

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

Проверил:  
Старцев Г.В.

Уфа, 2019 г.

# Содержание

<b>1. Описание прототипа программного продукта</b>	<b>3</b>
<b>2. Программный код</b>	<b>4</b>
<b>3. Описание тестирования работы программы.</b>	<b>5</b>
<b>4. Приложение</b>	<b>6</b>
4.1. Deposit.java - основной класс . . . . .	6
4.2. Test.java - unit-тесты . . . . .	9
4.3. Proc.java - эnumератор . . . . .	11
4.4. wind.java - GUI и ввод . . . . .	12

# 1. Описание прототипа программного продукта

Для создания депозитного калькулятора была выбрана среда выполнения Eclipse; 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

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

```

1 //ОБРАБОТКА ФЛАЖКА
2 ch_capital.addActionListener(new ActionListener() {
3     public void actionPerformed(ActionEvent arg0) {
4         if (ch_capital.isSelected()) //без капитализации вывод недоступен (т.к. ничего не меняет)
5             {c_type_proc.setEnabled(true);}else
6             {c_type_proc.setEnabled(false);}
7     }
8 });

```

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 эnumератор 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
- 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 текстовых, 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>

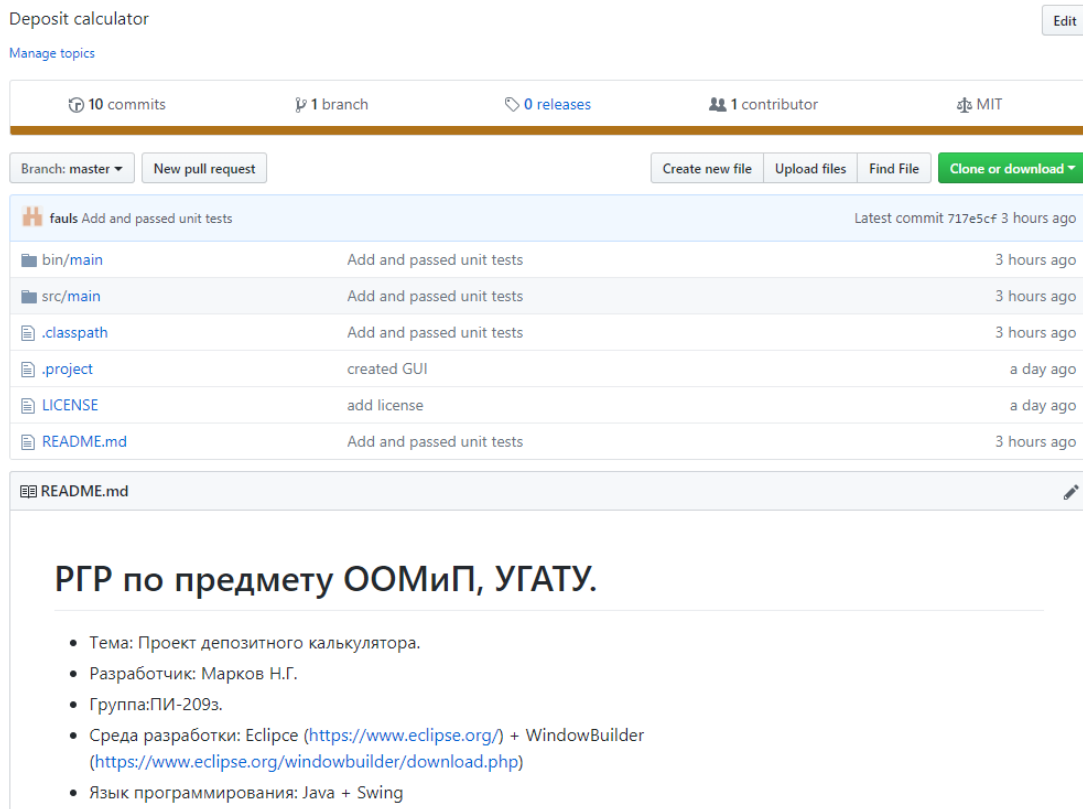


Рис. 2. Страница репозитория

### 3. Описание тестирования работы программы.

Для тестирования использовались JUnit5 тесты, содержащиеся в файле Test.java. В ходе тестирования 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 6 тестовых кейсов с различными условиями, полный код юнит-тестов можно найти в приложении.

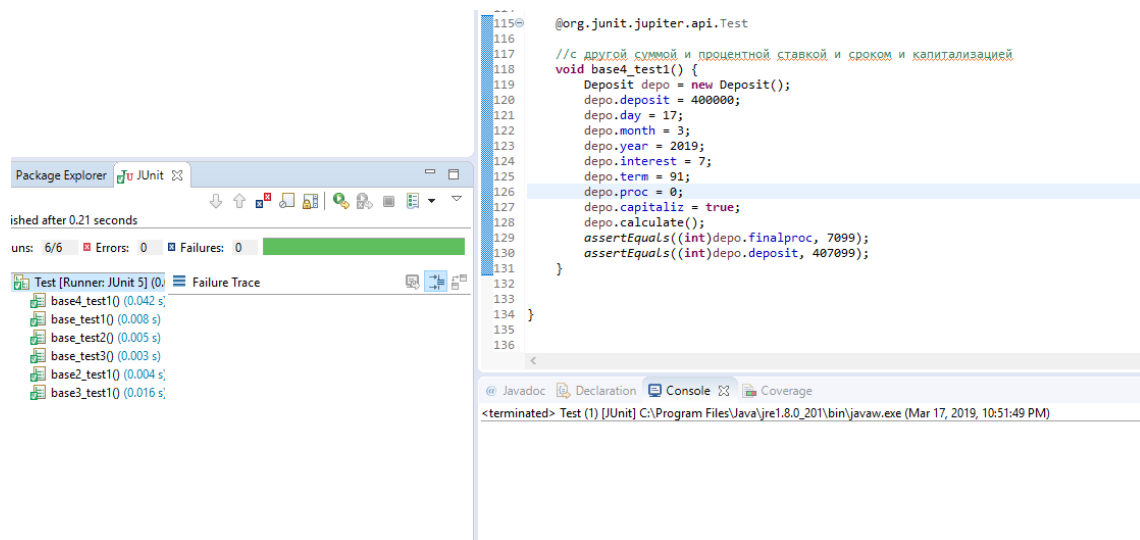


Рис. 3. Пример пройденного unit-тестирования

С помощью депозитного калькулятора вы сможете рассчитать ваш доход от размещения в банке вклада (депозита) на различные сроки и условия выплаты процентов. Рекомендуем задавать в калькуляторе условия, приближенные к рыночным на текущий момент.

Обращаясь в банк, проверьте, есть ли у него лицензия Центрального банка Российской Федерации, позволяющая привлекать вклады физических лиц, а также входит ли он в систему страхования вкладов. Помните, что застрахованная государством сумма вклада (а точнее – всех вкладов и счетов в одном банке) с учетом начисленных процентов не превышает 1,4 млн рублей.

Сумма вклада:  руб. -

Процентная ставка, % годовых:  ☐ Переносить даты платежей на понедельник

Дата открытия:

☒ Начисление процентов с учетом капитализации ☐ Досрочное закрытие вклада

Срок вклада:  год -

Периодичность капитализации:

Дополнительные пополнения вклада

Частичное изъятие вклада

Параметры вклада

Сумма вклада: 1 000 000,00 руб.

Срок вклада: 1 год

Процентная ставка: 10,00% годовых

Дата открытия вклада: 17.03.2019

Периодичность капитализации: Ежемесячно

Порядок начисления процентов: С капитализацией

**Депозитный калькулятор**

Сумма вклада:  Итоговая сумма:

Дата открытия:    Выплачено процентов:

Срок вклада:  дней.

Процентная ставка:  %

☒ Капитализация процентов

Частота капитализации:

Дата	Проценты	Остаток
17/04/19	8493.2	1008493.2
17/05/19	8289.0	1016782.1
17/06/19	8635.7	1025417.8
17/07/19	8428.1	1033845.9
17/08/19	8780.6	1042626.5
17/09/19	8855.2	1051481.7
17/10/19	8642.3	1060124.0
17/11/19	9003.8	1069127.8
17/12/19	8780.6	1077908.4
17/01/20	8428.1	1086336.5
17/02/20	8289.0	1094625.5
17/03/20	8493.2	1102818.7
17/04/20	8493.2	1111311.9
17/05/20	8289.0	1119599.9
17/06/20	8635.7	1128235.6
17/07/20	8428.1	1136663.7
17/08/20	8780.6	1145444.3
17/09/20	8855.2	1154299.5
17/10/20	8642.3	1163141.8
17/11/20	9003.8	1172145.6
17/12/20	8780.6	1181126.2
17/01/21	8428.1	1190154.3
17/02/21	8289.0	1199243.3
17/03/21	8493.2	1208336.5
17/04/21	8493.2	1217429.7
17/05/21	8289.0	1226518.7
17/06/21	8635.7	1235604.4
17/07/21	8428.1	1244682.5
17/08/21	8780.6	1253763.1
17/09/21	8855.2	1262848.3
17/10/21	8642.3	1271930.6
17/11/21	9003.8	1281034.4
17/12/21	8780.6	1290115.0
17/01/22	8428.1	1299183.1
17/02/22	8289.0	1308252.1
17/03/22	8493.2	1317325.3
17/04/22	8493.2	1326398.5
17/05/22	8289.0	1335477.5
17/06/22	8635.7	1344563.2
17/07/22	8428.1	1353641.3
17/08/22	8780.6	1362721.9
17/09/22	8855.2	1371807.1
17/10/22	8642.3	1380889.4
17/11/22	9003.8	1389993.2
17/12/22	8780.6	1399113.8
17/01/23	8428.1	1408241.9
17/02/23	8289.0	1417370.9
17/03/23	8493.2	1426494.1
17/04/23	8493.2	1435617.3
17/05/23	8289.0	1444736.3
17/06/23	8635.7	1453852.0
17/07/23	8428.1	1462960.1
17/08/23	8780.6	1472070.7
17/09/23	8855.2	1481185.9
17/10/23	8642.3	1490298.2
17/11/23	9003.8	1499412.0
17/12/23	8780.6	1508527.6
17/01/24	8428.1	1517645.7
17/02/24	8289.0	1526764.7
17/03/24	8493.2	1535887.9
17/04/24	8493.2	1545011.1
17/05/24	8289.0	1554130.1
17/06/24	8635.7	1563245.8
17/07/24	8428.1	1572363.9
17/08/24	8780.6	1581484.5
17/09/24	8855.2	1590609.7
17/10/24	8642.3	1599727.0
17/11/24	9003.8	1608848.8
17/12/24	8780.6	1617974.4
17/01/25	8428.1	1627102.5
17/02/25	8289.0	1636231.5
17/03/25	8493.2	1645364.7
17/04/25	8493.2	1654497.9
17/05/25	8289.0	1663630.9
17/06/25	8635.7	1672766.6
17/07/25	8428.1	1681904.7
17/08/25	8780.6	1691045.3
17/09/25	8855.2	1700189.5
17/10/25	8642.3	1709336.8
17/11/25	9003.8	1718488.6
17/12/25	8780.6	1727644.2
17/01/26	8428.1	1736802.3
17/02/26	8289.0	1745961.3
17/03/26	8493.2	1755124.5
17/04/26	8493.2	1764291.7
17/05/26	8289.0	1773460.7
17/06/26	8635.7	1782632.4
17/07/26	8428.1	1791806.5
17/08/26	8780.6	1800983.1
17/09/26	8855.2	1810162.3
17/10/26	8642.3	1819344.6
17/11/26	9003.8	1828530.4
17/12/26	8780.6	1837719.0
17/01/27	8428.1	1846910.1
17/02/27	8289.0	1856103.1
17/03/27	8493.2	1865298.3
17/04/27	8493.2	1874495.5
17/05/27	8289.0	1883694.5
17/06/27	8635.7	1892896.2
17/07/27	8428.1	1902099.3
17/08/27	8780.6	1911304.9
17/09/27	8855.2	1920513.1
17/10/27	8642.3	1929724.4
17/11/27	9003.8	1938938.2
17/12/27	8780.6	1948154.8
17/01/28	8428.1	1957373.9
17/02/28	8289.0	1966595.9
17/03/28	8493.2	1975819.1
17/04/28	8493.2	1985044.3
17/05/28	8289.0	1994271.3
17/06/28	8635.7	2003500.0
17/07/28	8428.1	2012730.1
17/08/28	8780.6	2021962.7
17/09/28	8855.2	2031197.9
17/10/28	8642.3	2040435.2
17/11/28	9003.8	2049674.0
17/12/28	8780.6	2058915.6
17/01/29	8428.1	2068159.7
17/02/29	8289.0	2077406.7
17/03/29	8493.2	2086655.9
17/04/29	8493.2	2095907.1
17/05/29	8289.0	2105160.1
17/06/29	8635.7	2114415.8
17/07/29	8428.1	2123673.9
17/08/29	8780.6	2132934.5
17/09/29	8855.2	2142197.7
17/10/29	8642.3	2151462.0
17/11/29	9003.8	2160728.8
17/12/29	8780.6	2169997.4
17/01/30	8428.1	2179268.5
17/02/30	8289.0	2188541.5
17/03/30	8493.2	2197816.7
17/04/30	8493.2	2207093.9
17/05/30	8289.0	2216372.9
17/06/30	8635.7	2225653.6
17/07/30	8428.1	2234936.7
17/08/30	8780.6	2244222.3
17/09/30	8855.2	2253510.5
17/10/30	8642.3	2262800.8
17/11/30	9003.8	2272093.6
17/12/30	8780.6	2281388.2
17/01/31	8428.1	2290684.3
17/02/31	8289.0	2300000.0
17/03/31	8493.2	2309316.2
17/04/31	8493.2	2318632.4
17/05/31	8289.0	2327949.4
17/06/31	8635.7	2337267.1
17/07/31	8428.1	2346585.2
17/08/31	8780.6	2355903.8
17/09/31	8855.2	2365223.0
17/10/31	8642.3	2374543.3
17/11/31	9003.8	2383864.1
17/12/31	8780.6	2393185.7
17/01/32	8428.1	2402507.8
17/02/32	8289.0	2411830.8
17/03/32	8493.2	2421154.0
17/04/32	8493.2	2430477.2
17/05/32	8289.0	2439800.2
17/06/32	8635.7	2449123.9
17/07/32	8428.1	2458448.0
17/08/32	8780.6	2467773.6
17/09/32	8855.2	2477099.8
17/10/32	8642.3	2486426.1
17/11/32	9003.8	2495752.9
17/12/32	8780.6	2505080.5
17/01/33	8428.1	2514408.6
17/02/33	8289.0	2523736.6
17/03/33	8493.2	2533064.8
17/04/33	8493.2	2542392.0
17/05/33	8289.0	2551719.0
17/06/33	8635.7	2561045.7
17/07/33	8428.1	2570372.8
17/08/33	8780.6	2579700.4
17/09/33	8855.2	2589028.6
17/10/33	8642.3	2598356.9
17/11/33	9003.8	2607685.7
17/12/33	8780.6	2617014.3
17/01/34	8428.1	2626342.4
17/02/34	8289.0	2635670.4
17/03/34	8493.2	2645000.6
17/04/34	8493.2	2654330.8
17/05/34	8289.0	2663660.8
17/06/34	8635.7	2672990.5
17/07/34	8428.1	2682320.6
17/08/34	8780.6	2691651.2
17/09/34	8855.2	2700982.4
17/10/34	8642.3	2710313.7
17/11/34	9003.8	2719645.5
17/12/34	8780.6	2728977.1
17/01/35	8428.1	2738308.2
17/02/35	8289.0	2747639.2
17/03/35	8493.2	2756970.4
17/04/35	8493.2	2766301.6
17/05/35	8289.0	2775632.6
17/06/35	8635.7	2784963.3
17/07/35	8428.1	2794294.4
17/08/35	8780.6	2803625.0
17/09/35	8855.2	2812956.2
17/10/35	8642.3	2822287.5
17/11/35	9003.8	2831618.3
17/12/35	8780.6	2840949.9
17/01/36	8428.1	2850281.0
17/02/36	8289.0	2859612.0
17/03/36	8493.2	2868943.2
17/04/36	8493.2	2878274.4
17/05/36	8289.0	2887605.4
17/06/36	8635.7	2896936.1
17/07/36	8428.1	2906267.2
17/08/36	8780.6	2915597.8
17/09/36	8855.2	2924929.0
17/10/36	8642.3	2934260.3
17/11/36	9003.8	2943591.5
17/12/36	8780.6	2952922.1
17/01/37	8428.1	2962253.2
17/02/37	8289.0	2971584.2
17/03/37	8493.2	2980915.4
17/04/37	8493.2	2990246.6
17/05/37	8289.0	2999577.6
17/06/37	8635.7	3008908.3
17/07/37	8428.1	3018239.4
17/08/37	8780.6	3027570.0
17/09/37	8855.2	3036901.2
17/10/37	8642.3	3046232.5
17/11/37	9003.8	3055563.7
17/12/37	8780.6	3064894.3
17/01/38	8428.1	3074225.4
17/02/38	8289.0	3083556.4
17/03/38	8493.2	3092887.6
17/04/38	8493.2	3102218.8
17/05/38	8289.0	3111549.8
17/06/38	8635.7	3120880.5
17/07/38	8428.1	3130211.6
17/08/38	8780.6	3139542.2
17/09/38	8855.2	3148873.4
17/10/38	8642.3	3158204.7
17/11/38	9003.8	3167535.9
17/12/38	8780.6	3176867.1
17/01/39	8428.1	3186198.2
17/02/39	8289.0	3195529.2
17/03/39	8493.2	3204860.4
17/04/39	8493.2	3214191.6
17/05/39	8289.0	3223522.6
17/06/39	8635.7	3232853.3
17/07/39	8428.1	3242184.4
17/08/39	8780.6	3251515.0
17/09/39	8855.2	3260846.2
17/10/39	8642.3	3270177.5
17/11/39	9003.8	3279508.7
17/12/39	8780.6	3288839.3
17/01/40	8428.1	3298170.4
17/02/40	8289.0	3307501.4
17/03/40	8493.2	3316832.6
17/04/40	8493.2	3326163.8
17/05/40	8289.0	3335494.8
17/06/40	8635.7	3344825.5
17/07/40	8428.1	3354156.6
17/08/40	8780.6	3363487.2
17/09/40	8855.2	3372818.4
17/10/40	8642.3	3382149.7
17/11/40	9003.8	3391480.9
17/12/40	8780.6	3400811.5
17/01/41	8428.1	3410142.6
17/02/41	8289.0	3419473.6
17/03/41	8493.2	3428804.8
17/04/41	8493.2	3438136.0
17/05/41	8289.0	3447467.0
17/06/41	8635.7	3456797.7
17/07/41	8428.1	3466128.8
17/		

```

20
21 private LocalDate _startdate; //начальная дата
22 private LocalDate _stopdate; //конечная
23 private int _periods; //кол-во месяцев (проценты добавляются ежемесячно)
24 double finalproc=0; //итоговые проценты
25
26 public void calculate()
27 {
28     dayInPeriod();//подсчёт срока
29
30     if (capitaliz)
31     {hardInterest();}
32     else
33     {simpleInterest();}//если не капитализация, то это простые проценты
34 }
35
36 private void dayInPeriod()//считаем время от вложения вклада до забора
37 {
38     _startdate = LocalDate.of(year, month, day); //конвертируем дату из формата ввода в удобный
39     _stopdate = _startdate;
40     _stopdate = _stopdate.plusDays(term+1);//начальная дата + срок = финальная дата +1 т.к финальная не
41     учитывается
42     _periods = (int) ChronoUnit.MONTHS.between(_startdate, _stopdate);//кол-во полных месяцев в периоде
43 }
44
45 private double howDay()//сколько дней в месяце
46 {
47     return _startdate.lengthOfMonth();
48 }
49
50 private double mathMoney()//подсчёт по формуле
51 {
52     return deposit*((interest/100)*(howDay())/_startdate.lengthOfYear()); //подневный расчёт
53 }
54
55 private void addCapital(double capMoney, int i)//добавить сумму к финальной, внести строку в таблицу,
56 сдвинуть дату
57 {
58     finalproc += capMoney; //финальные проценты - сумма всех процентов
59     switch(proc)//добавление месяцев, в зависимости от частоты вывода в таблицу
60     {
61         case 0: _startdate = _startdate.plusMonths(1); break;//добавление месяца
62         case 1: _startdate = _startdate.plusMonths(3); break;
63         case 2: _startdate = _startdate.plusYears(1); break;
64     }
65     table[i][0]=_startdate.format(DateTimeFormatter.ofPattern("d/MM/YY")); //форматированный вывод даты в
66     первую колонку
67     table[i][1]=String.format("%.1f",capMoney); //формат до 2х знаков после запятой
68     table[i][2]=String.format("%.1f",deposit);
69 }
70
71 private void simpleInterest()//простые проценты, для них выдача/невыдача денег не важна, так что проц
72 игнорируется
73 {
74     table = new String[_periods][3]; //инициализация таблицы
75
76     for(int i=0; i<_periods;i++) //добавление процентов каждый период
77     {addCapital(mathMoney(), i); //считаем
78     }
79     deposit+=finalproc; //к итогу добавляем проценты
80 }
81
82 private void hardInterest()//сложные проценты
83 {
84     table = new String[_periods][3]; //инициализация таблицы
85     double capMoney = 0; //накопление процентов за период
86     int clock = 0; //для отсчёта периодов
87     int period =0; //для ровного вывода строк
88     for(int i=0; i<_periods;i++) //добавление процентов каждый период
89     {
90         capMoney+=mathMoney(); //накапливать за период
91         clock++; //отсчёт периодов
92
93         switch(proc)//проведение капитализации в зависимости от выбранного варианта
94         {
95             case 0: if (clock==1) {deposit +=capMoney; addCapital(capMoney, period); period++; capMoney
96                 =0; clock=0;} break;

```

```

93         case 1: if (clock==3) {deposit +=capMoney; addCapital(capMoney, period); period++; capMoney
94             =0; clock=0;} break;
95         case 2: if (clock==12) {deposit +=capMoney; addCapital(capMoney, period); period++; capMoney
96             =0; clock=0;} break;
97     }
98 }
99
100 }

```

## 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
134 }

```

### 4.3. Proc.java - эnumератор

## Листинг 4. ENUM для текстовкса

```
1 package main;
2 //эnumератор 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                 )
91                 {c_type_proc.setEnabled(true);}else
92                 {c_type_proc.setEnabled(false);}
93         }
94     });
95
96     //ОБРАБОТКА КНОПКИ
97     b_go.addActionListener(new ActionListener() {
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{//день
113                 int day = Integer.parseInt(t_day.getText());
114
115                 if (day<1 || day>31) {throw new NumberFormatException();}else//генерируем
116                     исключение если выходим за пределы
117                     {deposit.day = day;}
118             } catch (NumberFormatException e) {
119                 JOptionPane.showMessageDialog(null, "Введите_день_числом_от_1_до_31");
120             }
121             try{//месяц
122                 int month = Integer.parseInt(t_month.getText());
123
124                 if (month<1 || month>12) {throw new NumberFormatException();}else
125                     {deposit.month = month;}
126             } catch (NumberFormatException e) {
127                 JOptionPane.showMessageDialog(null, "Введите_месяц_числом_от_1_до_12");
128             }
129             try{//год
130                 int year = Integer.parseInt(t_year.getText());
131
132                 if (year<0) {throw new NumberFormatException();}else
133                     {deposit.year = year;}
134             } catch (NumberFormatException e) {
135                 JOptionPane.showMessageDialog(null, "Введите_год_числом_больше_0");
136             }
137
138             try{//срок вклада
139                 int term = Integer.parseInt(t_time_vklad.getText());

```



[illegible]

```

285         .addGroup(gl_contentPane.createParallelGroup(Alignment.
286             LEADING)
287             .addGroup(gl_contentPane.createSequentialGroup()
288                 .addComponent(l_sum)
289                 .addPreferredGap(ComponentPlacement.RELATED)
290                 .addComponent(t_sum, GroupLayout.
291                     DEFAULT_SIZE, 151, Short.MAX_VALUE))
292             .addGroup(gl_contentPane.createSequentialGroup()
293                 .addComponent(l_date)
294                 .addPreferredGap(ComponentPlacement.RELATED)
295                 .addGroup(gl_contentPane.createParallelGroup
296                     (Alignment.LEADING)
297                     .addGroup(gl_contentPane.
298                         createSequentialGroup()
299                         .addComponent(t_day,
300                             GroupLayout.
301                                 PREFERRED_SIZE, 30,
302                                 GroupLayout.
303                                     PREFERRED_SIZE)
304                         .addPreferredGap(
305                             ComponentPlacement.
306                                 RELATED, GroupLayout.
307                                     DEFAULT_SIZE, Short.
308                                         MAX_VALUE)
309                         .addComponent(t_month,
310                             GroupLayout.
311                                 PREFERRED_SIZE, 34,
312                                 GroupLayout.
313                                     PREFERRED_SIZE))
314                         .addComponent(t_time_vklad, 0, 0,
315                             Short.MAX_VALUE))
316                     .addPreferredGap(ComponentPlacement.RELATED)
317                     .addGroup(gl_contentPane.createParallelGroup
318                         (Alignment.LEADING)
319                         .addComponent(label)
320                         .addComponent(t_year, GroupLayout.
321                             DEFAULT_SIZE, 74, Short.
322                                 MAX_VALUE))))))
323         .addGroup(gl_contentPane.createSequentialGroup()
324             .addContainerGap()
325             .addComponent(l_proc)
326             .addPreferredGap(ComponentPlacement.RELATED)
327             .addComponent(t_proc, GroupLayout.PREFERRED_SIZE, 29,
328                 GroupLayout.PREFERRED_SIZE)
329             .addPreferredGap(ComponentPlacement.RELATED)
330             .addComponent(label_2))
331         .addGroup(gl_contentPane.createSequentialGroup()
332             .addContainerGap()
333             .addComponent(ch_capital))
334         .addGroup(gl_contentPane.createSequentialGroup()
335             .addContainerGap()
336             .addComponent(label_1)
337             .addPreferredGap(ComponentPlacement.RELATED)
338             .addComponent(c_type_proc, GroupLayout.PREFERRED_SIZE,
339                 GroupLayout.DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
340             ))
341         .addGroup(gl_contentPane.createSequentialGroup()
342             .addContainerGap()
343             .addComponent(b_go, GroupLayout.DEFAULT_SIZE, 266, Short.
344                 MAX_VALUE)
345             .addGap(18)))
346     .addGap(18)
347     .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING)
348         .addComponent(scrollPane, GroupLayout.DEFAULT_SIZE, 277, Short.MAX_VALUE)
349         .addGroup(gl_contentPane.createSequentialGroup()
350             .addComponent(label_3)
351             .addPreferredGap(ComponentPlacement.RELATED)
352             .addComponent(t_ost_vklad, GroupLayout.DEFAULT_SIZE, 180, Short.
353                 MAX_VALUE))
354         .addGroup(gl_contentPane.createSequentialGroup()
355             .addComponent(label_4)
356             .addPreferredGap(ComponentPlacement.RELATED)
357             .addComponent(t_prok_out, GroupLayout.DEFAULT_SIZE, 140, Short.
358                 MAX_VALUE)))
359     .addContainerGap())
360 );
361 gl_contentPane.setVerticalGroup(
362     gl_contentPane.createParallelGroup(Alignment.LEADING)
363         .addGroup(gl_contentPane.createSequentialGroup()

```

```

338         .addContainerGap()
339     .addGroup(gl_contentPane.createParallelGroup(Alignment.TRAILING)
340         .addGroup(gl_contentPane.createParallelGroup(Alignment.BASELINE)
341             .addComponent(l_sum)
342             .addComponent(t_sum, GroupLayout.PREFERRED_SIZE, GroupLayout.
343                 DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
344             .addComponent(label_3))
345         .addGroup(gl_contentPane.createParallelGroup(Alignment.TRAILING)
346             .addGroup(gl_contentPane.createParallelGroup(Alignment.BASELINE)
347                 .addComponent(l_date)
348                 .addComponent(t_day, GroupLayout.PREFERRED_SIZE, GroupLayout.
349                     DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
350                 .addComponent(t_year, GroupLayout.PREFERRED_SIZE, GroupLayout.
351                     DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
352                 .addComponent(t_month, GroupLayout.PREFERRED_SIZE, GroupLayout.
353                     DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
354                 .addComponent(label_4))
355             .addComponent(t_prok_out, GroupLayout.PREFERRED_SIZE, GroupLayout.
356                 DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE))
357         .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING)
358             .addGroup(gl_contentPane.createSequentialGroup()
359                 .addGroup(gl_contentPane.createParallelGroup(
360                     Alignment.BASELINE)
361                     .addComponent(l_time_vklad)
362                     .addComponent(t_time_vklad, GroupLayout.
363                         PREFERRED_SIZE, GroupLayout.
364                         DEFAULT_SIZE, GroupLayout.
365                         PREFERRED_SIZE))
366                 .addPreferredGap(ComponentPlacement.RELATED)
367                 .addGroup(gl_contentPane.createParallelGroup(
368                     Alignment.BASELINE)
369                     .addComponent(l_proc)
370                     .addComponent(t_proc, GroupLayout.
371                         PREFERRED_SIZE, GroupLayout.
372                         DEFAULT_SIZE, GroupLayout.
373                         PREFERRED_SIZE)
374                     .addComponent(label_2)))
375                 .addComponent(label))
376             .addPreferredGap(ComponentPlacement.RELATED)
377             .addComponent(ch_capital)
378             .addPreferredGap(ComponentPlacement.RELATED)
379             .addGroup(gl_contentPane.createParallelGroup(Alignment.BASELINE)
380                 .addComponent(label_1)
381                 .addComponent(c_type_proc, GroupLayout.PREFERRED_SIZE,
382                     GroupLayout.DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
383             )
384             .addPreferredGap(ComponentPlacement.RELATED)
385             .addComponent(b_go))
386         .addGroup(gl_contentPane.createSequentialGroup()
387             .addPreferredGap(ComponentPlacement.RELATED)
388             .addComponent(scrollPane, GroupLayout.DEFAULT_SIZE, 171, Short.
389                 MAX_VALUE)
390             .addContainerGap()))))
391     );
392
393     table = new JTable();
394     scrollPane.setViewportViewView(table);
395     table.setShowVerticalLines(true);
396     table.setRowSelectionAllowed(false);
397     contentPane.setLayout(gl_contentPane);
398 }

```