup()
                                                        .addContainerGap()
                                                        .addComponent(ch_capital))
                                                .addGroup(gl_contentPane.createSequentialGroup()
                                                        .addContainerGap()
                                                        .addComponent(label_1)
                                                        .addPreferredGap(ComponentPlacement.RELATED)
                                                        .addComponent(c_type_proc, GroupLayout.PREFERRED_SIZE,
                                                              GroupLayout.DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
                                                              ))
                                        .addGroup(gl_contentPane.createSequentialGroup()
                                                .addContainerGap()
                                                .addComponent(b_go, GroupLayout.DEFAULT_SIZE, 266, Short.
                                                      MAX_VALUE)
                                                .addGap(18)))
                                .addGap(18)
                                .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING)
                                        .addComponent(scrollPane, GroupLayout.DEFAULT_SIZE, 277, Short.MAX_VALUE)
                                        .addGroup(gl_contentPane.createSequentialGroup()
                                                .addComponent(label_3)
                                                .addPreferredGap(ComponentPlacement.RELATED)
                                                .addComponent(t_ost_vklad, GroupLayout.DEFAULT_SIZE, 180, Short.
                                                      MAX_VALUE))
                                        .addGroup(gl_contentPane.createSequentialGroup()
                                                .addComponent(label_4)
                                                .addPreferredGap(ComponentPlacement.RELATED)
                                                .addComponent(t_prok_out, GroupLayout.DEFAULT_SIZE, 140, Short.
                                                      MAX_VALUE)))
                                .addContainerGap())
                );
                gl_contentPane.setVerticalGroup(
                        gl_contentPane.createParallelGroup(Alignment.LEADING)
                                .addGroup(gl_contentPane.createSequentialGroup()
                                        .addContainerGap()
                                        .addGroup(gl_contentPane.createParallelGroup(Alignment.TRAILING)
                                                .addGroup(gl_contentPane.createParallelGroup(Alignment.BASELINE)
                                                        .addComponent(l_sum)
                                                        .addComponent(t_sum, GroupLayout.PREFERRED_SIZE, GroupLayout.
                                                              DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
                                                        .addComponent(label_3))
                                                .addComponent(t_ost_vklad, GroupLayout.PREFERRED_SIZE, 27, GroupLayout.
                                                      PREFERRED_SIZE))
                                        .addPreferredGap(ComponentPlacement.RELATED)
                                        .addGroup(gl_contentPane.createParallelGroup(Alignment.TRAILING)
                                                .addGroup(gl_contentPane.createParallelGroup(Alignment.BASELINE)
                                                        .addComponent(l_date)
                                                        .addComponent(t_day, GroupLayout.PREFERRED_SIZE, GroupLayout.
                                                              DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
                                                        .addComponent(t_year, GroupLayout.PREFERRED_SIZE, GroupLayout.
                                                              DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
                                                        .addComponent(t_month, GroupLayout.PREFERRED_SIZE, GroupLayout.
                                                              DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
                                                        .addComponent(label_4))
                                                .addComponent(t_prok_out, GroupLayout.PREFERRED_SIZE, GroupLayout.
                                                      DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE))
                                        .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING)
                                                .addGroup(gl_contentPane.createSequentialGroup()
                                                        .addGap(12)
                                                        .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING)
                                                                .addGroup(gl_contentPane.createSequentialGroup()
                                                                        .addGroup(gl_contentPane.createParallelGroup(
                                                                              Alignment.BASELINE)
                                                                                .addComponent(l_time_vklad)
                                                                                .addComponent(t_time_vklad, GroupLayout.
                                                                                      PREFERRED_SIZE, GroupLayout.
```

```
                                                                        DEFAULT_SIZE, GroupLayout.
                                                                        PREFERRED_SIZE))
364                                                             .addPreferredGap(ComponentPlacement.RELATED)
365                                                             .addGroup(gl_contentPane.createParallelGroup(
                                                                      Alignment.BASELINE)
366                                                                 .addComponent(l_proc)
367                                                                 .addComponent(t_proc, GroupLayout.
                                                                        PREFERRED_SIZE, GroupLayout.
                                                                        DEFAULT_SIZE, GroupLayout.
                                                                        PREFERRED_SIZE)
368                                                                 .addComponent(label_2)))
369                                                     .addComponent(label))
370                                             .addPreferredGap(ComponentPlacement.RELATED)
371                                             .addComponent(ch_capital)
372                                             .addPreferredGap(ComponentPlacement.RELATED)
373                                             .addGroup(gl_contentPane.createParallelGroup(Alignment.BASELINE)
374                                                     .addComponent(label_1)
375                                                     .addComponent(c_type_proc, GroupLayout.PREFERRED_SIZE,
                                                           GroupLayout.DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
                                                           )
376                                             .addPreferredGap(ComponentPlacement.RELATED)
377                                             .addComponent(b_go))
378                                     .addGroup(gl_contentPane.createSequentialGroup()
379                                             .addPreferredGap(ComponentPlacement.RELATED)
380                                             .addComponent(scrollPane, GroupLayout.DEFAULT_SIZE, 171, Short.
                                                   MAX_VALUE)
381                                             .addContainerGap()))
382             );
383
384             table = new JTable();
385             scrollPane.setViewportView(table);
386             table.setShowVerticalLines(true);
387             table.setRowSelectionAllowed(false);
388             contentPane.setLayout(gl_contentPane);
389     }
390 }
```