01	1с 8.2. Практическое пособие разработчика. Радченко М.Г. Код.

1	1с 8.2. Практическое пособие разработчика. Радченко М.Г. Код.
	1. Добавление шаблона. **2. Справочники.** **3. Проверка изменения БД.** **4. Документы.** **5. Формы. Модули. Контекст модуля формы. Синтаксис помощник. Анализ кода с помощью отладчика. Объекты.** **6. Регистры накопления.** **6.2. Коллекция.** **7. Отчет простейший.** **8. Макеты. Редактирование макетов и форм.** **9. Периодические регистры сведений.** **9.2. Автоматическая подстановка цены в документ при выборе номенклатуры** **10. Перечисления.** **10.2. ОказаниеУслуги – регистрация расхода номенклатуры-материала.** **11. Проведение документа по нескольким регистрам.** **12. Оборотные регистры накопления.** **13. Отчеты. Реестр документов.** **Способы доступа к данным. Система компоновки данных. Макет.** **13.1.Отчет - Реестр документов оказание услуги.** **13.2. Выбор данных из 2-х таблиц.** **Отчет РейтингУслуг.** **13.3. Отчет ВыручкаМастеров.** **Вывод данных по всем дням в выбранном периоде.** **13.4. Отчет ОбъемВыручки. Диаграмма.** **13.5. Отчет Переченьуслуг. Получение актуальных значений из периодического регистра сведений.** **13.6. Отчет. Рейтинг клиентов. Вычисляемые поля.** **13.7. Отчет универсальный. Вывод данных в таблицу.** **Продажи.Обороты.** **14. Оптимизация проведения документа «Оказание услуги»** **14.2. Автоматический расчет стоимости.** **Определение стоимости по среднему.** **14.3. Контроль остатков.** **15. План видов характеристик.** **15.2. Отчет, использующий характеристики.** **16. Бухгалтерский учет. План счетов.** **16.2. Регистр бухгалтерии.** **16.3. Отчет Оборотно-сальдовая ведомость.** **17. План видов расчета.** **17.2. Регистр расчета.** **18. Использование регистра расчета.** **18.2. Отчет Перерасчет.** **18.3. Расчет записей регистра расчета.** **18.4. Диаграмма начислений.** **19. Поиск в базе данных.** **20. Выполнение заданий по расписанию.** **21. Редактирование движений в форме документа.**
2	**Программное редактирование записей регистра.**

22. Список пользователей и их роли.
22.2. Ограничение доступа к данным на уровне записей и полей базы данных
23. Рабочий стол и настройка командного интерфейса.
24. Обмен данными.
25. Функциональные опции.
25.2. Опция УчетКлиентов.
26. Подборы и ввод на основании.
26.2. Множественный подбор с использованием множественного выбора.
26.3. Ввод на основании.
26.4. Критерий отбора.
27. Приемы разработки форм.
27.2. Связанные списки.
27.3. Оформление строк в виде списка.
27.4. Вычисляемые колонки в списках.
27.5. Список выбора для поля ввода.
27.6. Форма выбора для поля, содержащего ссылочный реквизит
27.7. Автоматическая проверка заполнения.
28.8. Использование параметризированных команд.
29.1. Объекты встроенного языка для работы с прикалдными объектами.
29.2. Манипулирование данными объектов.
29.3. Константы.
29.4. Справочники.

005-КОД 1-28.

```
{1,
{19,
{"Практическое пособие разработчика",1,0,"",""},
{5,
{"Занятие 4. Документы",1,0,"",""},
{0,
{"4.1 Процедура МатериалыКоличествоПриИзменении()",0,0,"4.1","СтрокаТабличнойЧасти
= Элементы.Материалы.ТекущиеДанные;
СтрокаТабличнойЧасти.Сумма = СтрокаТабличнойЧасти.Количество *
СтрокаТабличнойЧасти.Цена;
"}
},
{0,
{"4.2 Процедура РассчитатьСумму()",0,0,"4.2","Процедура
РассчитатьСумму(СтрокаТабличнойЧасти) Экспорт

        СтрокаТабличнойЧасти.Сумма = СтрокаТабличнойЧасти.Количество *
        СтрокаТабличнойЧасти.Цена;

КонецПроцедуры
"}
},
{0,
{"4.3 Процедура
МатериалыКоличествоПриИзменении()",0,0,"4.3","РаботаСДокументами.РассчитатьСумму(
СтрокаТабличнойЧасти);"}
},
{0,
{"4.4 Процедура МатериалыЦенаПриИзменении()",0,0,"4.4","СтрокаТабличнойЧасти =
Элементы.Материалы.ТекущиеДанные;
РаботаСДокументами.РассчитатьСумму(СтрокаТабличнойЧасти);
"}
```

```
},
{0,
{"4.5 Процедура ПеречеьНоменклатурыКоличествоПриИзменении() и
ПеречеьНоменклатурыЦенаПриИзменении()",0,0,"","СтрокаТабличнойЧасти = Элементы.
ПеречеьНоменклатуры.ТекущиеДанные;
РаботаСДокументами.РассчитатьСумму(СтрокаТабличнойЧасти);
"}
}
},
{1,
{"Занятие 8. Редактирование макетов и форм",1,0,"",""},
{0,
{"8.1 Печать формы документа в модуле менеджера",0,0,"8.1","
Процедура Печать(ТабДок, Ссылка) Экспорт
	//{{_КОНСТРУКТОР_ПЕЧАТИ(Печать)
	Макет = Документы.ОказаниеУслуги.ПолучитьМакет(""Печать"");
	Запрос = Новый Запрос;
	Запрос.Текст =
	""ВЫБРАТЬ
	|	ОказаниеУслуги.Дата,
	|	ОказаниеУслуги.Клиент,
	|	ОказаниеУслуги.Мастер,
	|	ОказаниеУслуги.Номер,
	|	ОказаниеУслуги.Склад,
	|	ОказаниеУслуги.ПеречеьНоменклатуры.(
	|		НомерСтроки,
	|		Номенклатура,
	|		Количество,
	|		Цена,
	|		Сумма
	|	)
	|ИЗ
	|	Документ.ОказаниеУслуги КАК ОказаниеУслуги
	|ГДЕ
	|	ОказаниеУслуги.Ссылка В (&Ссылка)"";
	Запрос.Параметры.Вставить(""Ссылка"", Ссылка);
	Выборка = Запрос.Выполнить().Выбрать();


	ОбластьЗаголовок = Макет.ПолучитьОбласть(""Заголовок"");
	Шапка = Макет.ПолучитьОбласть(""Шапка"");
	ОбластьПеречеьНоменклатурыШапка =
Макет.ПолучитьОбласть(""ПеречеьНоменклатурыШапка"");
	ОбластьПеречеьНоменклатуры =
Макет.ПолучитьОбласть(""ПеречеьНоменклатуры"");
	ОбластьИтог = Макет.ПолучитьОбласть(""Всего"");
	ТабДок.Очистить();

	ВставлятьРазделительСтраниц = Ложь;
	Пока Выборка.Следующий() Цикл
		Если ВставлятьРазделительСтраниц Тогда
			ТабДок.ВывестиГоризонтальныйРазделительСтраниц();
		КонецЕсли;

		ТабДок.Вывести(ОбластьЗаголовок);

		Шапка.Параметры.Заполнить(Выборка);
```

```
            ТабДок.Вывести(Шапка, Выборка.Уровень());

            ТабДок.Вывести(ОбластьПеречеьНоменклатурыШапка);
            ВыборкаПеречеьНоменклатуры = Выборка.ПеречеьНоменклатуры.Выбрать();
            СуммаИтог = 0;
            Пока ВыборкаПеречеьНоменклатуры.Следующий() Цикл

        ОбластьПеречеьНоменклатуры.Параметры.Заполнить(ВыборкаПеречеьНоменклатуры);
                ТабДок.Вывести(ОбластьПеречеьНоменклатуры, ВыборкаПеречеьНоменклатуры.Уровень());
                СуммаИтог = СуммаИтог + ВыборкаПеречеьНоменклатуры.Сумма;
            КонецЦикла;

            ОбластьИтог.Параметры.ВсегоПоДокументу = СуммаИтог;
            ТабДок.Вывести(ОбластьИтог);

            ВставлятьРазделительСтраниц = Истина;
        КонецЦикла;
        //}}
КонецПроцедуры
"}
}
},
{2,
{"Занятие 9. Периодический регистр сведений",1,0,"",""},
{0,
{"9.1 Функция РозничнаяЦена()",0,0,"9.1","Функция РозничнаяЦена(АктуальнаяДата, ЭлементНоменклатуры) Экспорт

        // Создать вспомогательный объект Отбор
        Отбор = Новый Структура(""Номенклатура"", ЭлементНоменклатуры);

        // Получить актуальные значения ресурсов регистра
        ЗначенияРесурсов = РегистрыСведений.Цены.ПолучитьПоследнее(АктуальнаяДата, Отбор);
        Возврат ЗначенияРесурсов.Цена;

КонецФункции
"}
},
{0,
{"9.2 Процедура ПеречеьНоменклатурыНоменклатураПриИзменении()",0,0,"9.2","//
Получить текущую строку табличной части
СтрокаТабличнойЧасти = Элементы.ПеречеьНоменклатуры.ТекущиеДанные;

// Установить цену
СтрокаТабличнойЧасти.Цена = РаботаСоСправочниками.РозничнаяЦена(Объект.Дата, СтрокаТабличнойЧасти.Номенклатура);

// Пересчитать сумму строки
РаботаСДокументами.РассчитатьСумму(СтрокаТабличнойЧасти);
"}
}
},
{1,
{"Занятие 10. Перечисления",1,0,"",""},
```

```
{0,
{"10.2 Движения документа ОказаниеУслуги",0,0,"10.2","Процедура
ОбработкаПроведения(Отказ, Режим)

	//{{__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ
	// Данный фрагмент построен конструктором.
	// При повторном использовании конструктора, внесенные вручную изменения будут
утеряны!!!
	Движения.ОстаткиМатериалов.Записывать = Истина;

	Для Каждого ТекСтрокаПеречеьНоменклатуры Из ПеречеьНоменклатуры Цикл
		Если ТекСтрокаПеречеьНоменклатуры.Номенклатура.ВидНоменклатуры =
Перечисления.ВидыНоменклатуры.Материал Тогда
			// регистр ОстаткиМатериалов Расход
			Движение = Движения.ОстаткиМатериалов.Добавить();
			Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
			Движение.Период = Дата;
			Движение.Материал = ТекСтрокаПеречеьНоменклатуры.Номенклатура;
			Движение.Склад = Склад;
			Движение.Количество = ТекСтрокаПеречеьНоменклатуры.Количество;
		КонецЕсли;
	КонецЦикла;
	//}}__КОНСТРУКТОР_ДВИЖЕНИЙ_РЕГИСТРОВ

КонецПроцедуры
"}
}
},
{4,
{"Занятие 11. Проведение документа по нескольким регистрам",1,0,"",""},
{0,
{"11.1  Движения документа ПриходнаяНакладная по регистру ""Стоимость
материалов""",0,0,"11.1","// регистр Стоимость Материалов Приход
Движение = Движения.СтоимостьМатериалов.Добавить();
Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
Движение.Период = Дата;
Движение.Материал = ТекСтрокаМатериалы.Материал;
Движение.Стоимость = ТекСтрокаМатериалы.Сумма;
"}
},
{0,
{"11.2 Движения документа ПриходнаяНакладная",0,0,"11.2","Процедура
ОбработкаПроведения(Отказ, Режим)

	Движения.ОстаткиМатериалов.Записывать = Истина;
	Движения.СтоимостьМатериалов.Записывать = Истина;

	Для Каждого ТекСтрокаМатериалы Из Материалы Цикл

		// регистр ОстаткиМатериалов Приход
		Движение = Движения.ОстаткиМатериалов.Добавить();
		Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
		Движение.Период = Дата;
		Движение.Материал = ТекСтрокаМатериалы.Материал;
		Движение.Склад = Склад;
		Движение.Количество = ТекСтрокаМатериалы.Количество;
```

```
                // регистр Стоимость Материалов Приход
                Движение = Движения.СтоимостьМатериалов.Добавить();
                Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
                Движение.Период = Дата;
                Движение.Материал = ТекСтрокаМатериалы.Материал;
                Движение.Стоимость = ТекСтрокаМатериалы.Сумма;
        КонецЦикла;

КонецПроцедуры
"}
},
{0,
{"11.3  Движения документа ОказаниеУслуги по регистру ""Стоимость
материалов""",0,0,"11.3","// регистр СтоимостьМатериалов Расход
Движение = Движения.СтоимостьМатериалов.Добавить();
Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
Движение.Период = Дата;
Движение.Материал = ТекСтрокаПеречеНоменклатуры.Номенклатура;
Движение.Стоимость =
ТекСтрокаПеречеНоменклатуры.Количество*ТекСтрокаПеречеНоменклатуры.Стоимос
ть;
"}
},
{0,
{"11.4  Движения документа ОказаниеУслуги",0,0,"11.4","Процедура
ОбработкаПроведения(Отказ, Режим)

        Движения.ОстаткиМатериалов.Записывать = Истина;
        Движения.СтоимостьМатериалов.Записывать = Истина;

        Для Каждого ТекСтрокаПеречеНоменклатуры Из ПеречеНоменклатуры Цикл
                Если ТекСтрокаПеречеНоменклатуры.Номенклатура.ВидНоменклатуры =
Перечисления.ВидыНоменклатуры.Материал Тогда

                        // регистр ОстаткиМатериалов Расход
                        Движение = Движения.ОстаткиМатериалов.Добавить();
                        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
                        Движение.Период = Дата;
                        Движение.Материал = ТекСтрокаПеречеНоменклатуры.Номенклатура;
                        Движение.Склад = Склад;
                        Движение.Количество = ТекСтрокаПеречеНоменклатуры.Количество;

                        // регистр СтоимостьМатериалов Расход
                        Движение = Движения.СтоимостьМатериалов.Добавить();
                        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
                        Движение.Период = Дата;
                        Движение.Материал = ТекСтрокаПеречеНоменклатуры.Номенклатура;
                        Движение.Стоимость =
ТекСтрокаПеречеНоменклатуры.Количество*ТекСтрокаПеречеНоменклатуры.Стоимос
ть;
                КонецЕсли;
        КонецЦикла;

КонецПроцедуры
"}
}
},
```

```
{2,
{"Занятие 12. Оборотный регистр накопления",1,0,"","},
{0,
{"12.1  Движения документа ОказаниеУслуги по регистру ""Продажи""",0,0,"12.1","//
Регистр Продажи
Движение = Движения.Продажи.Добавить();
Движение.Период = Дата;
Движение.Номенклатура = ТекСтрокаПеречеьНоменклатуры.Номенклатура;
Движение.Клиент = Клиент;
Движение.Мастер = Мастер;
Движение.Количество = ТекСтрокаПеречеьНоменклатуры.Количество;
Движение.Выручка = ТекСтрокаПеречеьНоменклатуры.Сумма;
Движение.Стоимость = ТекСтрокаПеречеьНоменклатуры.Стоимость *
ТекСтрокаПеречеьНоменклатуры.Количество;
"}
},
{0,
{"12.2 Движения документа ОказаниеУслуги",0,0,"12.2","Процедура
ОбработкаПроведения(Отказ, Режим)

        Движения.ОстаткиМатериалов.Записывать = Истина;
        Движения.СтоимостьМатериалов.Записывать = Истина;
        Движения.Продажи.Записывать = Истина;

        Для Каждого ТекСтрокаПеречеьНоменклатуры Из ПеречеьНоменклатуры Цикл
                Если ТекСтрокаПеречеьНоменклатуры.Номенклатура.ВидНоменклатуры =
Перечисления.ВидыНоменклатуры.Материал Тогда

                        // регистр ОстаткиМатериалов Расход
                        Движение = Движения.ОстаткиМатериалов.Добавить();
                        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
                        Движение.Период = Дата;
                        Движение.Материал = ТекСтрокаПеречеьНоменклатуры.Номенклатура;
                        Движение.Склад = Склад;
                        Движение.Количество = ТекСтрокаПеречеьНоменклатуры.Количество;

                        // регистр СтоимостьМатериалов Расход
                        Движение = Движения.СтоимостьМатериалов.Добавить();
                        Движение.ВидДвижения = ВидДвиженияНакопления.Расход;
                        Движение.Период = Дата;
                        Движение.Материал = ТекСтрокаПеречеьНоменклатуры.Номенклатура;
                        Движение.Стоимость =
ТекСтрокаПеречеьНоменклатуры.Количество*ТекСтрокаПеречеьНоменклатуры.Стоимос
ть;
                КонецЕсли;

                // Регистр Продажи
                Движение = Движения.Продажи.Добавить();
                Движение.Период = Дата;
                Движение.Номенклатура = ТекСтрокаПеречеьНоменклатуры.Номенклатура;
                Движение.Клиент = Клиент;
                Движение.Мастер = Мастер;
                Движение.Количество = ТекСтрокаПеречеьНоменклатуры.Количество;
                Движение.Выручка = ТекСтрокаПеречеьНоменклатуры.Сумма;
                Движение.Стоимость = ТекСтрокаПеречеьНоменклатуры.Стоимость *
ТекСтрокаПеречеьНоменклатуры.Количество;
        КонецЦикла;
```

```
КонецПроцедуры
"}
}
},
{3,
{"Занятие 13. Отчеты",1,0,"",""},
{0,
{"13.5 Условие запроса",0,0,"13.5","спрНоменклатура.ЭтоГруппа = ЛОЖЬ"}
},
{0,
{"13.9 Выражение для расчета значения параметра
КонецПериода",0,0,"13.9","КонецПериода(&ДатаОкончания,""День"")"}
},
{0,
{"13.14 Выражение для расчета вычисляемого поля Доход",0,0,"13.14","Выручка -
Стоимость"}
}
},
{14,
{"Занятие 14. Оптимизация процедуры проведения документа Оказание услуги",1,0,"",""},
{0,
{"14.17 Использование менеджера временных таблиц",0,0,"14.17","// Создать менеджер
временных таблиц
МенеджерВТ = Новый МенеджерВременныхТаблиц;

Запрос = Новый Запрос;

// Укажем, какой менеджер временных таблиц использует этот запрос
Запрос.МенеджерВременныхТаблиц = МенеджерВТ;

"}
},
{0,
{"14.22 Создание второго запроса",0,0,"14.22","Запрос2 = Новый Запрос;
Запрос2.МенеджерВременныхТаблиц = МенеджерВТ;
Запрос2.Текст = """";
"}
},
{0,
{"14.24 Условие виртуальной таблицы
СтоимостьМатериалов.Остатки",0,0,"14.24","Материал В (ВЫБРАТЬ
НоменклатураДокумента.Номенклатура ИЗ НоменклатураДокумента)"}
},
{0,
{"14.26 Условие виртуальной таблицы ОстаткиМатериалов.Остатки",0,0,"14.26","Материал
В (ВЫБРАТЬ НоменклатураДокумента.Номенклатура ИЗ НоменклатураДокумента)"}
},
{0,
{"14.29 Выражение для расчета поля СтоимостьОстаток в
запросе",0,0,"14.29","ЕСТЬNULL(СтоимостьМатериаловОстатки.СтоимостьОстаток, 0)"}
},
{0,
{"14.31 Выполнение второго запроса",0,0,"14.31","Результат = Запрос2.Выполнить();
                                "}
},
{0,
```

```
{"14.32 Расчет Стоимости",0,0,"14.32","Если ВыборкаДетальныеЗаписи.Количество = 0 Тогда
        СтоимостьМатериала = 0;
Иначе
        СтоимостьМатериала = ВыборкаДетальныеЗаписи.Стоимость /
ВыборкаДетальныеЗаписи.Количество;
КонецЕсли;
"}
},
{0,
{"14.33 Запись пустых наборов движений",0,0,"14.33","// Запишем пустые наборы записей
чтобы читать остатки без учета данных в документе
Движения.СтоимостьМатериалов.Записать();
Движения.ОстаткиМатериалов.Записать();
"}
},
{0,
{"14.34 Выгрузка результатов запроса в ТЗ",0,0,"14.34","ТЗ = Результат.Выгрузить();"}
},
{0,
{"14.36 Заготовка контроля остатков при проведении
документа",0,0,"14.36","Движения.Записать();

Если Режим = РежимПроведенияДокумента.Оперативный Тогда
        // Проверить отрицательные остатки
        Запрос3 = Новый Запрос;
        Запрос3.МенеджерВременныхТаблиц = МенеджерВТ;
        Запрос3.Текст = """";
КонецЕсли;
"}
},
{0,
{"14.37 Условие виртуальной таблицы ОстаткиМатериалов.Остатки",0,0,"14.37","Материал
В (ВЫБРАТЬ НоменклатураДокумента.Номенклатура ИЗ НоменклатураДокумента) И Склад
= &Склад"}
},
{0,
{"14.38 Условие контроля запроса
остатков",0,0,"14.38","ОстаткиМатериаловОстатки.КоличествоОстаток < 0"}
},
{0,
{"14.40 Выполнение третьего запроса",0,0,"14.40","Запрос3.УстановитьПараметр(""Склад"",
Склад);

Результат = Запрос3.Выполнить();
ВыборкаДетальныеЗаписи = Результат.Выбрать();

Пока ВыборкаДетальныеЗаписи.Следующий() Цикл
        Сообщение = Новый СообщениеПользователю();
        Сообщение.Текст = ""Не хватает "" + Строка(-
ВыборкаДетальныеЗаписи.КоличествоОстаток) + "" единиц материала """""""" +
ВыборкаДетальныеЗаписи.Материал + """""""""""";
        Сообщение.Сообщить();
        Отказ = Истина;
КонецЦикла;
"}
},
{0,
```

```
{"14.41 Блокировка данных перед чтением",0,0,"14.41","// Установим необходимость
блокировки данных в регистрах СтоимостьМатериалов и ОстаткиМатериалов
Движения.СтоимостьМатериалов.БлокироватьДляИзменения = Истина;
Движения.ОстаткиМатериалов.БлокироватьДляИзменения = Истина;
   "}
}
},
{5,
{"Занятие 15. План видов характеристик",1,0,"",""},
{0,
{"15.1 Обработчик события формы списка справочника ВариантыНоменклатуры
ПриСозданииНаСервере()",0,0,"15.1","Если Параметры.Отбор.Свойство(""Владелец"")
Тогда
        Элементы.Код.Видимость = Ложь;
КонецЕсли;
"}
},
{0,
{"15.2 Обработчик события формы списка регистра сведений
ЗначенияСвойствНоменклатуры ПриСозданииНаСервере()",0,0,"15.2","Если
Параметры.Отбор.Свойство(""НаборСвойств "") Тогда
        Элементы.НаборСвойств.Видимость = Ложь;
КонецЕсли;
"}
},
{0,
{"15.3 Доработка процедуры ОбработкаПроведения() документа
ПриходнаяНакладная",0,0,"15.3","Движение.НаборСвойств =
ТекСтрокаМатериалы.НаборСвойств;"}
},
{0,
{"15.4 Доработка процедуры ОбработкаПроведения() документа ОказаниеУслуг -
1",0,0,"15.4","Движение.НаборСвойств = ВыборкаДетальныеЗаписи.НаборСвойств;"}
},
{0,
{"15.5 Доработка процедуры ОбработкаПроведения() документа ОказаниеУслуг -
2",0,0,"15.5","Запрос.Текст =
        ""ВЫБРАТЬ
        |       ОказаниеУслугиПеречеьНоменклатуры.Номенклатура,
        |       ОказаниеУслугиПеречеьНоменклатуры.Номенклатура.ВидНоменклатуры КАК
ВидНоменклатуры,
        |       ОказаниеУслугиПеречеьНоменклатуры.НаборСвойств,
        |       СУММА(ОказаниеУслугиПеречеьНоменклатуры.Количество) КАК
КоличествоВДокументе,
        |       СУММА(ОказаниеУслугиПеречеьНоменклатуры.Сумма) КАК
СуммаВДокументе
        |ПОМЕСТИТЬ НоменклатураДокумента
        |ИЗ
        |       Документ.ОказаниеУслуги.ПеречеьНоменклатуры КАК
ОказаниеУслугиПеречеьНоменклатуры
        |ГДЕ
        |       ОказаниеУслугиПеречеьНоменклатуры.Ссылка = &Ссылка
        |
        |СГРУППИРОВАТЬ ПО
        |       ОказаниеУслугиПеречеьНоменклатуры.Номенклатура,
        |       ОказаниеУслугиПеречеьНоменклатуры.Номенклатура.ВидНоменклатуры,
        |       ОказаниеУслугиПеречеьНоменклатуры.НаборСвойств"";
```

```
...
Запрос2 = Новый Запрос;
Запрос2.МенеджерВременныхТаблиц = МенеджерВТ;
Запрос2.Текст = ""ВЫБРАТЬ
                |    НоменклатураДокумента.Номенклатура,
                |    НоменклатураДокумента.ВидНоменклатуры,
                |    НоменклатураДокумента.НаборСвойств,
                |    НоменклатураДокумента.КоличествоВДокументе,
                |    НоменклатураДокумента.СуммаВДокументе,
..."}
}
},
{5,
{"Занятие 16. Бухгалтерский учет",1,0,"","},
{0,
{"16.1  Движения документа ПриходнаяНакладная по регистру
""Управленческий""",0,0,"16.1","// Регистр Управленческий
Движение = Движения.Управленческий.Добавить();
Движение.СчетДт = ПланыСчетов.Основной.Товары;
Движение.СчетКт = ПланыСчетов.Основной.РасчетыСПоставщиками;
Движение.Период = Дата;
Движение.Сумма = ТекСтрокаМатериалы.Сумма;
Движение.КоличествоДт = ТекСтрокаМатериалы.Количество;
Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Материалы] =
ТекСтрокаМатериалы.Материал;
"}
},
{0,
{"16.2 Движения документа ПриходнаяНакладная",0,0,"16.2","Процедура
ОбработкаПроведения(Отказ, Режим)

        Движения.ОстаткиМатериалов.Записывать = Истина;
        Движения.СтоимостьМатериалов.Записывать = Истина;
        Движения.Управленческий.Записывать = Истина;

        Для Каждого ТекСтрокаМатериалы Из Материалы Цикл

                // регистр ОстаткиМатериалов Приход
                Движение = Движения.ОстаткиМатериалов.Добавить();
                Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
                Движение.Период = Дата;
                Движение.Материал = ТекСтрокаМатериалы.Материал;
                Движение.НаборСвойств = ТекСтрокаМатериалы.НаборСвойств;
                Движение.Склад = Склад;
                Движение.Количество = ТекСтрокаМатериалы.Количество;

                // регистр Стоимость Материалов Приход
                Движение = Движения.СтоимостьМатериалов.Добавить();
                Движение.ВидДвижения = ВидДвиженияНакопления.Приход;
                Движение.Период = Дата;
                Движение.Материал = ТекСтрокаМатериалы.Материал;
                Движение.Стоимость = ТекСтрокаМатериалы.Сумма;

                // Регистр Управленческий
                Движение = Движения.Управленческий.Добавить();
                Движение.СчетДт = ПланыСчетов.Основной.Товары;
                Движение.СчетКт = ПланыСчетов.Основной.РасчетыСПоставщиками;
```

```
                Движение.Период = Дата;
                Движение.Сумма = ТекСтрокаМатериалы.Сумма;
                Движение.КоличествоДт = ТекСтрокаМатериалы.Количество;
                Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Материалы]
= ТекСтрокаМатериалы.Материал;
        КонецЦикла;

КонецПроцедуры
"}
},
{0,
{"16.3  Движения документа ОказаниеУслуги по регистру ""Управленческий""",0,0,"16.3","//
Регистр Управленческий
// Первая проводка:          Д 62(ДебиторскаяЗадолженность) – К 90 (Капитал)  Розничная
сумма
Движение = Движения.Управленческий.Добавить();
Движение.СчетДт = ПланыСчетов.Основной.ДебиторскаяЗадолженность;
Движение.СчетКт = ПланыСчетов.Основной.Капитал;
Движение.Период = Дата;
Движение.Сумма = ВыборкаДетальныеЗаписи.СуммаВДокументе;
Движение.СубконтоДт[ПланыВидовХарактеристик.ВидыСубконто.Клиенты] = Клиент;
// Вторая проводка: Д 90 (Капитал) – К 41 (Товары) – себестоимость
Движение = Движения.Управленческий.Добавить();
Движение.СчетДт = ПланыСчетов.Основной.Капитал;
Движение.СчетКт = ПланыСчетов.Основной.Товары;
Движение.Период = Дата;
Движение.Сумма = СтоимостьМатериала *
ВыборкаДетальныеЗаписи.КоличествоВДокументе;
Движение.КоличествоКт = ВыборкаДетальныеЗаписи.КоличествоВДокументе;
Движение.СубконтоКт[ПланыВидовХарактеристик.ВидыСубконто.Материалы] =
ВыборкаДетальныеЗаписи.Номенклатура;
"}
},
{0,
{"16.4 Запись движений регистра
бухгалтерии",0,0,"16.4","Движения.Управленческий.Записывать = Истина; "}
},
{0,
{"16.5 Выражение для расчета параметров НачалоПериода и
КонецПериода",0,0,"16.5","&Период.ДатаНачала
&Период.ДатаОкончания
"}
}
},
{11,
{"Занятие 18. Использование регистра расчета",1,0,"",""},
{0,
{"18.2 Обработчик проведения документа НачисленияСотрудникам",0,0,"18.2","//
Записываем движения регистров
Движения.Начисления.Записать();

// Получим список всех сотрудников, содержащихся в документе
Запрос = Новый Запрос(
        ""ВЫБРАТЬ РАЗЛИЧНЫЕ
        | НачисленияСотрудникамНачисления.Сотрудник
        |ИЗ
        | Документ.НачисленияСотрудникам.Начисления
```

```
        | КАК НачисленияСотрудникамНачисления
        |
        |ГДЕ
        | НачисленияСотрудникамНачисления.Ссылка = &ТекущийДокумент""");
Запрос.УстановитьПараметр(""ТекущийДокумент"", Ссылка);

// Сформируем список сотрудников
ТаблЗнач = Запрос.Выполнить().Выгрузить();
МассивСотрудников = ТаблЗнач.ВыгрузитьКолонку(""Сотрудник"");

//Вызов процедуры РассчитатьНачисления из общего модуля
ПроведениеРасчетов.РассчитатьНачисления(Движения.Начисления,
ПланыВидовРасчета.ОсновныеНачисления.Оклад, МассивСотрудников);
Движения.Начисления.Записать( , Истина);

ПроведениеРасчетов.РассчитатьНачисления(Движения.Начисления,
ПланыВидовРасчета.ОсновныеНачисления.Премия, МассивСотрудников);
Движения.Начисления.Записать( , Истина);
"}
},
{0,
{"18.3 Заготовка процедуры РассчитатьНачисления",0,0,"18.3","Процедура
РассчитатьНачисления(НаборЗаписейРегистра, ТребуемыйВидРасчета, СписокСотрудников)
Экспорт

        Регистратор=НаборЗаписейРегистра.Отбор.Регистратор.Значение;

        // Рассчитать первичные записи
        Если ТребуемыйВидРасчета = ПланыВидовРасчета.ОсновныеНачисления.Оклад Тогда

        // Рассчитать вторичные записи
        ИначеЕсли ТребуемыйВидРасчета =
ПланыВидовРасчета.ОсновныеНачисления.Премия Тогда
        КонецЕсли;

КонецПроцедуры
"}
},
{0,
{"18.4 Изменение процедуры РасчитатьНачисления",0,0,"18.4","Запрос = Новый Запрос;
Запрос.Текст =
        ""ВЫБРАТЬ
        | НачисленияДанныеГрафика.ЗначениеПериодДействия КАК Норма,
        | НачисленияДанныеГрафика.ЗначениеФактическийПериодДействия КАК Факт,
        | НачисленияДанныеГрафика.НомерСтроки КАК НомерСтроки
        |ИЗ
        | РегистрРасчета.Начисления.ДанныеГрафика(Регистратор = &Регистратор И
        | ВидРасчета = &ВидРасчета И Сотрудник В (&СписокСотрудников))
        | КАК НачисленияДанныеГрафика"";

Запрос.УстановитьПараметр(""Регистратор"", Регистратор);
Запрос.УстановитьПараметр(""ВидРасчета"", ТребуемыйВидРасчета);
Запрос.УстановитьПараметр(""СписокСотрудников"", СписокСотрудников);

ВыборкаРезультата = Запрос.Выполнить().Выбрать();
"}
},
```

```
{0,
{"18.5 Добавление обхода набора записей и расчета первичных записей",0,0,"18.5","Для
Каждого ЗаписьРегистра Из НаборЗаписейРегистра Цикл
        СтруктураНомер = Новый Структура(""НомерСтроки"");
        СтруктураНомер.НомерСтроки = ЗаписьРегистра.НомерСтроки;
        ВыборкаРезультата.Сбросить();
        Если ВыборкаРезультата.НайтиСледующий(СтруктураНомер) Тогда
                Если ВыборкаРезультата.Норма = 0 Тогда
                        Сообщение = Новый СообщениеПользователю;
                        Сообщение.Текст = ""Вид расчета: Оклад – Нет рабочих дней в заданном
периоде"";
                        Сообщение.Сообщить();
                        ЗаписьРегистра.Результат = 0;
                Иначе

                        // Рассчитать оклад по фактическому периоду и исходным данным
                        ЗаписьРегистра.Результат = (ЗаписьРегистра.ИсходныеДанные
/ВыборкаРезультата.Норма) * ВыборкаРезультата.Факт;
                        Сообщение = Новый СообщениеПользователю;
                        Сообщение.Текст = ""Выполнен расчет"" + ЗаписьРегистра.Регистратор +
"" – ""  + ЗаписьРегистра.ВидРасчета + "" – ""
                         + ЗаписьРегистра.Сотрудник;
                        Сообщение.Сообщить();
                КонецЕсли;
        КонецЕсли;
КонецЦикла;
"}
},
{0,
{"18.6 Добавление текста запроса во вторую строку условия",0,0,"18.6","Запрос = Новый
Запрос;
Запрос.Текст =
        ""ВЫБРАТЬ
        | НачисленияБазаНачисления.РезультатБаза КАК База,
        | НачисленияБазаНачисления.НомерСтроки КАК НомерСтроки
        |ИЗ
        | РегистрРасчета.Начисления.БазаНачисления(&ИзмерениеОсновного,
        | &ИзмерениеБазового, , Регистратор = &Регистратор И ВидРасчета = &ВидРасчета И
        | Сотрудник В (&СписокСотрудников)) КАК НачисленияБазаНачисления"";

Измер = Новый Массив(1);
Измер[0] = ""Сотрудник"";

Запрос.УстановитьПараметр(""ИзмерениеОсновного"", Измер);
Запрос.УстановитьПараметр(""ИзмерениеБазового"", Измер);
Запрос.УстановитьПараметр(""Регистратор"", Регистратор);
Запрос.УстановитьПараметр(""ВидРасчета"", ТребуемыйВидРасчета);
Запрос.УстановитьПараметр(""СписокСотрудников"", СписокСотрудников);

ВыборкаРезультата = Запрос.Выполнить().Выбрать(); "}
},
{0,
{"18.7 Добавление обхода набора записей регистра и вычисления результата вторичных
записей",0,0,"18.7","Для Каждого ЗаписьРегистра Из НаборЗаписейРегистра Цикл
        СтруктураНомер = Новый Структура(""НомерСтроки"");
        СтруктураНомер.НомерСтроки = ЗаписьРегистра.НомерСтроки;
        ВыборкаРезультата.Сбросить();
```

```
        Если ВыборкаРезультата.НайтиСледующий(СтруктураНомер)  Тогда
                ЗаписьРегистра.Результат = ВыборкаРезультата.База * (10 / 100);
                Сообщение = Новый СообщениеПользователю;
                Сообщение.Текст = ""Выполнен расчет"" + ЗаписьРегистра.Регистратор + "" –
""  + ЗаписьРегистра.ВидРасчета + "" – "" + ЗаписьРегистра.Сотрудник;
                Сообщение.Сообщить();
        КонецЕсли;
КонецЦикла;

"}
},
{0,
{"18.8 Обработчик команды
Перерассчитать",0,0,"18.8","ПроведениеРасчетов.ПерерассчитатьНачисления(Предопределен
ноеЗначение(""ПланВидовРасчета.ОсновныеНачисления.Оклад""));
ПроведениеРасчетов.ПерерассчитатьНачисления(ПредопределенноеЗначение(""ПланВидовР
асчета.ОсновныеНачисления.Премия""));


"}
},
{0,
{"18.9 Процедура перерасчета начислений",0,0,"18.9","Процедура
ПерерассчитатьНачисления(ТребуемыйВидРасчета) Экспорт

        // Здесь следует выбрать из набора записей перерасчета записи в следующей
последовательности:
        // записи документа1 для сотрудников из списка,
        // записи документа2 для сотрудников из списка, и т. д.
        Запрос = Новый Запрос(
        ""ВЫБРАТЬ
        | НачисленияПерерасчет.ОбъектПерерасчета,
        | НачисленияПерерасчет.Сотрудник
        |ИЗ
        | РегистрРасчета.Начисления.Перерасчет КАК НачисленияПерерасчет
        |
        |ГДЕ
        | НачисленияПерерасчет.ВидРасчета = &ТребуемыйВидРасчета
        |
        |ИТОГИ ПО
        | НачисленияПерерасчет.ОбъектПерерасчета"");

        Запрос.УстановитьПараметр(""ТребуемыйВидРасчета"", ТребуемыйВидРасчета);
        СписокСотрудников = Новый СписокЗначений;

        // Перебрать группировку по регистратору.
        ВыборкаПоРегистратору =
Запрос.Выполнить().Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);
        Пока ВыборкаПоРегистратору.Следующий() Цикл
                Регистратор = ВыборкаПоРегистратору.ОбъектПерерасчета;

                // Перебрать группировку по сотрудникам для выбранного регистратора
                // и создать список сотрудников.
                ВыборкаПоСотрудникам = ВыборкаПоРегистратору.Выбрать();
                СписокСотрудников.Очистить();

                Пока ВыборкаПоСотрудникам.Следующий() Цикл
```

```
                    СписокСотрудников.Добавить(ВыборкаПоСотрудникам.Сотрудник);
                КонецЦикла;

                // Получить набор записей регистра расчета для выбранного регистратора.
                НаборЗаписей = РегистрыРасчета.Начисления.СоздатьНаборЗаписей();
                НаборЗаписей.Отбор.Регистратор.Значение = Регистратор;
                НаборЗаписей.Прочитать();

                РассчитатьНачисления(НаборЗаписей, ТребуемыйВидРасчета,
СписокСотрудников);
                НаборЗаписей.Записать( , Истина);

                // Очистить перерассчитанные записи в перерасчете.
                НаборЗаписейПерерасчета =
РегистрыРасчета.Начисления.Перерасчеты.Перерасчет.СоздатьНаборЗаписей();
                НаборЗаписейПерерасчета.Отбор.ОбъектПерерасчета.Значение = Регистратор;
                НаборЗаписейПерерасчета.Записать();
        КонецЦикла;

КонецПроцедуры
"}
},
{0,
{"18.10 Обработчик команды
Сформировать",0,0,"18.10","СформироватьНаСервере(ДиаграммаГанта);"}
},
{0,
{"18.11 Шаблон процедуры
СформироватьНаСервере()",0,0,"18.11","&НаСервереБезКонтекста
Процедура СформироватьНаСервере(Диаграмма)

        Запрос = Новый Запрос;
        Запрос.Текст = ;

КонецПроцедуры
"}
},
{0,
{"18.12 Изменение процедуры СформироватьНаСервере()",0,0,"18.12","ВыборкаРезультата =
Запрос.Выполнить().Выбрать();

// Запретить обновление диаграммы
Диаграмма.Обновление = Ложь;

Диаграмма.Очистить();
Диаграмма.ОтображатьЗаголовок = Ложь;

// Заполнить диаграмму
Пока ВыборкаРезультата.Следующий() цикл
        // Получить серию, точку и значение для них
        ТекущаяСерия = Диаграмма.УстановитьСерию(ВыборкаРезультата.ВидРасчета);
        ТекущаяТочка = Диаграмма.УстановитьТочку(ВыборкаРезультата.Сотрудник);
        ТекущееЗначение = Диаграмма.ПолучитьЗначение(ТекущаяТочка, ТекущаяСерия);

        // Создать нужные нам интервалы в значении
        ТекущийИнтервал = ТекущееЗначение.Добавить();
        ТекущийИнтервал.Начало = ВыборкаРезультата.ПериодДействияНачало;
```

```
        ТекущийИнтервал.Конец = ВыборкаРезультата.ПериодДействияКонец;
        ТекущийИнтервал.Текст = ВыборкаРезультата.РегистраторПредставление;
        ТекущийИнтервал.Расшифровка = ВыборкаРезультата.Регистратор;
КонецЦикла;

// Раскрасить серии своими цветами
Для Каждого Серия из Диаграмма.Серии Цикл
        Если Серия.Значение = ПланыВидовРасчета.ОсновныеНачисления.Оклад Тогда
                Серия.Цвет = WEBЦвета.Желтый;
        ИначеЕсли Серия.Значение = ПланыВидовРасчета.ОсновныеНачисления.Премия
Тогда
                Серия.Цвет = WEBЦвета.Зеленый;
        ИначеЕсли Серия.Значение = ПланыВидовРасчета.ОсновныеНачисления.Невыход
Тогда
                Серия.Цвет = WEBЦвета.Красный;
        КонецЕсли;
КонецЦикла;

// Разрешить обновление диаграммы
Диаграмма.Обновление = Истина;
"}
}
},
{4,
{"Занятие 19. Поиск в базе даных",1,0,"","""},
{0,
{"19.1 Обработчики событий нажатия на кнопки Поиск, ПредыдущаяПорция,
СледующаяПорция",0,0,"19.1","&НаКлиенте
Процедура Поиск()
        Искать(0);
КонецПроцедуры

&НаКлиенте
Процедура ПредыдущаяПорция()
        Искать(-1);
КонецПроцедуры

&НаКлиенте
Процедура СледующаяПорция()
        Искать(1);
КонецПроцедуры
"}
},
{0,
{"19.2 Процедура поиска на клиенте",0,0,"19.2","&НаКлиенте
// Процедура поиска, получение и отображение результата
Процедура Искать(Направление)
        Если ПустаяСтрока(ПоисковоеВыражение) Тогда
                Предупреждение("""Не задана строка поиска.""");
                Возврат;
        КонецЕсли;

        ИскатьСервер(Направление);
КонецПроцедуры
"}
},
{0,
```

```
{"19.3 Процедура поиска на сервере",0,0,"19.3","&НаСервере
Процедура ИскатьСервер(Направление) Экспорт
	СписокПоиска = ПолнотекстовыйПоиск.СоздатьСписок();
	СписокПоиска.СтрокаПоиска = ПоисковоеВыражение;

	Если Направление = 0 Тогда
		СписокПоиска.ПервыйЧасть();
	ИначеЕсли Направление = -1 Тогда
		СписокПоиска.ПредыдущаяЧасть(ТекущаяПозиция);
	ИначеЕсли Направление = 1 Тогда
		СписокПоиска.СледующаяЧасть(ТекущаяПозиция);
	КонецЕсли;

	РезультатыПоиска.Очистить();
	Для Каждого Результат Из СписокПоиска Цикл
		РезультатыПоиска.Добавить(Результат.Значение);
	КонецЦикла;

	РезультатПоиска =
СписокПоиска.ПолучитьОтображение(ВидОтображенияПолнотекстовогоПоиска.HTMLТекст
);
	ТекущаяПозиция = СписокПоиска.НачальнаяПозиция();
	ПолноеКоличество = СписокПоиска.ПолноеКоличество();

	Если СписокПоиска.Количество() <> 0 Тогда
		СообщениеОРезультате = ""Показаны "" +   Строка(ТекущаяПозиция + 1) + "" -
"" +
			Строка(ТекущаяПозиция + СписокПоиска.Количество()) + "" из "" +
Строка(ПолноеКоличество);
		Элементы.СледующаяПорция.Доступность = (ПолноеКоличество -
ТекущаяПозиция) > СписокПоиска.Количество();
		Элементы.ПредыдущаяПорция.Доступность = (ТекущаяПозиция > 0);
	Иначе
		СообщениеОРезультате = ""Не найдено"";
		Элементы.СледующаяПорция.Доступность = Ложь;
		Элементы.ПредыдущаяПорция.Доступность = Ложь;
	КонецЕсли;
КонецПроцедуры
"}
},
{0,
{"19.4 Обработчик события ПриНажатии() поля РезультатПоиска",0,0,"19.4","ЭлементHTML
= ДанныеСобытия.Event.srcElement;
Если (ЭлементHTML.id = ""FullTextSearchListItem"") Тогда

	// Получить имя файла (номер строки списка поиска), содержащегося в гиперссылке
	НомерВСписке = Число(ЭлементHTML.nameProp);

	// Получить строку списка поиска по номеру
	ВыбраннаяСтрока = РезультатыПоиска[НомерВСписке].Значение;

	// Открыть форму найденного объекта
	ОткрытьЗначение(ВыбраннаяСтрока);
	СтандартнаяОбработка = Ложь;
КонецЕсли;
"}
}
```

```
},
{5,
{"Занятие 20. Выполнение заданий по расписанию",1,0,"",""},
{0,
{"20.1 Процедура ОбновлениеИндекса()",0,0,"20.1","Если
ПолнотекстовыйПоиск.ПолучитьРежимПолнотекстовогоПоиска() =
РежимПолнотекстовогоПоиска.Разрешить Тогда
        Если Не ПолнотекстовыйПоиск.ИндексАктуален() Тогда
                ПолнотекстовыйПоиск.ОбновитьИндекс( , Истина);
        КонецЕсли;
КонецЕсли;
"}
},
{0,
{"20.2 Процедура СлияниеИндексов()",0,0,"20.2","Если
ПолнотекстовыйПоиск.ПолучитьРежимПолнотекстовогоПоиска() =
РежимПолнотекстовогоПоиска.Разрешить Тогда
        Если Не ПолнотекстовыйПоиск.ИндексАктуален() Тогда
                ПолнотекстовыйПоиск.ОбновитьИндекс(Истина);
        КонецЕсли;
КонецЕсли;
"}
},
{0,
{"20.3  Обработчик события формы обработки ПланировщикЗаданий
ПриОткрытии()",0,0,"20.3","#Если ТолстыйКлиентУправляемоеПриложение Тогда
        ПодключитьОбработчикОжидания(""ОбработкаЗаданий"", 3);
#Иначе
        Предупреждение(""Обработка может быть запущена только в толстом клиенте!"");
        Закрыть();
#КонецЕсли    "}
},
{0,
{"20.4 Обработчик ожидания",0,0,"20.4","&НаКлиенте
Процедура ОбработкаЗаданий()

        #Если ТолстыйКлиентУправляемоеПриложение Тогда
                ВыполнитьОбработкуЗаданий();
        #КонецЕсли

КонецПроцедуры

"}
},
{0,
{"20.5 Сообщение о запуске регламентного задания",0,0,"20.5","Сообщение = Новый
СообщениеПользователю;
Сообщение.Текст = ""Запуск регламентного задания Обновление индекса "" +
ТекущаяДата();
Сообщение.Сообщить();
"}
}
},
{2,
{"Занятие 21. Создание документа ввода начальных остатков",1,0,"",""},
{0,
{"21.1 Обработчик события ПередЗаписью формы документа",0,0,"21.1","Для Каждого
```

```
ЗаписьРегистра Из Объект.Движения.ОстаткиМатериалов Цикл
        ЗаписьРегистра.Период = Объект.Дата;
КонецЦикла;
"}
},
{0,
{"21.2 Обработчик события ПередЗаписью модуля  объекта",0,0,"21.2","Процедура
ПередЗаписью(Отказ, РежимЗаписи, РежимПроведения)

        // Определить, нужно ли обновлять дату в движениях
        ОбновитьДатуДвижений = ЭтоНовый() Или
Движения.ОстаткиМатериалов.Модифицированность();
        Если Не ОбновитьДатуДвижений Тогда
                // Проверить, что дата изменилась
                Запрос = Новый Запрос;
                Запрос.УстановитьПараметр(""ТекущийДокумент"", Ссылка);
                Запрос.Текст =
                ""ВЫБРАТЬ
                |Дата
                |ИЗ
                |Документ.ВводНачальныхОстатковНоменклатуры
                |ГДЕ Ссылка = &ТекущийДокумент"";

                Выборка = Запрос.Выполнить().Выбрать();
                Выборка.Следующий();
                ОбновитьДатуДвижений = Выборка.Дата <> Дата;
        КонецЕсли;

        // Установить всем новую дату, если нужно
        Если ОбновитьДатуДвижений Тогда
                Если Не Движения.ОстаткиМатериалов.Выбран() И
                Не Движения.ОстаткиМатериалов.Модифицированность() Тогда
                        Движения.ОстаткиМатериалов.Прочитать();
                КонецЕсли;
                Для Каждого ЗаписьРегистра Из Движения.ОстаткиМатериалов Цикл
                        ЗаписьРегистра.Период = Дата;
                КонецЦикла;
        КонецЕсли;

КонецПроцедуры
"}
}
},
{4,
{"Занятие 22. Список пользователей и их роли",1,0,"",""},
{0,
{"22.1 Ограничение доступа к
данным",0,0,"22.1","НачисленияСотрудникам.Начисления.ВидРасчета <>
ЗНАЧЕНИЕ(ПланВидовРасчета.ОсновныеНачисления.Премия)"}
},
{0,
{"22.2 Ограничение доступа к данным",0,0,"22.2","ВЫБРАТЬ
        1
 ИЗ
        Документ.НачисленияСотрудникам.Начисления
 ГДЕ
         Документ.НачисленияСотрудникам.Начисления.ВидРасчета =
```

```
ЗНАЧЕНИЕ(ПланВидовРасчета.ОсновныеНачисления.Премия)
        И Документ.НачисленияСотрудникам.Начисления.Ссылка = #Параметр(1).Ссылка
"}
},
{0,
{"22.4 Ограничение доступа к данным",0,0,"22.4","ДокНачисления ГДЕ НЕ 1 В
(#ЕстьПремия(""ДокНачисления""))"}
},
{0,
{"22.5 Ограничение доступа к данным",0,0,"22.5","ДокНачисления ГДЕ НЕ 1 В (
        ВЫБРАТЬ
                1
         ИЗ
                Документ.НачисленияСотрудникам.Начисления
        ГДЕ
                Документ.НачисленияСотрудникам.Начисления.ВидРасчета =
ЗНАЧЕНИЕ(ПланВидовРасчета.ОсновныеНачисления.Премия)
                И Документ.НачисленияСотрудникам.Начисления.Ссылка =
ДокНачисления.Ссылка)
"}
}
},
{36,
{"Занятие 24. Обмен данными",1,0,"",""},
{0,
{"24.1 Функция формирования префикса номера",0,0,"24.1","Функция
ПолучитьПрефиксНомера() Экспорт

        Возврат Константы.ПрефиксНумерации.Получить();

КонецФункции
"}
},
{0,
{"24.2 Обработчик события ПриУстановкеНовогоКода",0,0,"24.2","Процедура
ПриУстановкеНовогоКода(СтандартнаяОбработка, Префикс)

        Префикс = Обмен.ПолучитьПрефиксНомера();

КонецПроцедуры
"}
},
{0,
{"24.3 Обработчик события ПриУстановкеНовогоНомера",0,0,"24.3","Процедура
ПриУстановкеНовогоНомера(СтандартнаяОбработка, Префикс)

        Префикс = Обмен.ПолучитьПрефиксНомера();

КонецПроцедуры
"}
},
{0,
{"24.4 Обработчик события формы узла плана обмена
ПриСозданииНаСервере",0,0,"24.4","Если Объект.Ссылка =
ПланыОбмена.Филиалы.ЭтотУзел() Тогда
        Элементы.Главный.Доступность = Ложь;
КонецЕсли;
```

```
"}
},
{0,
{"24.5 Обработчик выполнения команды
ЗарегистрироватьИзменения",0,0,"24.5","РегистрацияИзмененийНаСервере(Элементы.Списо
к.ТекущаяСтрока);"}
},
{0,
{"24.6 Процедура РегистрацияИзмененийНаСервере",0,0,"24.6","&НаСервереБезКонтекста
Процедура РегистрацияИзмененийНаСервере(Узел)

	// Регистрация изменений всех данных для узла
	ПланыОбмена.ЗарегистрироватьИзменения(Узел);

КонецПроцедуры
"}
},
{0,
{"24.7 Функция ПредопределенныйУзел()",0,0,"24.7","&НаСервереБезКонтекста
Функция ПредопределенныйУзел(Узел)

	Возврат Узел = ПланыОбмена.Филиалы.ЭтотУзел();

КонецФункции
"}
},
{0,
{"24.8 Обработчик события СписокПриАктивизацииСтроки() элемента формы
Список",0,0,"24.8","Если ПредопределенныйУзел(Элемент.ТекущаяСтрока) Тогда
	Элементы.ЗарегистрироватьИзменения.Доступность = Ложь;
Иначе
	Элементы.ЗарегистрироватьИзменения.Доступность = Истина;
КонецЕсли;
"}
},
{0,
{"24.9 Обработчик команды ВыполнитьОбмен",0,0,"24.9","ОбменСФилиалами();"}
},
{0,
{"24.10 Создание процедуры ОбменСФилиалами",0,0,"24.10","&НаСервереБезКонтекста
Процедура ОбменСФилиалами() Экспорт

	ВыборкаУзлов = ПланыОбмена.Филиалы.Выбрать();

	Пока ВыборкаУзлов.Следующий() Цикл

		// Произвести обмен данными со всеми узлами, кроме текущего (ЭтотУзел)
		Если ВыборкаУзлов.Ссылка <> ПланыОбмена.Филиалы.ЭтотУзел() Тогда
			УзелОбъект = ВыборкаУзлов.ПолучитьОбъект();

			// Получить сообщение
			УзелОбъект.ПрочитатьСообщениеСИзменениями();

			// Сформировать сообщение
			УзелОбъект.ЗаписатьСообщениеСИзменениями();

		КонецЕсли;
```

```
        КонецЦикла;

КонецПроцедуры
"}
},
{0,
{"24.11 Формирование имени файла в процедуре записи данных",0,0,"24.11","Процедура
ЗаписатьСообщениеСИзменениями() Экспорт

        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = ""-------- Выгрузка в узел "" + Строка(ЭтотОбъект) + "" ----------"";
        Сообщение.Сообщить();
        Каталог = КаталогВременныхФайлов();

        // Сформировать имя временного файла
        ИмяФайла = Каталог + ?(Прав(Каталог, 1) = ""\"","""", ""\"") + ""Message"" +
СокрЛП(ПланыОбмена.
                        Филиалы.ЭтотУзел().Код) + ""_"" + СокрЛП(Ссылка.Код) + "".xml"";
        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = ""-------- Конец выгрузки ----------"";
        Сообщение.Сообщить();

КонецПроцедуры
"}
},
{0,
{"24.12. Создание объекта записи XML в процедуре записи данных",0,0,"24.12","     // Создать
объект записи XML
        // *** ЗаписьXML-документов
        ЗаписьXML = Новый ЗаписьXML;
        ЗаписьXML.ОткрытьФайл(ИмяФайла);
        ЗаписьXML.ЗаписатьОбъявлениеXML();
        ЗаписьXML.Закрыть();

"}
},
{0,
{"24.13 Создание очередного номера сообщения и запись заголовка сообщения в
XML",0,0,"24.13","// *** Инфраструктура сообщений
        ЗаписьСообщения = ПланыОбмена.СоздатьЗаписьСообщения();
        ЗаписьСообщения.НачатьЗапись(ЗаписьXML, Ссылка);
        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = "" Номер сообщения: "" + ЗаписьСообщения.НомерСообщения;
        Сообщение.Сообщить();
        ЗаписьСообщения.ЗакончитьЗапись();
"}
},
{0,
{"24.14 Получение выборки из записей регистрации изменений, предназначенных данному
узлу",0,0,"24.14","  // Получить выборку измененных данных
        // *** Механизм регистрации изменений
        ВыборкаИзменений =
ПланыОбмена.ВыбратьИзменения(ЗаписьСообщения.Получатель,ЗаписьСообщения.НомерС
ообщения);
"}
},
{0,
```

```
{"24.15 Перебор выборки записей и сериализация их в открытый XML-файл",0,0,"24.15","
        Пока ВыборкаИзменений.Следующий() Цикл
                // Записать данные в сообщение *** XML-сериализация
                ЗаписатьXML(ЗаписьXML, ВыборкаИзменений.Получить());
        КонецЦикла;
"}
},
{0,
{"24.16 Формирование имени файла, содержащего данные обмена",0,0,"24.16","Процедура
ПрочитатьСообщениеСИзменениями() Экспорт

        Каталог = КаталогВременныхФайлов();

        // Сформировать имя файла
        ИмяФайла = Каталог + ?(Прав(Каталог, 1) = """\"", """""", """\"") +
         """Message"" + СокрЛП(Ссылка.Код) + ""_"" +
СокрЛП(ПланыОбмена.Филиалы.ЭтотУзел().Код) + "".xml"";
        Файл = Новый Файл(ИмяФайла);
        Если Не Файл.Существует() Тогда
                Возврат;
        КонецЕсли;

        УдалитьФайлы(ИмяФайла);
        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = ""-------- Конец загрузки ----------"";
        Сообщение.Сообщить();

КонецПроцедуры
"}
},
{0,
{"24.17.Добавление чтения найденного файла с данными обмена",0,0,"24.17","        // ***
Чтение документов XML
        // Попытаться открыть файл
        ЧтениеXML = Новый ЧтениеXML;
        Попытка
                ЧтениеXML.ОткрытьФайл(ИмяФайла);
        Исключение
                Сообщение = Новый СообщениеПользователю;
                Сообщение.Текст = ""Невозможно открыть файл обмена данными."";
                Сообщение.Сообщить();
      Возврат;
        КонецПопытки;
        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = ""-------- Загрузка из "" + Строка(ЭтотОбъект) + "" ----------"";
        Сообщение.Сообщить();
        Сообщение = Новый СообщениеПользователю;
        Сообщение.Текст = "" – Считывается файл "" + ИмяФайла;
        Сообщение.Сообщить();

        ЧтениеXML.Закрыть();
"}
},
{0,
{"24.18 Добавление чтения заголовка XML-сообщения",0,0,"24.18","    // Загрузить из
найденного файла
        // *** Инфраструктура сообщений
```

```
        ЧтениеСообщения = ПланыОбмена.СоздатьЧтениеСообщения();

        // Читать заголовок сообщения обмена данными – файла XML
        ЧтениеСообщения.НачатьЧтение(ЧтениеXML);

        ЧтениеСообщения.ЗакончитьЧтение();
"}
},
{0,
{"24.19 Добавление проверки сообщения",0,0,"24.19","      // Сообщение предназначено не для этого узла
        Если ЧтениеСообщения.Отправитель <> Ссылка Тогда ВызватьИсключение ""Неверный узел"";
        КонецЕсли;
"}
},
{0,
{"24.20 Удаление записей регистрации изменений для узла отправителя",0,0,"24.20","       // Удаляем регистрацию изменений для узла отправителя сообщения.
        // *** Служба регистрации изменений
        ПланыОбмена.УдалитьРегистрациюИзменений(ЧтениеСообщения.Отправитель,ЧтениеСообщения.НомерПринятого);
"}
},
{0,
{"24.21 Чтение данных из сообщения",0,0,"24.21"," // Читаем данные из сообщения *** XML-сериализация
        Пока ВозможностьЧтенияXML(ЧтениеXML) Цикл

        КонецЦикла;
"}
},
{0,
{"24.22 Представление данных XML в виде значения, имеющего тип",0,0,"24.22","   // Читаем очередное значение
        Данные = ПрочитатьXML(ЧтениеXML);
"}
},
{0,
{"24.23 Разрешение возможных коллизий",0,0,"24.23","              // Не переносим изменение, полученное в главный из неглавного, если есть регистрация изменения
                Если Не ЧтениеСообщения.Отправитель.Главный И

        ПланыОбмена.ИзменениеЗарегистрировано(ЧтениеСообщения.Отправитель, Данные) Тогда
                        Сообщение = Новый СообщениеПользователю;
                        Сообщение.Текст = "" – Изменения отклонены"";
                        Сообщение.Сообщить();
        Продолжить;
                КонецЕсли;
"}
},
{0,
{"24.24 Запись полученных данных",0,0,"24.24","             // Записать полученные данные
                Данные.ОбменДанными.Отправитель = ЧтениеСообщения.Отправитель;
                Данные.ОбменДанными.Загрузка = Истина;
                Данные.Записать();
```

```
"}
},
{0,
{"24.25 Функция ПредопределенныйУзел()",0,0,"24.25","&НаСервереБезКонтекста
Функция ПредопределенныйУзел(Узел)

	Возврат Узел = ПланыОбмена.Отделения.ЭтотУзел();

КонецФункции
"}
},
{0,
{"24.26 Процедура ПолеВводаОтделениеОбработкаВыбора()",0,0,"24.26","	Если
ПредопределенныйУзел(ВыбранноеЗначение)Тогда
		Элементы.СоздатьНачальныйОбраз.Доступность = Ложь;
	Иначе
		Элементы.СоздатьНачальныйОбраз.Доступность = Истина;
	КонецЕсли;
"}
},
{0,
{"24.27 Обработчик нажатия кнопки Создать начальный образ",0,0,"24.27","Диалог = Новый
ДиалогВыбораФайла(РежимДиалогаВыбораФайла.ВыборКаталога);
	Диалог.Заголовок = ""Укажите каталог информационной базы:"";
	Если Диалог.Выбрать() Тогда
		СоздатьНачальныйОбразНаСервере(ПолеВводаОтделение, Диалог.Каталог);
		Предупреждение(""Создание начального образа узла завершено."");
	КонецЕсли;

"}
},
{0,
{"24.28 Процедура
СоздатьНачальныйОбразНаСервере",0,0,"24.28","&НаСервереБезКонтекста
Процедура СоздатьНачальныйОбразНаСервере(Узел, КаталогСоединения)

	ПланыОбмена.СоздатьНачальныйОбраз(Узел, ""File ="" + КаталогСоединения);

КонецПроцедуры
"}
},
{0,
{"24.29 Обработчик нажатия кнопки Записать изменения",0,0,"24.29","	Диалог = Новый
ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Сохранение);
	Диалог.Заголовок = ""Укажите файл обмена:"";
	Если Диалог.Выбрать() Тогда
		ЗаписатьИзмененияНаСервере(ПолеВводаОтделение, Диалог.ПолноеИмяФайла);
		Предупреждение(""Запись изменений завершена."");
	КонецЕсли;
"}
},
{0,
{"24.30 Процедура ЗаписатьИзмененияНаСервере",0,0,"24.30","&НаСервереБезКонтекста
Процедура ЗаписатьИзмененияНаСервере(Узел, ИмяФайла)

	// Создать и проинициализировать объект ЗаписьXML
	ЗаписьXML = Новый ЗаписьXML;
```

```
        ЗаписьXML.ОткрытьФайл(ИмяФайла);

        // Создать объект ЗаписьСообщенияОбмена и начать запись сообщения
        ЗаписьСообщения = ПланыОбмена.СоздатьЗаписьСообщения();
        ЗаписьСообщения.НачатьЗапись(ЗаписьXML, Узел);

        // Записать содержимое тела сообщения обмена данными распределенной ИБ
        ПланыОбмена.ЗаписатьИзменения(ЗаписьСообщения);

        // Закончить запись сообщения и запись XML
        ЗаписьСообщения.ЗакончитьЗапись();
        ЗаписьXML.Закрыть();

КонецПроцедуры
"}
},
{0,
{"24.31 Обработчик нажатия кнопки Прочитать изменения",0,0,"24.31","      Диалог = Новый
ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);
        Диалог.Заголовок = ""Укажите файл обмена:"";
        Если Диалог.Выбрать() Тогда
                ПрочитатьИзмененияНаСервере(Диалог.ПолноеИмяФайла);
                Предупреждение(""Чтение изменений завершено."");
        КонецЕсли;
"}
},
{0,
{"24.32 Процедура ПрочитатьИзмененияНаСервере",0,0,"24.32","&НаСервереБезКонтекста
Процедура ПрочитатьИзмененияНаСервере(ИмяФайла)

        // Создать и проинициализировать объект ЧтениеXML
        ЧтениеXML = Новый ЧтениеXML;
        ЧтениеXML.ОткрытьФайл(ИмяФайла);

        // Создать объект ЧтениеСообщенияОбмена и начать чтение сообщения
        ЧтениеСообщения = ПланыОбмена.СоздатьЧтениеСообщения();
        ЧтениеСообщения.НачатьЧтение(ЧтениеXML);

        // Прочитать содержимое тела сообщения
        ПланыОбмена.ПрочитатьИзменения(ЧтениеСообщения);

        // Закончить чтение сообщения и чтение XML
        ЧтениеСообщения.ЗакончитьЧтение();
        ЧтениеXML.Закрыть();

КонецПроцедуры
"}
},
{0,
{"24.33 Просмотр работы событий объекта ПланОбменаОбъект",0,0,"24.33","Процедура
ПриОтправкеДанныхГлавному(ЭлементДанных, ОтправкаЭлемента)

  Сообщить(""ПриОтправкеДанныхГлавному ""+ ЭлементДанных);

КонецПроцедуры

Процедура ПриОтправкеДанныхПодчиненному(ЭлементДанных, ОтправкаЭлемента)
```

```
  Сообщить(""ПриОтправкеДанныхПодчиненному ""+ ЭлементДанных);

КонецПроцедуры

Процедура ПриПолученииДанныхОтГлавного(ЭлементДанных, ПолучениеЭлемента,
ОтправкаНазад)

  Сообщить(""ПриПолученииДанныхОтГлавного ""+ ЭлементДанных);

КонецПроцедуры

Процедура ПриПолученииДанныхОтПодчиненного(ЭлементДанных, ПолучениеЭлемента,
ОтправкаНазад)

  Сообщить(""ПриПолученииДанныхОтПодчиненного ""+ ЭлементДанных);

КонецПроцедуры
"}
},
{0,
{"24.34 Перемещение Узла2 в корень дерева",0,0,"24.34","// В информационной базе Узла2.
ПланыОбменаМенеджер.УстановитьГлавныйУзел(Неопределено);

// В информационной базе Узла1.
ПланыОбменаМенеджер.УстановитьГлавныйУзел(Узел2);
"}
},
{0,
{"24.35 Отключение поддерева от дерева",0,0,"24.35","// В информационной базе Узла1.
ПланыОбменаМенеджер.УстановитьГлавныйУзел(Неопределено);
"}
},
{0,
{"24.36 Создание распределенной информационной базы из баз с идентичной
конфигурацией",0,0,"24.36","// В информационных базах Узла2, Узла3 и Узла4.
ПланыОбменаМенеджер.УстановитьГлавныйУзел(Узел1);
"}
}
},
{1,
{"Занятие 25. Функциональные опции",1,0,"",""},
{0,
{"25.1 Обработчик события ПослеЗаписи формы
констант",0,0,"25.1","ОбновитьИнтерфейс();"}
}
},
{6,
{"Занятие 26. Общие приемы разработки",1,0,"",""},
{0,
{"26.1 Обработчик нажатия кнопки
Подбор",0,0,"26.1","ОткрытьФорму(""Справочник.Номенклатура.ФормаВыбора"", ,
Элементы.Материалы);


"}
},
```

```
{0,
{"26.2 Обработчик события ОбработкаВыбора таблицы Материалы",0,0,"26.2","НоваяСтрока
= Объект.Материалы.Добавить();
НоваяСтрока.Материал = ВыбранноеЗначение;
"}
},
{0,
{"26.3 Обработчик нажатия кнопки Подбор",0,0,"26.3","ПараметрыФормы = Новый
Структура(""ЗакрыватьПриВыборе"", Ложь);
ОткрытьФорму(""Справочник.Номенклатура.ФормаВыбора"", ПараметрыФормы,
Элементы.Материалы);


"}
},
{0,
{"26.4 Обработчик нажатия кнопки Подбор",0,0,"26.4","   ПараметрыФормы = Новый
Структура(""МножественныйВыбор"", Истина);
        ОткрытьФорму(""Справочник.Номенклатура.ФормаВыбора"", ПараметрыФормы,
Элементы.Материалы);
"}
},
{0,
{"26.5 Обработчик события ОбработкаВыбора таблицы Материалы",0,0,"26.5","     Для
Каждого ВыбранныйЭлемент Из ВыбранноеЗначение Цикл
      НоваяСтрока = Объект.Материалы.Добавить();
                НоваяСтрока.Материал = ВыбранныйЭлемент;
         КонецЦикла;
         "}
},
{0,
{"26.6 Обработчик нажатия кнопки Подбор",0,0,"26.6","   ПараметрыФормы = Новый
Структура(""ЗакрыватьПриВыборе, МножественныйВыбор"", Ложь, Истина);
        ОткрытьФорму(""Справочник.Номенклатура.ФормаВыбора"", ПараметрыФормы,
Элементы.Материалы);
        "}
}
},
{4,
{"Занятие 27. Формы",1,0,"",""},
{0,
{"27.3 Обработчик события формы ПриСозданииНаСервере","0,0,"27.3","
СписокСправочника = Элементы.Список;

        // Задать режим отображения справочника
        СписокСправочника.Отображение = ОтображениеТаблицы.Список;

        // Скрыть линии сетки
        СписокСправочника.ВертикальныеЛинии = Ложь;
        СписокСправочника.ГоризонтальныеЛинии = Ложь;

"}
},
{0,
{"27.4 Обработчик события ОбработкаПроверкиЗаполнения",0,0,"27.4","Процедура
ОбработкаПроверкиЗаполнения(Отказ, ПроверяемыеРеквизиты)
```

```
        ПроверяемыеРеквизиты.Удалить(ПроверяемыеРеквизиты.Найти(""ПеречеьНоменклатуры.НаборСвойств""));
        Индекс = 0;

        Для Каждого ТекСтрокаПеречеьНоменклатуры Из ПеречеьНоменклатуры Цикл
            Индекс = Индекс + 1;

            Если ТекСтрокаПеречеьНоменклатуры.Номенклатура.ВидНоменклатуры = Перечисления.ВидыНоменклатуры.Материал Тогда

                Если Не ЗначениеЗаполнено(ТекСтрокаПеречеьНоменклатуры.НаборСвойств) Тогда
                    Сообщение = Новый СообщениеПользователю();
                    Сообщение.Текст = ""В строке "" + Индекс + "" списка Перечень номенклатуры не заполнена колонка Набор свойств"";
                    Сообщение.Поле =  ""ПеречеьНоменклатуры["" + Строка(Индекс - 1) + ""].НаборСвойств"";
                    Сообщение.УстановитьДанные(ЭтотОбъект);
                    Сообщение.Сообщить();
                    Отказ = Истина;
                КонецЕсли;

            КонецЕсли;

        КонецЦикла;

КонецПроцедуры
"}
},
{0,
{"27.5 Добавление реквизита в массив проверяемых реквизитов",0,0,"27.5","ПроверяемыеРеквизиты.Добавить(""ПеречеьНоменклатуры.НаборСвойств"");"}
},
{0,
{"27.6 Обработчик выполнения команды",0,0,"27.6","ПараметрыФормы = Новый Структура(""Отбор,КлючНазначенияИспользования,СформироватьПриОткрытии"", Новый Структура(""Склад"", ПараметрКоманды), ""ОстаткиПоСкладу"", Истина);
ОткрытьФорму(""Отчет.Материалы.Форма"", ПараметрыФормы, ПараметрыВыполненияКоманды.Источник, ПараметрыВыполненияКоманды.Уникальность, ПараметрыВыполненияКоманды.Окно);
        "}
}
}
}
}
```