

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	3
2. Программный код	4
3. Описание тестирования работы программы.	5
4. Приложение	6
4.1. Deposit.java - основной класс	6
4.2. Test.java - unit-тесты	8
4.3. Proc.java - эnumератор	9
4.4. wind.java - GUI и ввод	10

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
- 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 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>

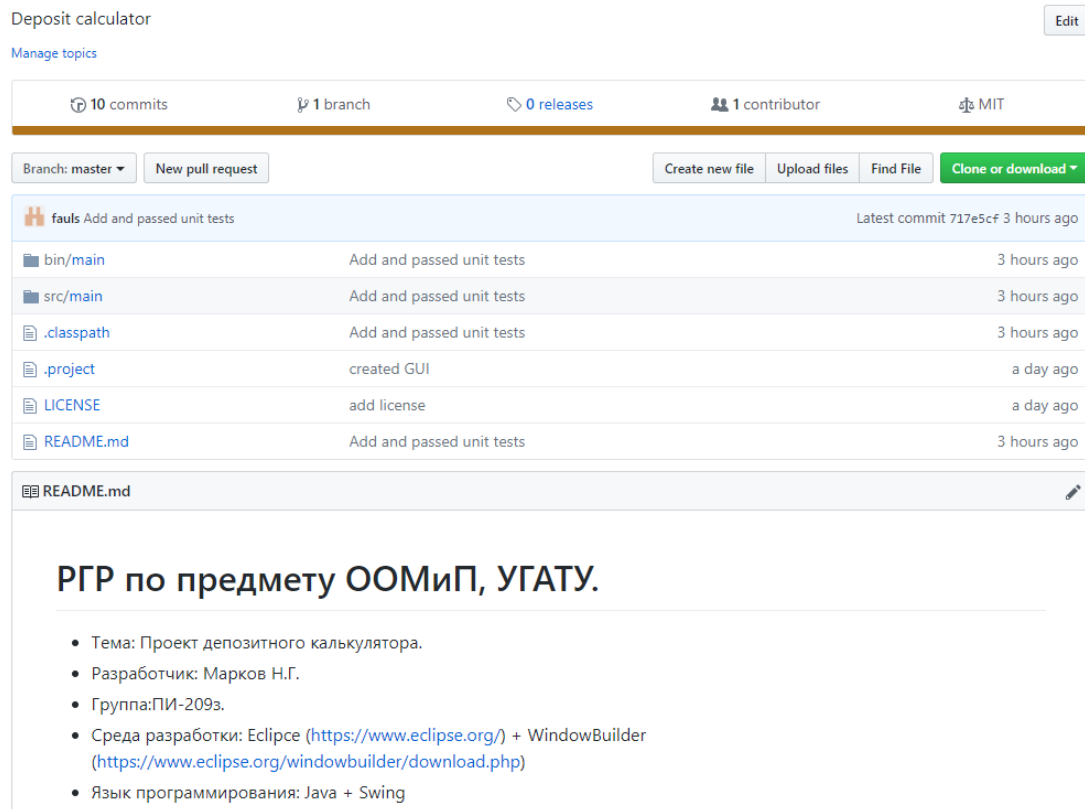


Рис. 2. Страница репозитория

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

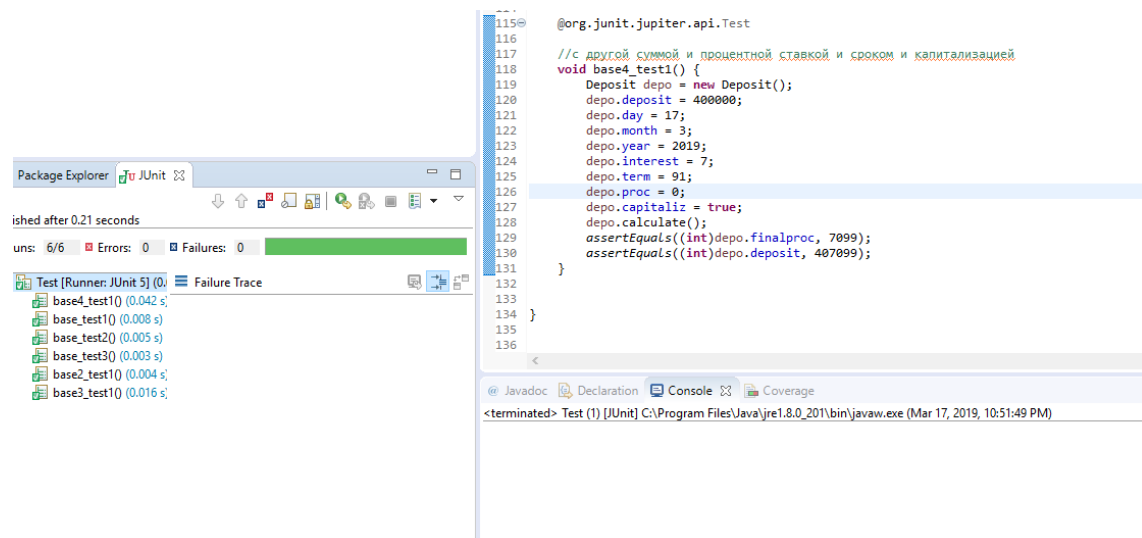


Рис. 3. Пример пройденного unit-тестирования

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: 1 000 000,00 руб. -

Процентная ставка, % годовых: 10,00

Дата открытия: 17.03.2019

Срок вклада: 1 год

Периодичность капитализации: Ежемесячно

Дополнительные пополнения вклада

Добавить пополнение

Частичное изъятие вклада

Добавить изъятие

Параметры вклада

Сумма вклада: 1 000 000

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: 1000000

Дата открытия: 17 3 2019

Срок вклада: 365 дней

Процентная ставка: 10 %

☒ Капитализация процентов

Частота капитализации: ежемесячно

Итоговая сумма: 1104963.84

Выплачено процентов: 104963.84

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	1119600.9
17/06/20	8635.7	1128236.6
17/07/20	8428.1	1136664.7
17/08/20	8780.6	1145445.3
17/09/20	8855.2	1154290.5
17/10/20	8642.3	1163032.8
17/11/20	9003.8	1171736.6
17/12/20	8780.6	1180417.2
17/01/21	8428.1	1188945.3
17/02/21	8289.0	1197234.3
17/03/21	8493.2	1205427.5
17/04/21	8493.2	1213620.7
17/05/21	8289.0	1221609.7
17/06/21	8635.7	1229445.4
17/07/21	8428.1	1237073.5
17/08/21	8780.6	1244554.1
17/09/21	8855.2	1251889.3
17/10/21	8642.3	1259031.6
17/11/21	9003.8	1265935.4
17/12/21	8780.6	1272616.0
17/01/22	8428.1	1279044.1
17/02/22	8289.0	1285233.1
17/03/22	8493.2	1291226.3
17/04/22	8493.2	1297019.5
17/05/22	8289.0	1302608.5
17/06/22	8635.7	1308044.2
17/07/22	8428.1	1313272.3
17/08/22	8780.6	1318292.9
17/09/22	8855.2	1323108.1
17/10/22	8642.3	1327750.4
17/11/22	9003.8	1332254.2
17/12/22	8780.6	1336634.8
17/01/23	8428.1	1340802.9
17/02/23	8289.0	1344761.9
17/03/23	8493.2	1348515.1
17/04/23	8493.2	1352068.3
17/05/23	8289.0	1355457.3
17/06/23	8635.7	1358693.0
17/07/23	8428.1	1361771.1
17/08/23	8780.6	1364691.7
17/09/23	8855.2	1367446.9
17/10/23	8642.3	1370039.2
17/11/23	9003.8	1372443.0
17/12/23	8780.6	1374623.6
17/01/24	8428.1	1376581.7
17/02/24	8289.0	1378320.7
17/03/24	8493.2	1379813.9
17/04/24	8493.2	1381167.1
17/05/24	8289.0	1382356.1
17/06/24	8635.7	1383391.8
17/07/24	8428.1	1384279.9
17/08/24	8780.6	1385010.5
17/09/24	8855.2	1385595.7
17/10/24	8642.3	1386038.0
17/11/24	9003.8	1386341.8
17/12/24	8780.6	1386522.4
17/01/25	8428.1	1386580.5
17/02/25	8289.0	1386511.5
17/03/25	8493.2	1386324.7
17/04/25	8493.2	1385931.9
17/05/25	8289.0	1385342.9
17/06/25	8635.7	1384557.6
17/07/25	8428.1	1383579.7
17/08/25	8780.6	1382400.3
17/09/25	8855.2	1381025.5
17/10/25	8642.3	1379453.2
17/11/25	9003.8	1377677.0
17/12/25	8780.6	1375696.4
17/01/26	8428.1	1373518.5
17/02/26	8289.0	1371129.5
17/03/26	8493.2	1368542.7
17/04/26	8493.2	1365749.9
17/05/26	8289.0	1362760.9
17/06/26	8635.7	1359576.6
17/07/26	8428.1	1356198.7
17/08/26	8780.6	1352618.3
17/09/26	8855.2	1348833.5
17/10/26	8642.3	1344841.2
17/11/26	9003.8	1340637.4
17/12/26	8780.6	1336223.0
17/01/27	8428.1	1331599.1
17/02/27	8289.0	1326760.1
17/03/27	8493.2	1321706.9
17/04/27	8493.2	1316439.1
17/05/27	8289.0	1310950.1
17/06/27	8635.7	1305241.8
17/07/27	8428.1	1299313.9
17/08/27	8780.6	1293164.5
17/09/27	8855.2	1286794.7
17/10/27	8642.3	1280202.4
17/11/27	9003.8	1273386.2
17/12/27	8780.6	1266345.6
17/01/28	8428.1	1259077.7
17/02/28	8289.0	1251588.7
17/03/28	8493.2	1243881.9
17/04/28	8493.2	1235955.1
17/05/28	8289.0	1227796.1
17/06/28	8635.7	1219410.8
17/07/28	8428.1	1210782.9
17/08/28	8780.6	1201902.5
17/09/28	8855.2	1192767.7
17/10/28	8642.3	1183375.4
17/11/28	9003.8	1173726.2
17/12/28	8780.6	1163830.8
17/01/29	8428.1	1153688.5
17/02/29	8289.0	1143299.5
17/03/29	8493.2	1132662.7
17/04/29	8493.2	1121769.9
17/05/29	8289.0	1110620.9
17/06/29	8635.7	1099216.6
17/07/29	8428.1	1087548.7
17/08/29	8780.6	1075608.3
17/09/29	8855.2	1063393.5
17/10/29	8642.3	1050901.2
17/11/29	9003.8	1038127.4
17/12/29	8780.6	1025076.0
17/01/30	8428.1	1011747.1
17/02/30	8289.0	998138.1
17/03/30	8493.2	984244.9
17/04/30	8493.2	969951.1
17/05/30	8289.0	955262.1
17/06/30	8635.7	939976.8
17/07/30	8428.1	924098.9
17/08/30	8780.6	907518.5
17/09/30	8855.2	890233.7
17/10/30	8642.3	872241.4
17/11/30	9003.8	853535.2
17/12/30	8780.6	834115.6
17/01/31	8428.1	813987.7
17/02/31	8289.0	793148.7
17/03/31	8493.2	771555.9
17/04/31	8493.2	749202.1
17/05/31	8289.0	726083.1
17/06/31	8635.7	702197.8
17/07/31	8428.1	677545.9
17/08/31	8780.6	652125.5
17/09/31	8855.2	625930.7
17/10/31	8642.3	598968.4
17/11/31	9003.8	571232.2
17/12/31	8780.6	542711.6
17/01/32	8428.1	513403.7
17/02/32	8289.0	483214.7
17/03/32	8493.2	452121.9
17/04/32	8493.2	420128.1
17/05/32	8289.0	387239.1
17/06/32	8635.7	353453.8
17/07/32	8428.1	318775.9
17/08/32	8780.6	283195.5
17/09/32	8855.2	246710.7
17/10/32	8642.3	209318.4
17/11/32	9003.8	170914.2
17/12/32	8780.6	131483.6
17/01/33	8428.1	90925.7
17/02/33	8289.0	50136.7
17/03/33	8493.2	9000.9
17/04/33	8493.2	-3093.1
17/05/33	8289.0	-6182.1
17/06/33	8635.7	-9267.8
17/07/33	8428.1	-12339.9
17/08/33	8780.6	-15387.5
17/09/33	8855.2	-18402.3
17/10/33	8642.3	-21384.6
17/11/33	9003.8	-24325.4
17/12/33	8780.6	-27225.0
17/01/34	8428.1	-30083.1
17/02/34	8289.0	-32899.1
17/03/34	8493.2	-35672.9
17/04/34	8493.2	-38404.1
17/05/34	8289.0	-41093.1
17/06/34	8635.7	-43739.8
17/07/34	8428.1	-46344.9
17/08/34	8780.6	-48908.5
17/09/34	8855.2	-51429.7
17/10/34	8642.3	-53907.4
17/11/34	9003.8	-56341.2
17/12/34	8780.6	-58731.6
17/01/35	8428.1	-61078.7
17/02/35	8289.0	-63382.7
17/03/35	8493.2	-65633.9
17/04/35	8493.2	-67832.1
17/05/35	8289.0	-70077.1
17/06/35	8635.7	-72268.8
17/07/35	8428.1	-74407.9
17/08/35	8780.6	-76494.5
17/09/35	8855.2	-78528.7
17/10/35	8642.3	-80510.4
17/11/35	9003.8	-82439.2
17/12/35	8780.6	-84315.0
17/01/36	8428.1	-86137.1
17/02/36	8289.0	-87896.1
17/03/36	8493.2	-89592.9
17/04/36	8493.2	-91227.1
17/05/36	8289.0	-92808.1
17/06/36	8635.7	-94326.8
17/07/36	8428.1	-95783.9
17/08/36	8780.6	-97179.5
17/09/36	8855.2	-98513.7
17/10/36	8642.3	-99786.4
17/11/36	9003.8	-101097.2
17/12/36	8780.6	-102346.0
17/01/37	8428.1	-103532.1
17/02/37	8289.0	-104655.1
17/03/37	8493.2	-105715.9
17/04/37	8493.2	-106714.1
17/05/37	8289.0	-107649.1
17/06/37	8635.7	-108520.8
17/07/37	8428.1	-109329.9
17/08/37	8780.6	-110076.5
17/09/37	8855.2	-110760.7
17/10/37	8642.3	-111382.4
17/11/37	9003.8	-111941.2
17/12/37	8780.6	-112437.0
17/01/38	8428.1	-112869.1
17/02/38	8289.0	-113237.1
17/03/38	8493.2	-113541.9
17/04/38	8493.2	-113783.1
17/05/38	8289.0	-113960.1
17/06/38	8635.7	-114072.8
17/07/38	8428.1	-114121.9
17/08/38	8780.6	-114107.5
17/09/38	8855.2	-114029.7
17/10/38	8642.3	-113887.4
17/11/38	9003.8	-113681.2
17/12/38	8780.6	-113411.0
17/01/39	8428.1	-113076.1
17/02/39	8289.0	-112677.1
17/03/39	8493.2	-112214.9
17/04/39	8493.2	-111689.1
17/05/39	8289.0	-111099.1
17/06/39	8635.7	-110444.8
17/07/39	8428.1	-109726.9
17/08/39	8780.6	-108945.5
17/09/39	8855.2	-108090.7
17/10/39	8642.3	-107162.4
17/11/39	9003.8	-106160.2
17/12/39	8780.6	-105084.0
17/01/40	8428.1	-103934.1
17/02/40	8289.0	-102710.1
17/03/40	8493.2	-101411.9
17/04/40	8493.2	-100039.1
17/05/40	8289.0	-98592.1
17/06/40	8635.7	-97070.8
17/07/40	8428.1	-95475.9
17/08/40	8780.6	-93807.5
17/09/40	8855.2	-92065.7
17/10/40	8642.3	-90250.4
17/11/40	9003.8	-88361.2
17/12/40	8780.6	-86397.0
17/01/41	8428.1	-84357.1
17/02/41	8289.0	-82242.1
17/03/41	8493.2	-80052.9
17/04/41	8493.2	-77789.1
17/05/41	8289.0	-75450.1
17/06/41	8635.7	-73036.8
17/07/41	8428.1	-70549.9
17/08/41	8780.6	-67989.5
17/09/41	8855.2	-65354.7
17/10/41	8642.3	-62645.4
17/11/41	9003.8	-59861.2
17/12/41	8780.6	-56992.0
17/01/42	8428.1	-54037.1
17/02/42	8289.0	-51096.1
17/03/42	8493.2	-48169.9
17/04/42	8493.2	-45157.1
17/05/42	8289.0	-4

```

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 ежемесячно)
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
51     учитывается
52     _periods = (int) ChronoUnit.MONTHS.between(_startdate, _stopdate);//кол-во полных месяцев в периоде
53 }
54
55 private double howDay()//сколько дней в месяце
56 {
57     return _startdate.lengthOfMonth();
58 }
59
60 private double mathMoney()//подсчёт по формуле
61 {
62     return deposit*((interest/100)*(howDay())/_startdate.lengthOfYear()); //подневный расчет
63 }
64
65 private void addCapital(double capMoney, int i)//добавить сумму к финальной, внести строку в таблицу,
66 сдвинуть дату
67 {
68     finalproc += capMoney; //финальные проценты - сумма всех процентов
69     switch(proc)//добавление месяцев, в зависимости от частоты вывода в таблицу
70     {
71         case 0: _startdate = _startdate.plusMonths(1); break;//добавление месяца
72         case 1: _startdate = _startdate.plusMonths(3); break;
73         case 2: _startdate = _startdate.plusYears(1); break;
74     }
75     table[i][0]=_startdate.format(DateTimeFormatter.ofPattern("d/MM/YY")); //форматированный вывод даты в
76     первую колонку
77     table[i][1]=String.format("%.1f",capMoney); //формат до 2х знаков после запятой
78     table[i][2]=String.format("%.1f",deposit);
79 }
80
81 private void simpleInterest()//простые проценты, для них выдача/невыдача денег не важна, так что proc
82 игнорируется
83 {
84
85     table = new String[_periods][3]; //инициализация таблицы
86
87     for(int i=0; i<_periods;i++) //добавление процентов каждый период
88     {addCapital(mathMoney(), i); //считаем
89     }
90     deposit+=finalproc; //к итогу добавляем проценты
91 }
92
93 private void hardInterest()//сложные проценты
94 {
95     table = new String[_periods][3]; //инициализация таблицы
96     double capMoney = 0; //накопление процентов за период
97     int clock = 0; //для отсчёта периодов
98     int period =0; //для равного 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 - эnumератор

Листинг 4. ENUM для текстовка

```

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         initComponents(//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() {

```

```

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
151     try{//процентная ставка
152         int interest = Integer.parseInt(t_proc.getText());
153
154         if (interest<0) {throw new NumberFormatException();}else
155         {deposit.interest = interest;}
156
157     } catch (NumberFormatException e) {
158         JOptionPane.showMessageDialog(null, "Введите_число_без_знака_процента");
159     }
160
161     deposit.capitaliz = ch_capital.isSelected(); //капитализация
162     deposit.proc = c_type_proc.getSelectedIndex();//капитализация, 0 - ежемесячно, 1 -
163     ежеквартально, 2 - ежегодно
164
165     //
166     //ВЫЧИСЛЕНИЯ
167     //
168     deposit.calculate();//вычисления
169
170     //
171     //ВЫВОД ИНФОРМАЦИИ
172     //
173     //текстбоксы
174     t_prok_out.setText(String.format("%.2f", deposit.finalproc));//форматный вывод
175     t_ost_vklad.setText(String.format("%.2f", deposit.deposit));
176     //таблица

```

[illegible]


```

302                                     .addGroup(gl_contentPane.createParallelGroup
303                                         (Alignment.LEADING)
304                                         .addComponent(label)
305                                         .addComponent(t_year, GroupLayout.
306                                             DEFAULT_SIZE, 74, Short.
307                                             MAX_VALUE))))))
308                                     .addGroup(gl_contentPane.createSequentialGroup())
309                                     .addContainerGap()
310                                     .addComponent(l_proc)
311                                     .addPreferredGap(ComponentPlacement.RELATED)
312                                     .addComponent(t_proc, GroupLayout.PREFERRED_SIZE, 29,
313                                         GroupLayout.PREFERRED_SIZE)
314                                     .addPreferredGap(ComponentPlacement.RELATED)
315                                     .addComponent(label_2))
316                                     .addGroup(gl_contentPane.createSequentialGroup())
317                                     .addContainerGap()
318                                     .addComponent(ch_capital))
319                                     .addGroup(gl_contentPane.createSequentialGroup())
320                                     .addContainerGap()
321                                     .addComponent(label_1)
322                                     .addPreferredGap(ComponentPlacement.RELATED)
323                                     .addComponent(c_type_proc, GroupLayout.PREFERRED_SIZE,
324                                         GroupLayout.DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
325                                     ))
326                                     .addGroup(gl_contentPane.createSequentialGroup())
327                                     .addContainerGap()
328                                     .addComponent(b_go, GroupLayout.DEFAULT_SIZE, 266, Short.
329                                         MAX_VALUE)
330                                     .addGap(18)))
331                                     .addGap(18)
332                                     .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING)
333                                         .addComponent(scrollPane, GroupLayout.DEFAULT_SIZE, 277, Short.MAX_VALUE)
334                                         .addGroup(gl_contentPane.createSequentialGroup())
335                                             .addComponent(label_3)
336                                             .addPreferredGap(ComponentPlacement.RELATED)
337                                             .addComponent(t_ost_vklad, GroupLayout.DEFAULT_SIZE, 180, Short.
338                                                 MAX_VALUE))
339                                         .addGroup(gl_contentPane.createSequentialGroup())
340                                             .addComponent(label_4)
341                                             .addPreferredGap(ComponentPlacement.RELATED)
342                                             .addComponent(t_prok_out, GroupLayout.DEFAULT_SIZE, 140, Short.
343                                                 MAX_VALUE)))
344                                     .addContainerGap())
345                                     );
346                                     gl_contentPane.setVerticalGroup(
347                                         gl_contentPane.createParallelGroup(Alignment.LEADING)
348                                         .addGroup(gl_contentPane.createSequentialGroup())
349                                             .addContainerGap()
350                                             .addGroup(gl_contentPane.createParallelGroup(Alignment.TRAILING)
351                                                 .addGroup(gl_contentPane.createParallelGroup(Alignment.BASELINE)
352                                                     .addComponent(l_sum)
353                                                     .addComponent(t_sum, GroupLayout.PREFERRED_SIZE, GroupLayout.
354                                                         DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
355                                                     .addComponent(label_3))
356                                                 .addComponent(t_ost_vklad, GroupLayout.PREFERRED_SIZE, 27, GroupLayout.
357                                                     PREFERRED_SIZE))
358                                             .addPreferredGap(ComponentPlacement.RELATED)
359                                             .addGroup(gl_contentPane.createParallelGroup(Alignment.TRAILING)
360                                                 .addGroup(gl_contentPane.createParallelGroup(Alignment.BASELINE)
361                                                     .addComponent(l_date)
362                                                     .addComponent(t_day, GroupLayout.PREFERRED_SIZE, GroupLayout.
363                                                         DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
364                                                     .addComponent(t_year, GroupLayout.PREFERRED_SIZE, GroupLayout.
365                                                         DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
366                                                     .addComponent(t_month, GroupLayout.PREFERRED_SIZE, GroupLayout.
367                                                         DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE)
368                                                     .addComponent(label_4))
369                                                 .addComponent(t_prok_out, GroupLayout.PREFERRED_SIZE, GroupLayout.
370                                                     DEFAULT_SIZE, GroupLayout.PREFERRED_SIZE))
371                                             .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING)
372                                                 .addGroup(gl_contentPane.createSequentialGroup())
373                                                     .addGap(12)
374                                                     .addGroup(gl_contentPane.createParallelGroup(Alignment.LEADING)
375                                                         .addGroup(gl_contentPane.createSequentialGroup())
376                                                             .addGroup(gl_contentPane.createParallelGroup(
377                                                                 Alignment.BASELINE)
378                                                                 .addComponent(l_time_vklad)
379                                                                 .addComponent(t_time_vklad, GroupLayout.
380                                                                     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
394     table = new JTable();
395     scrollPane.setViewportViewView(table);
396     table.setShowVerticalLines(true);
397     table.setRowSelectionAllowed(false);
398     contentPane.setLayout(gl_contentPane);
399 }
400 }

```