ТУРБО БУХГАЛТЕР 6

Справочник по программе

Стандартные процедуры и функции

Москва
2007

Турбо Бухгалтер 6: Справочник по программе. Стандартные процедуры и функции. М.:ДИЦ, 2007. — 304 с.

Разработчиком и генеральным распространителем программы Турбо Бухгалтер является ЗАО "ДИЦ" (Долгопрудненский исследовательский центр).

Адрес: 125057, Москва, Чапаевский пер., д. 6, стр. 1

Телефоны для справок: **(499) 157-08-20, 157-04-72, 956-12-50**

Телефоны для консультаций зарегистрированным пользователям:
(499) 157-03-15, 157-03-64

Факс: **(495) 913-2041**

E-Mail: **tb@dic.ru** (для писем), **hotline@dic.ru** (для консультаций)

Web: **http://www.dic.ru/**

Издание 13

Автор Руководства: Г.Тарасова, М.Самохина

Редактор: А.Костин

Разработчики программы: С.Алексеев, Б.Бабань, К.Бутусов, В.Ворончихин, Г.Гогин, М.Егоров, Ю.Загоруй, А.Медведев, Н.Миняшкин, Д.Надежин, М.Русов, Д.Серов, С.Сущик, Д.Умнов, А.Чайковский, В.Шевяков

Разработчик Справочной системы программы: М.Спасская

Компьютерная верстка: М.Самохина

Введение

Программа Турбо Бухгалтер 6 имеет большую библиотеку стандартных процедур и функций, используемых в журналах, типовых операциях, бланках, картотеках, калькуляторе.

Справочник стандартных процедур и функций организован следующим образом.

Сначала приводится классификация процедур и функций по функциональному назначению.

Далее идет их описание по каждому классу в алфавитном порядке с указанием

- обороты от начала журнала до даты, предшествующей начальной дате старого периода;
- заголовка процедуры (функции);
- ее англоязычного синонима;
- формата описания;
- входных параметров и их типов, а также типа выходного параметра (для функции);
- назначения процедуры (функции);
- примера использования.

Справа от заголовка процедуры (функции) располагается ряд пиктограмм, определяющих область ее использования:

 — в журналах и операциях;

 — в бланках;

 — в фильтрах картотек;

 — в калькуляторе;

 — только в модификации ***Сетевая***;

 — нет в модификации ***Сетевая***.

После описания процедур и функций в Справочнике приводятся константы, используемые в них.

В конце Справочника приводится алфавитный указатель имен процедур (функций) в русской и английской нотациях.

Для описания *формата вызова* стандартных процедур и функций Турбо Бухгалтера используются следующие синтаксисические обозначения.

Типы параметров процедур (функций) обозначаются следующим образом:

Д	—	Дата
З	—	Запись
Л	—	Логическое
Об	—	Объект
АвтоОбъект	—	Автоматический объект (OLE-объект)
О	—	Отчет
С	—	Строка
ДлС	—	Длинная строка
Упр	—	Управляющая переменная
Ц	—	Целое
Ч	—	Число

Название процедуры (функции) выделяется **БОЛЬШИМИ ЖИРНЫМИ** буквами. Далее *в круглых скобках* описываются *параметры* процедуры (функции), следующие через *точку с запятой*. После имени параметра через *двоеточие* следует описание его *типа*. После списка входных параметров *функции* через *двоеточие* следует описание *типа выходного параметра*. Описание процедуры (функции) завершается *точкой с запятой*. Например:

ОткрытьМассив (НомерРО : Ц; НомерЗ : Ц; ИмяСтрПоля : С;
ИмяПоляУп : С) : Ц;

ОчиститьБланк (ИмяБл : С; ОчищатьПоля : Л);

ВНИМАНИЕ! *Типы параметров при вызове процедуры (функции) указывать не нужно.*

Если параметр процедуры (функции) может иметь два или более возможных типа, то второй и следующие типы указываются через *вертикальную черту*, например:

Сло (Выражение : Ц | Ч) : С;

Пара квадратных скобок после имени параметра означает, что этот параметр является *переменной-массивом*, например:

Альтернатива (Заголовок : С; Строки[ ] : С; Ширина : Ц) : Ц;

Квадратные скобки означают *необязательность* использования, т.е. элементы, заключенные в квадратные скобки, могут отсутствовать в вызове процедуры (функции), например:

Обо (УсловиеОтбора; Дата : Д [; Показатель : С]) : Ч;

ВыборПризнака [(Условие : С)] : С;

Классификация стандартных процедур и функций Турбо Бухгалтера

По своему функциональному назначению процедуры и функции разделяются на следующие группы:

- математические функции;
- строковые функции;
- бухгалтерские функции;
- календарные функции;
- функции доступа к проводкам;
- процедуры и функции доступа к отчетам;
- функции и процедуры выдачи запросов на экран;
- функции и процедуры для работы с файлами;
- процедуры и функции для работы с бланками;
- процедуры и функции управления исключительными ситуациями в бланках;
- процедуры и функции для работы с картотеками через рабочие области;
- функции приведения типов свойств объектов;
- функции для обеспечения совместимости с предыдущими версиями Турбо Бухгалтера;
- сервисные функции;
- функции импорта/экспорта;
- процедуры и функции для работы с SQL-запросами.

Математические функции

Деб — выделяет положительную составляющую числового выражения;

ДробьВЦелое — представляет дробную часть вещественного числа в виде целого;

Если — вычисляет выражение по условию;

Корень — вычисляет значение квадратного корня;

Кре — выделяет отрицательную составляющую числового выражения;

Лог — вычисляет натуральный логарифм по основанию *e*;

Макс — выдает максимальное число из набора;

Мин — выдает минимальное число из набора;

Окр — округляет значение выражения;

Отбр — отбрасывает заданный разряд значения выражения;

Цел — преобразует значение в целое число;

Эксп — вычисляет значение экспоненты.

Строковые функции

Бол — преобразует строки символов из малых букв в большие;

Вставить — вставляет в текст подстроку, начиная с заданной позиции.

ВыделитьСлова — выделяет в тексте заданное количество слов, начиная со слова с заданным по порядку номером и через заданный разделитель.

ВыделитьСлово — выделяет в тексте слово, с заданным номером по порядку и через заданный разделитель.

Длина — возвращает длину исходной строки;

Заменить — заменяет в тексте подстрокой, начиная с заданной позиции.

КодСимвола — определяет код строки из одного символа.

КоличествоСлов — возвращает количество слов в тексте (Text) через заданные разделители (Delimiters).

Мал — преобразует строки символов из больших букв в малые;

Отрезать — отрезает лидирующие и финальные пробелы в строке.

ОтрезатьСлева — отрезает лидирующие пробелы в строке.

ОтрезатьСправа — отрезает финальные пробелы в строке.

Повтор — возвращает строку, полученную за счет повторения исходной заданное число раз;

Подстр — возвращает подстроку;

Поз — возвращает номер позиции заданной подстроки;

Символ — возвращает символ с заданным кодом;

Сло — записывает сумму денег (число) прописью по-русски;

Сло_ — записывает сумму денег (число) прописью на заданном языке;

СловоВСтроке — возвращает “ИСТИНА”, если заданное слово есть в тексте через заданный разделитель.

Соотв — проверяет соответствие заданного образа маске;

Стр — преобразует выражение в строковое представление.

Стр16 — преобразует целое число к его 16-тиричному представлению с заданным числом знаков. Если второй параметр пропущен, то он считается равным 8.

Удалить — удаляет из текста подстроку заданной длины, начиная с заданной позиции.

Формат — функция для форматирования строки.

Бухгалтерские функции

Автообработка — возвращает значение флага **Автообработка;**

БазоваяВалюта — возвращает строку с кратким названием базовой валюты системы, например, “РУБ”.

Бал_Строка — возвращает остаток счетов, заданных условиями отбора аналитических признаков и валют, на указанную дату, относящихся к заданной строке баланса;

Бал_Пров_Нул — возвращает остаток нулевого счета или группы счетов, заданных условием отбора, на указанную дату;

ВзятьГруппуСчетов — формирует в массиве группы счетов, удовлетворяющих условию отбора и списку номеров групп. Возвращает количество элементов в массиве;

ВключаетПризнак — возвращает значение “ИСТИНА”, если введен ный образец содержится в заданном поле картотеки или табличного журнала;

ВыборЖурнала — возвращает имя выбранного из списка журнала;

ВыборКоррСчета — возвращает идентификатор выбранного из списка счета, корреспондирующего с заданным;

ВыборПризнака — возвращает идентификатор выбранного из списка аналитического признака;

ВыборСчета — возвращает идентификатор выбранного из списка счета;

Единица — возвращает единицу измерения по имени аналитического признака или группы;

ЕстьПроводки — возвращает значение “ИСТИНА”, если существуют проводки удовлетворяющие условию отбора за указанный период.

ЗадатьАвтообработку — меняет значение флага **Автообработка;**

ЗадатьОбработкуРеп — задает новое значение флага **Обязательная обработка перед отчетами и раскрытием** в диалоге “Прочие настройки”. При значении “ИСТИНА” данный режим включен, иначе — выключен.

ЗакрытьПлан — закрывает текущий план бухгалтерии;

ИмяЖурнала — возвращает имя журнала по заданному номеру в списке журналов;

Корреспонденция — возвращает значение “ИСТИНА”, если заданные счета являются корреспондирующими;

МаскаПризнаков — возвращает условие отбора на признаки со всеми справочниками, обязательными для заданного счета или связанными с ним;

Наим_Б — возвращает наименование бланка;

Наим_В — возвращает наименование валюты по её коду

Наим Е — возвращает наименование показателя по его имени;

Наим_О — возвращает наименование операции по ее имени;

Наим_П — возвращает наименование признака по его имени;

Наим_С — возвращает наименование счета по его имени;

НомерЖурнала — возвращает номер журнала в списке журналов;

Обо — вычисляет оборот группы счетов;

Оборот — вычисляет оборот группы счетов за период;

Обработать — компилирует журналы хозяйственных операций;

ОбработкаРеп (RefreshRep) — возвращает значение, соответствующее текущему состоянию флага **Обязательная обработка перед отчетами и раскрытием** в диалоге «Прочие настройки». При значении “ИСТИНА” данный режим включен, при значении “ЛОЖЬ” — выключен;

Общ_Пер — возвращает значение общей переменной числового типа;

Общ_Пер_С — возвращает значение общей переменной строкового типа;

Общ_Пер_Файл — возвращает строку с именем файла, в котором описана общая переменная;

Обязат — возвращает значение “ИСТИНА”, если заданный признак обязателен для счета;

Ост — вычисляет остаток группы счетов;

Остаток — вычисляет остаток группы счетов на заданную дату;

ОткрытьПлан — открывает заданный файл с планом бухгалтерии.

ПравильныйПризнак — проверяет, является ли заданное выражение аналитическим признаком;

ПолучитьИмяСправ — возвращает полное имя аналитического справочника по его идентификатору;

ПризнакИспользован — возвращает значение “ИСТИНА”, если признак используется в проводках журналов;

Пров_Деб — проверяет, является ли значение остатка дебетовым;

Пров_Кре — проверяет, является ли значение остатка кредитовым;

Пров_Нул — проверяет, является ли значение остатка нулевым;

Разд_Деб_Ост — вычисляет дебетовый остаток, разделенный по признакам;

Разд_Кре_Ост — вычисляет кредитовый остаток, разделенный по признакам;

РольСправочника — возвращает роль справочника по его имени;

СверткаЖурналов — выполняет свертку журналов и возвращает число обработанных проводок;

СверткаЖурналовКонец — удаляет временные структуры данных, сформированные процедурой **СверткаЖурналовНачало**;

СверткаЖурналовНачало — формирует начальные данные для свертки;

СверткаЖурналовСлед — выполняет свертку журналов на заданном интервале времени и возвращает число обработанных проводок;

Связан — возвращает значение “ИСТИНА”, если заданный признак связан с заданным счетом;

СправИмеетКоличПризнак — возвращает значение “ИСТИНА”, если справочник имеет количественный учет, “ЛОЖЬ” — если не имеет.

СчетАктивный — возвращает значение одной из предопределенных констант (**AccIsActive** — счет активный, **AccIsPassive** — счет пассивный, **AccIsActPas** — счет активно-пассивный, **AccIsZero** — счет нулевой) по заданному номеру счета.

СчетБалансовый — возвращает значение “TRUE”, если счет с заданным номером балансовый;

СчетИспользован — возвращает значение “Истина”, если счет используется в проводках, иначе — “Ложь”;

ТипЖурнала — возвращает строку — тип журнала;

Точность — возвращает точность балансовых счетов;

ЧислоЖурналов — возвращает общее число журналов, содержащихся в текущем плане бухгалтерии.

Календарные функции

Время — возвращает число секунд от начала суток;

Год — по заданной дате вычисляет номер года;

Дат — по заданным номерам дня, месяца, года формирует переменную типа Дата;

День — по заданной дате вычисляет номер дня;

ДеньНедели — возвращает по дате номер дня недели: 0 - воср., 1- пн., 2-вт., 3 – ср., 4 – чт., 5 – пт., 6 – сб.;

ДобавитьМес — прибавляет к некоторой дате заданное количество месяцев;

КартотекаКалендарь — функция предназначена для вызова календаря с привязкой к картотеке, содержащей характеристики дней. Возвращает дату, выбранную пользователем или дату, которая была на входе функции в параметре *Дата*.

Мес — по заданной дате вычисляет номер месяца;

Сегодня — выдает текущую дату.

Функции доступа к проводкам

ПроводАналитика — возвращает список аналитических признаков проводки;

ПроводВалюта — возвращает идентификатор валюты проводки;

ПроводВалюта1 — возвращает идентификатор валюты, в которой задана первая сумма проводки;

ПроводВалюта2— возвращает идентификатор валюты, в которой задана вторая сумма проводки;

ПроводДата — возвращает дату проводки;

ПроводДеб — возвращает идентификатор счета дебета проводки;

ПроводЕдиница — возвращает наименование количественной единицы проводки;

ПроводЖурнал — дает номер журнала, из которого эта проводка получена;

ПроводЗапись — по заданному номеру проводки возвращает запись;

ПроводИмяПризн — возвращает идентификатор аналитического признака заданной проводки;

ПроводКартотека — по заданному номеру проводки возвращает имя картотеки;

ПроводКоличПризнак — возвращает идентификатор аналитического признака проводки, по которому ведется количественный учет;

ПроводКоммент — возвращает комментарий к проводке;

ПроводКре — возвращает идентификатор счета кредита проводки;

ПроводНомерПризнака — возвращает номер признака в проводке.

ПроводОткрытьЖурнал — открывает текстовый/табличный журнал или журнал-картотеку;

ПроводПризнаков — возвращает количество аналитических признаков проводки;

ПроводПризнКол — проверяет, содержит ли указанный аналитический признак проводки количественную сумму;

ПроводСумма — возвращает сумму проводки по заданному показателю;

ПроводСумма1 — возвращает первую сумму проводки;

ПроводСумма2 — возвращает вторую сумму проводки;

ПроводТочность — возвращает точность количественной единицы, в которой была выполнена проводка.

Процедуры и функции доступа к отчетам

ОтчВосстановить — делает текущей строкой строку итогов;

ОтчЗакрыть — закрывает структуру типа Отчет;

ОтчИтогОборот — возвращает оборот по отчету в целом;

ОтчИтогОстатокК — возвращает сальдо на конец периода по отчету в целом;

ОтчИтогОстатокН — возвращает сальдо на начало периода по отчету в целом;

ОтчКоммент — возвращает комментарий текущей строки отчета;

ОтчНазвание — возвращает идентификатор текущей строки отчета;

ОтчОборот — возвращает оборот по строке отчета;

ОтчОстатокК — возвращает сальдо на конец периода;

ОтчОстатокН — возвращает сальдо на начало периода;

ОтчОткрыть — открывает отчет;

ОтчСледующий — делает текущей следующую строку отчета;

ОтчЧислоСтрок — возвращает число строк в отчете.

В функциях доступа к отчетам для задания вида требуемой суммы используются константы **отчСверн** (свернутое сальдо), **отчДеб** (дебетовое сальдо), **отчКре** (кредитовое сальдо).

Процедуры и функции выдачи запросов на экран

Альтернатива — возвращает номер выбранной строки из списка;

Ввод — вводит значение в поле бланка;

ВопрДаНетОтказ — выдает диалог с сообщением и тремя вариантами ответов;

ВопрДаОтказ — выдает диалог с сообщением и двумя вариантами ответов;

Вопрос — выводит модальное окно с заданным сообщением или вопросом, в нижней части которого расположено несколько кнопок. Возвращает номер нажатой кнопки или 0, если окно было закрыто (или нажата клавиша *Esc*);

ДеревоАльтернатив — возвращает 0 при отказе от выбора или номер индекса в массиве выбранной в окне строки (выбор осуществляется двойным щелчком на требуемой строке);

ОкноВыполненияПоказать — открывает окно выполнения задания на экране;

ОкноВыполненияПоложение — в окне выполнения задания продвигает движущуюся полосу на заданную позицию;

ОкноВыполненияЗакрыть — закрывает окно выполнения задания;

ОчиститьТрассировку — удаляет все предупреждающие сообщения из окна сообщений;

Подсказка — выдает в строку состояния заданное сообщение;

Проверка — выдает сообщение об ошибке, если выражение неистинно;

Сообщение — открывает модальное окно с заданным сообщением;

Трассировка — выдает заданное сообщение в окно предупреждений.

Стандартные функции выдачи запросов на экран возвращают результат целого типа. Для проверки этого результата введены специальные целочисленные константы бланков: **кмдВерно, кмдДа, кмдНет, кмдОтказ, кмдПусто**.

Процедуры и функции для работы с файлами

ВыборФайла — открывает стандартный диалог для выбора файла;

ВыборПапки — открывает стандартный диалог для выбора папки;

Дописать — дописывает строку в файл без символа перевода строки;

ЕстьПапка — проверяет, существует ли папка с заданным именем. В случае положительного результата возвращает значение “ИСТИНА”.

ЕстьФайл — проверяет наличие файла с заданным именем. В случае положительного результата возвращает значение “ИСТИНА”;

ЗакрытьРедактор — закрывает окно редактора с заданным именем файла;

Записать — записывает строку в конец файла;

ОткрытьРедактор — открывает файл текстовым редактором;

ПереименоватьПапку — переименовывает папку с заданным именем.

ПереименоватьФайл — переименовывает файл с заданным именем.

ПечатьТекста — распечатывает текстовый файл;

СоздатьПапку — создает новую папку с заданным именем.

СоздатьФайл — создает новый текстовый файл на диске;

УдалитьПапку — удаляет папку с заданным именем.

УдалитьФайл — удаляет файл с заданным именем.

Процедуры и функции для работы с текстовыми файлами по их номерам

ТекстФайлВКонец — устанавливает указатель в конец файла;

ТекстФайлДописатьСтроку — записывает в файл строку, начиная с текущей позиции;

ТекстФайлЗакрыть — закрывает файл;

ТекстФайлЗаписатьСтроку — записывает в файл строку, начиная с текущей позиции, и заканчивает ее символами возврата каретки;

ТекстФайлКонецФайла — возвращает признак достижения конца файла;

ТекстФайлОткрытьНаЗапись — открывает файл на запись и возвращает его номер;

ТекстФайлОткрытьНаЧтение — открывает файл на чтение и возвращает его номер;

ТекстФайлПрочитать — читает строку, начиная с текущей позиции, не переходя на новую строку;

ТекстФайлПрочитатьСтроку — читает строку, начиная с текущей позиции;

ТекстФайлСоздать — возвращает номер созданного файла.

Процедуры и функции для работы с бланками

БланкИмяФайла — возвращает по имени бланка имя bln-файла;

БланкОткрыт — возвращает значение “ИСТИНА”, если бланк открыт.

БланкСуществует — возвращает значение “ИСТИНА”, если бланк существует.

ВзятьВерсиюКартотеки — позволяет получить по заданному номеру нужную версию картотеки.

ВзятьГУИД — возвращает глобальный уникальный идентификатор.

ВзятьИменаБланков (GetBlankNames) — предназначена для заполнения строкового массива бланков именами бланков по условию отбора. Возвращает количество элементов в массиве.

ВзятьИмяАктивногоБланка — возвращает имя активного бланка. В случае, если активое окно не является окном с бланком, то возвращает пустую строку.

ВзятьКакМассивSQL — возвращает количество загруженных записей в массив.

ВзятьСправочникАналит — возвращает наименование аналитического справочника по аналитическому признаку.

ВключитьКоманды — предназначена для включения или выключения тех или иных команд в интерфейсе пользователя ТБ на время выполнения того или иного бланка, из которого она вызвана. Значение “ИСТИНА” — включает команды, “ЛОЖЬ” — выключает.

ВремяМодификацииСек — возвращает число секунд с начала суток, в которые была выполнена последняя модификация текущей записи;

ВремяСозданияСек — возвращает число секунд с начала тех суток, в которые была создана текущая редактируемая запись;

ВставитьВМассив — вставляет заданный элемент в массив на заданную позицию;

ВставитьВМассивСорт (проц.) **—** вставляет элемент в упорядоченный массив с сохранением упорядоченности;

ВставитьВМассивСорт (функ.) — возвращает номер позиции, на которую ставится новый элемент в упорядоченный массив;

ВставитьРамку — вставляет рамку в повторяющуюся секцию;

Вставка — вставляет проводку/операцию в текстовый журнал;

ВставкаВТаблЖурнал — вставляет проводку/операцию в табличный журнал;

ВыполнитьПроцедуру — процедура бланка для вызова процедуры по её имени;

ВыполнитьФункцию — процедура бланка для вызова функции по её имени;

ДлинаМассива — возвращает длину динамического или статического массива по заданному имени;

ЗагрузитьИнтернетФайл — возвращает значение "ИСТИНА", если страница интернета загружена в файл успешно.

ЗагрузитьРаздел — загрузка данных из текстового файла в заданный раздел табличного журнала;

Закрыть — осуществляет выход из бланка;

КартотечныйФильтрBSQL — возвращает значение "ИСТИНА", если картотечный фильтр удалось полностью перевести в SQL-фильтр;

КопироватьЗапись — копирование записи из одной рабочей области в другую для картотек с одинаковой структурой. Дублирование записей в картотеке;

КопироватьМассив — копирует содержимое одного массива в другой.

КопироватьФайл — копирует файл или папку с заданным именем;

МаксВМассиве — возвращает индекс максимального элемента массива;

МинВМассиве — возвращает индекс минимального элемента массива;

МодальныйРежим — возвращает значение "ИСТИНА", если бланк работает в модальном режиме;

Модуль — возвращает абсолютное значение переменной целого или числового типа;

НайтиОбъект — возвращает "ИСТИНА", если объект на шаблоне найден и описан в бланке. Если объект не найден, то функция возвращает "ЛОЖЬ";

ОбработатьАналит — предназначена для обновления (перекомпиляции аналитики). В случае, когда отключена автообработка.

ОткрБланк — открывает бланк;

ОткрытьТабличныйЖурнал — открывает раздел табличного журнала с заданным именем на заданной записи. Функция возвращает “ИСТИНА”, если удалось открыть раздел табличного журнала;

ОчиститьБланк — очищает значение входных полей бланка;

ОчиститьПеременную — очищает переменную с заданным именем;

ПеременнаяСуществует (VariableExist) — возвращает значение “ИСТИНА”, если переменная бланка существует.

ПеременныеМодифицированы — возвращает значение “ИСТИНА”, если переменные бланка с заданным именем были изменены;

Печать — печатает бланк из процедуры;

ПоискВМассиве — возвращает индекс указанного элемента в массиве (динамическом или статическом), если он существует, или “-1” — если его нет;

ПравоваяПоддержка — открывает информационно-правовую систему "Референт” или “Гарант”;

РедакторОтменить — отменяет все сделанные в бланке-редакторе картотеки изменения;

РедакторПрименить — вносит изменения, сделанные в бланке-редакторе картотеки;

СнятьМодификацию — очищает признак модифицированности полей бланка;

СортироватьМассив — сортирует массив с заданным именем.

СортироватьСекцию — сортирует секцию по одному из столбцов.

СохранитьРаздел — сохраняет информацию из раздела табличного журнала в заданном текстовом файле;

СуммаМассива — суммирует значения заданного числа элементов массива;

Текущая — возвращает ссылку на текущую редактируемую запись;

ТекущееСостояние — определяет состояние текущей записи в бланке-редакторе картотеки. Может возвращать одно из 4-х значений: запись не изменялась, запись изменена, запись вставлена и запись удалена.

УдалитьИзМассива — удаляет элемент с заданной позицией из массива;

УдалитьИзДинамическогоМассива — удаляет элемент с заданной позицией из динамического массива.

УдалитьРамку — удаляет рамку из повторяющейся секции;

Процедуры и функции управления исключительными ситуациями в бланках

БазисПользовательскихОшибок — возвращает номер начальной границы пользовательских исключений;

КодОшибки — возвращает уникальный номер исключительной ситуации;

СообщениеОшибки — возвращает строку сообщения возникшей ошибки и может быть использована только в блоке Except ... End.

УстОшибку — устанавливает исключительную ситуацию с заданным уникальным номером.

Процедуры и функции для работы с картотеками

ВзятьГрупповоеИмяПоля — возвращает имя поля в иерархической картотеке, которое хранит ссылку на запись группы.

ВзятьГрупповоеИнфоИмяПоля — возвращает информационное имя поля в иерархической картотеке.

ВзятьКартотекуПоСсылочномуПолю — возвращает строку с именем картотеки, на которую ссылается заданное поле в заданной картотеке;

ВзятьКартПоле — возвращает содержимое заданного поля для указанного ключа;

ВзятьПоляКартотеки — заполняет строковый массив полями картотеки указанного типа и возвращает длину массива, т.е. количество полей.

ВзятьСхему — возвращает имя схемы доступа пользователя в сетевой версии или пустую строку в локальной версии.

ВзятьТипПоляКартотеки — возвращает целое значение, которое соответствует следующим предопределенным константам: *mtlREAL*, *mtlINTEGER*, *mtlBOOLEAN*, *mtlSTRING*, *mtlDATE*, *mtlSUBTAB*, *mtlREF*, *mtlTimed*;

Видимая — возвращает значение “ИСТИНА”, если картотека с заданным именем видимая. В локальной версии всегда возвращает значение “ИСТИНА”;

ВходитВГруппу — служит для определения вхождения записи в группу иерархической картотеки. Возвращает значение “ИСТИНА”, если ссылка на запись в картотеке входит в группу по ссылке;

ВыборКартотеки — возвращает константу *кмдВерно* (*cmOK*), если картотека выбрана, иначе — *кмдОтказ* (*cmCancel*);

Задан — осуществляет проверку наличия ссылки в поле картотеки;

ЗаписьСуществует — возвращает значение “ИСТИНА”, если запись по ссылке существует в картотеке.

ЕстьКартотека — возвращает значение “ИСТИНА”, если картотека есть в базе данных.

Иерархическия — возвращает значение “ИСТИНА” если картотека с заданным именем является иерархической или значение “ЛОЖЬ”, если картотека неиерархическая.

КартГруппа — возвращает значение “ИСТИНА”, если запись по ссылке в картотеке является группой.

КартотекаЖурнала — возвращает имя картотеки, с которой связан передаваемый jdf-файл;

КартотекаПризнака — возвращает имя картотеки, если найден заданный признак, иначе — пустую строку;

КартотекаПризнакЗаписи — возвращает строку с формированным аналитическим признаком из справочника и значения поля картотеки. Если картотека — иерархическая, то признак формируется с учётом иерархии.

МожноВставить — возвращает значение “ИСТИНА”, если в картотеку с заданным именем можно добавлять записи или значение “ЛОЖЬ”, если нельзя.

МожноИзменить — возвращает значение “ИСТИНА”, если в картотеке с заданным именем можно изменять записи или значение “ЛОЖЬ”, если нельзя.

МожноУдалять — возвращает значение “ИСТИНА”, если в картотеке с заданным именем можно удалять записи или значение “ЛОЖЬ”, если нельзя.

ОткрытьКартотеку (п)— открывает окно с картотекой;

ОткрытьКартотеку (ф) — возвращает ссылку на запись картотеки;

ПризнакВЗапись — возвращает указатель на запись, в которой описан заданный признак;

ПризнакЗаписи — возвращает идентификатор признака, созданного записью картотеки, по его порядковому номеру;

ТолькоНаЧтение — возвращает значение “ИСТИНА”, если в картотека с заданным именем доступна только для чтения или значение “ЛОЖЬ”, если в ней можно производить изменения.

ЧислоПризнаковЗаписи — возвращает число признаков, созданных записью картотеки.

ЭкспортКартотекиВЭксель — сохраняет изображение картотеки в формате таблицы Excel.

Процедуры и функции для работы с картотеками через рабочие области

ВзятьГруппа — возвращает значение “ИСТИНА”, если запись с заданным номером является группой.

ВзятьКакМассив — считывает структурное поле картотеки в переменные-массивы бланка;

ВзятьКлючЗаписи — возвращает ссылку на запись с заданным номером в картотеке;

ВзятьНомерЗаписи — возвращает номер записи в рабочей области;

ВзятьПоле* — возвращает значение поля картотеки;

ВзятьПоляВПеременные — процедура для загрузки в переменные бланка полей картотеки из рабочей области. Записывает значение заданного поля записи с заданным номером в заданной рабочей области в переменную бланка.

ВзятьФильтр — возвращает текущий фильтр, установленный для заданной рабочей области;

ВставитьЗапись — вставляет пустую запись в картотеку и возвращает ее номер;

ВставитьЗаписьПоКлючу — вставляет пустую запись с заданным уникальным ключом в картотеку и возвращает ее номер;

ЗаблокироватьЗапись — возвращает значение “ИСТИНА”, если запись удалось заблокировать;

ЗакончитьИзменения — вносит все изменения отредактированных записей в картотеку;

ЗакрытьРабочуюОбласть — закрывает рабочую область;

ЗаписатьВПоляПеременные — записывает значение переменной в заданное поле записи с заданным номером в заданной рабочей области. Возвращает номер записи.

ЗаписатьКакМассив — запись переменных-массивов из бланка в картотеку;

ЗаписатьПоле* — записывает значение в поле и возвращает номер записи;

Записей — возвращает число записей картотеки;

КартотекаРабочейОбласти — возвращает имя картотеки по заданному номеру рабочей области;

КопироватьЗапись — копирование записи из одной рабочей области в другую для картотек с одинаковой структурой. Дублирование записей в одной картотеке;

НачатьИзменения — начинает редактирование записей картотеки;

ОткрытьБланкРедактор — открывает бланк-редактор для заданной записи картотеки;

ОткрытьМассив — открывает рабочую область на структурное или периодическое поле картотеки;

ОткрытьРабочуюОбласть — открывает рабочую область картотеки;

ОтменитьИзменения — отменяет сделанные изменения;

Поиск — выполняет поиск записи в картотеке;

Поиск* — выполняет поиск записи в картотеке и возвращает переменную логического типа;

УдалитьВсеЗаписи — удаляет все записи с учетом фильтра в открытой рабочей области;

УдалитьЗапись — удаляет запись из картотеки;

Упорядочить — упорядочивает рабочую область по одному или нескольким полям;

УстФильтр — устанавливает фильтр отбора записей.

Функции проверки типов свойств объектов

В языке бланков не различаются виды объектов (т.е. и кнопке действия, и флагу, и списку в процедурах соответствуют переменные одного и того же типа — “Объект”). В связи с тем, что наборы свойств и методов у них разные, не представляется возможным на этапе компиляции бланка определить тип того или иного свойства и даже факт его наличия или отсутствия у данного объекта. Поэтому для чтения значения этих свойств используются следующие функции проверки типов свойств объектов:

КакДата — проверяет, соответствует ли тип заданного свойства объекта типу Дата;

КакЗапись — проверяет, соответствует тип заданного свойства объекта типу ЗАПИСЬ;

КакЛогическое — проверяет, соответствует ли тип заданного свойства объекта логическому типу;

КакСтрока — проверяет, соответствует ли тип заданного свойства объекта типу Строка;

КакЦелое — проверяет, соответствует ли тип заданного свойства объекта целому типу;

КакЧисло — проверяет, соответствует ли тип заданного свойства объекта числовому типу.

Функции поддержки автоматических объектов (OLE Avtomation)

СоздатьАвтоОбъект — функция служит для создания автоматического объекта. По имени класса OLE-объекта, зарегистрированного в системе Windows, создает его экземпляр и возвращает указатель на вновь созданный объект. Если объект невозможно создать, генерируется исключительная ситуация (ошибка). Созданный объект обязательно надо освобождать процедурой **ОчиститьПеременную (ClearVariable)**.

ОткрытьАвтоОбъект — функция служит для открытия автоматического объекта. По имени класса OLE-объекта, зарегистрированного в системе Windows, проверяет, существует ли экземляр указанного класса. Если объект уже существует, функция возвращает ссылку на него, и далее с объектом можно работать точно также, как и с объектом, созданным с помощью функции **СоздатьАвтоОбъект (CreateAutoObject)**. В противном случае, если объектов указанного класса пока нет, то функция может создать новый объект при условии, что второй параметр равен “ИСТИНА”. Открытый обеъкт обязательно надо освобождать процедурой **ОчиститьПеременную** (ClearVariable).

КакАвтоОбъект — функция служит для преобразования ссылки OLE-объекта к типу автоматический объект (АвтоОбъект). Открытый таким образом автообъект обязательно надо освобождать процедурой **ОчиститьПеременную (ClearVariable)**.

Процедуры и функции для работы с DBF-файлами

ДобавитьБланкBDbf — записывает содержимое бланка в новую запись DBF-файла;

ЗакрытьDbf — закрывает DBF-файл, открытый функцией **ОткрытьDbf**;

ЗаписатьПолеBDbf — записывает информацию из бланка в текущую запись DBF-файла;

КонецФайлаDbf — возвращает значение “ИСТИНА”, если достигнут конец DBF-файла;

ОткрытьDbf — открывает DBF-файл на чтение и запись;

ПерваяЗаписьDbf — переходит в начало DBF-файла;

СледующаяЗаписьDbf — переходит на следующую запись в DBF-файле;

УдалитьЗаписьDbf — удаляет текущую запись в DBF-файле;

ЧислоЗаписейBDbf — возвращает число записей в DBF-файле;

ЧитатьБланкИзDbf — считывает текущую запись из DBF-файла в бланк;

ЧитатьПолеИзDbf — считывает информацию из поля текущей записи DBF-файла в переменную бланка.

Функции для обеспечения совместимости с предыдущими версиями Турбо Бухгалтера

Маска — формирует условие отбора проводок;

Масч — возвращает корректную маску имени счета;

Месяц — выдает русское название месяца по его номеру;

Месяца — выдает русское название месяца по его номеру в родительном падеже;

НДС_Втч — возвращает сумму НДС по указанной ставке;

Прих — вычисляет сумму остатка на начало месяца и дебетового оборота с начала месяца;

Lang — преобразует одну строку в другую, которая может быть именем файла MS-DOS.

Функции приведения типов

ДатуВСтроку — по заданной дате возвращает ее строковое представление в формате ДД.ММ.ГГГГ;

ЗаписьВСтроку — возвращает строковое представление заданной ссылки на картотечную запись;

ЗаписьКчислу — возвращает числовой тип записи по заданной ссылке;

СтрокуВДату — по текстовому представлению даты возвращает значение типа дата;

СтрокуВЧисло — по текстовому представлению числа возвращает значение типа Число.

СтрокуВЗапись — возвращает ссылку на картотечную запись;

ЧислоВЗапись — возвращает ссылку на картотечную запись.

Кроме перечисленных функций приведения типов можно использовать аналогичные функции из исходного описания бланка ***Раб. Формат***.

Сервисные процедуры и функции

Версия — выдает номер модификации и платформу (локальная или сетевая) Турбо Бухгалтера;

ВзятьВерсиюБД — возвращает версию сервера БД, на котором хранится заданная картотека;

ВзятьТипБД — возвращает тип сервера БД, на котором хранится заданная картотека;

ВыполнитьПрограмму (п) — запускает заданное приложение с командной строки;

ВыполнитьПрограмму (ф) — запускает заданное приложение с командной строки и возвращает "0" или константу ошибки;

ЗакрытьПрог — закрывает программу Турбо Бухгалтер;

ЗаписатьПеремВДВТ — записывает переменные из бланка в DBT-файл;

ЗаписатьПеремВКартотеку — записывает переменные бланка в картотеку;

Звук — выдает звуковой сигнал средствами Windows;

ИмяБазы — возвращает псевдоним базы (алиас в плане бухгалтерии), в которой находится картотека с заданным именем.

ИмяПользователя — возвращает имя пользователя, подключившегося к заданной БД;

ИмяСхемы — возвращает имя схемы доступа, по которой подключились к заданной БД;

Календарь — открывает модальное окно с календарем;

Калькулятор — открывает модальное окно с калькулятором;

КонтекстнаяСправка — вызывает Справочную систему;

Отчет — создает внутренний отчет с заданным именем;

ПеремПБ — возвращает значение макроса из плана бухгалтерии;

ПроверкаЛицензии — возвращает число лицензий (рабочих мест);

СчитатьПеремИзДВТ — считывает переменные из DBT-файла в открытый бланк;

СчитатьПеремИзКартотеки — считывает переменные бланка из картотеки.

Функции импорта/экспорта

ИмпортБазы — импорт данных из файла *.crd (константа ieCRD) или *.dbf (константа ieDBF), содержащего структуру базы данных, в базу данных;

ИмпортБланка — импорт содержимого документа MS WORD (константа формата ieWord|изВорд) или текстового документа (константа формата ieText|изТекст) в заданный бланк;

ИмпортКартотеки — импорт из файла *.crd (константа ieCRD) или *.dbf (константа ieDBF) в заданную картотеку;

ЭкспортБазы — в зависимости от выбранной константы формата осуществляет экспорт всей базы данных в форматы *.crd, *.dbf, *.xls;

ЭкспортБланка — в зависимости от выбранной константы формата осуществляет экспорт бланка в форматы *.rtf, *.doc, *.html, *.xls;

ЭкспортКартотеки — в зависимости от выбранной константы формата осуществляет экспорт картотеки в форматы .

Процедуры и функции для работы с SQL-запросами

SQLЗапросыРазрешены — возвращает значение “ИСТИНА”, если у пользователя в плане ТБ.Сервера установлена привилегия “SQL запросы”, то есть разрешены прямые SQL-запросы с помощью ОткрытьРабочуюОбластьSQL, ВыполнитьSQL.

ВзятьДействитИмяКартотеки — возвращает физическое имя картотеки по ее логическому имени;

ВыполнитьSQL — выполняет SQL-запрос;

ОбновитьКартотеку — принудительно обновляет данные картотеки на клиентских местах;

ОткрытьРабочуюОбластьSQL — открывает рабочую область по SQL-запросу.

Математические функции

Функция Деб

Ж Б ф К

Синоним:

Deb

Формат описания:

Деб (Выражение : Ч) : Ч;

Входные параметры:

Выражение — любое числовое выражение

Назначение:

Возвращает значение своего параметра, если оно неотрицательно, или ноль — в противном случае.

Примечание:

Этой функцией можно воспользоваться для выделения дебетовой составляющей свернутого сальдо счета.

Примеры:

```
Д = Деб(5); -- Д = 5
Д = Деб(-3); -- Д = 0
```

Функция ДробьВЦелое

Ж Б ф К

Синоним:

FracToInteger

Формат описания:

ДробьВЦелое (Выр : Ч | С [; aPrec: Integer): Integer;

Окр (Выр : Ч | С [; ЧислоЗн : Ц]) : Ч;

Входные параметры:

Выр — числовое значение или его строковое представление

ЧислоЗн — число знаков дробной части

Назначение:

Функция представления дробной части вещественного числа в виде целого. Если в заданном строковом представлении числа содержится ошибка, то результат функции будет равен нулю.

Примечания:

1. Если второй параметр > 0, то берется для представления указанное в параметре число знаков дробной части.

2. Если второй параметр < 0, то берется для представления указанное в параметре число знаков целой части справа налево от разделителя дробной части.

3. Если второй параметр = 0, то берутся для представления все знаки дробной части.

4. Если второй параметр не указан, то число знаков = 0.

Пример:

```
FracToInteger(123.456, 3);   -- 456
FracToInteger(123.456, 2);   -- 45
FracToInteger(123.456, 4);   -- 4560
FracToInteger(123.0456, 5);  -- 4560
FracToInteger(123.456);      -- 456
FracToInteger(-123.456, 2);  -- 45
FracToInteger(123.456, -2);  -- 23
FracToInteger(123.456, -3);  -- 123
FracToInteger(123.456, -4);  -- 123
FracToInteger(-123.456, -2); -- 23
FracToInteger(«123.456», 0); -- 456
FracToInteger(«123.456S»);   -- 0
```

Функция Если

Ж Б ф К

Синоним:

If

1 вариант

Формат описания:

Если (ЛогВыр1 : Выр1 [| ЛогВыр2 : Выр2]; Выр3);

Входные параметры:

*ЛогВыр** — логическое выражение

*Выр** — выражение любого типа

Назначение:

Вычисляет выражение в соответствии с заданным условием.

В данном варианте функции **ЕСЛИ**, если *ЛогВыр* истинно, то вычисляется *Выр1*. Если истинно какое-либо из выражений, обозначенных как *ЛогВыр2*, то вычисляется соответствующее ему *Выр2*. Если же ни одно из заданных условий не выполнено, то вычисляется *Выр3*. При этом все выражения, обозначенные как "*ЛогВыр**" должны иметь логический тип. А все выражения, обозначенные как "*Выр**", — иметь одинаковый тип. Единственное исключение из последнего правила: допустимо в одной функции **ЕСЛИ** смешивать выражения целого и числового типа.

Примечание:

Знак "|" в формате вызова отделяет одно условие от другого.

Пример:

```
c = Если ( a>0:a | a=0:5 | a<0:0 )
-- Если a=1, c=1.
-- Если a=0, c=5.
-- Если a=-1, c=0.
```

2 вариант

ЗАМЕЧАНИЕ. *Второй вариант функции* ***ЕСЛИ*** *включен для того, чтобы обеспечить совместимость с данными ранних версий программы.*

Формат описания:

Если (ЛогВыр1; Выр1; Выр2);

Входные параметры:

*ЛогВыр** — логическое выражение

*Выр** — выражение любого типа

Назначение:

В данном варианте стандартные *Выр1* и *Выр2* должны иметь одинаковый тип. Допустимо при этом, чтобы одно из этих выражений было числового типа, а другое — целого.

Пример:

```
c = Если ( a>0, 5, 0 );
-- Если a=1, c=5.
-- Если a=0, c=0.
-- Если a=-1, c=0.
```

Функция Корень

Ж Б ф К

Синоним:

Sqrt

Формат описания:

Корень (Выражение : Ч) : Ч;

Входные параметры:

Выражение — любое числовое выражение

Назначение:

Возвращает квадратный корень из числового выражения.

Пример:

```
ПРОЦ Р1;
 ПЕРЕМ Стор1, Стор2 : Ч;
 ПЕРЕМ кмдРез : Ц;
 -- вводим сторону прямоугольника
 кмдРез = Ввод (Стор1,"Введите длину первой стороны"+" прямоугольника ");
 -- вводим другую сторону прямоугольника
 кмдРез = Ввод (Стор2,"Введите длину второй стороны"+" прямоугольника ");
 -- выводим длину диагонали
 Сообщение ("Длина диагонали прямоугольника равна " +
             Стр(Sqrt(Стор1*Стор1+Стор2*Стор2),2));
Конец;
```

Функция Кре

Ж Б ф К

Синоним:

Cre

Формат описания:

Кре (Выражение : Ч) : Ч;

Входные параметры:

Выражение — любое числовое выражение

Назначение:

Возвращает абсолютное значение своего параметра (если оно отрицательно) или ноль (в противном случае).

Примечание:

Этой функцией можно воспользоваться для выделения кредитовой составляющей свернутого сальдо счета.

Примеры:

```
Кр = Кре(5); -- Кр = 0
Кр = Кре(-3); -- Кр = 3
```

Функция Лог

Ж Б ф К

Синоним:

Log

Формат описания:

Лог (Выражение : Ц | Ч) : Ч;

Входные параметры:

Выражение — выражение целого или числового типа

Назначение:

Вычисляет натуральный (по основанию *e*) логарифм значения параметра. Если параметр не положителен, то при вычислении этой функции генерируется ошибка.

Примеры:

```
a = Лог (1); -- получим a=0.0
b = Лог (2.7182818); -- получим b=1.0
```

Функция Макс

Ж Б ф К

Синоним:

Max

Формат описания:

Макс (Выр1 : Ч ; Выр2 : Ч; .... ; ВырN : Ч) : Ч;

Входные параметры:

Выр1, Выр2, ... , Выр — любые числовые выражения

Назначение:

Возвращает максимальное число из списка числовых выражений.

Пример:

```
МаксЧ = Макс (2, 9, 3, 2+5, -2.3); -- МаксЧ = 9
```

Функция Мин

Ж Б ф К

Синоним:

Min

Формат описания:

Мин (Выр1 : Ч; Выр2 : Ч; .... ; ВырN : Ч) : Ч;

Входные параметры:

Выр1, Выр2, ... , Выр — любые числовые выражения

Назначение:

Возвращает минимальное число из списка числовых выражений.

Пример:

```
МинЧ = Мин (2, 9, 3, 2+5); -- МинЧ = 2
```

Функция Окр

Ж Б ф К

Синоним:

Round

Формат описания:

Окр (Выр : Ч | С [; ЧислоЗн : Ц]) : Ч;

Входные параметры:

Выр — любое числовое выражение или его строковое представление

ЧислоЗн — число знаков после десятичной точки

Назначение:

Округляет значение первого параметра по математическим правилам. Если в заданном строковом представлении числа содержится ошибка, то результат функции будет равен нулю.

Примечания:

1. Если второй параметр отсутствует, то округление происходит до целого числа. Если второй параметр положителен, то он задает количество знаков в дробной чисти результирующего числа после

десятичной точки. При отрицательном значении параметра округляются разряды числа, начиная с номера, равного модулю значения второго параметра.

Примеры:

```
Число = Окр (49.125); -- Число = 49
Число = Окр ("49.125"); -- Число = 49
Число = Окр (49.125,2); -- Число = 49.13
Число = Окр (49.12,-1); -- Число = 50
Число = Окр ("49.12S"); -- Число = 0
```

2. Можно управлять режимом округления в процессе работы программы, установив при запуске Турбо Бухгалтера следующие ключи:

- Ключ **RoundAbs;**

  Командная строка имеет следующий вид: **Tbw.exe /RoundAbs**

  При этом отрицательные числа округляются по правилу: Res := -Окр(-x).

Пример:

```
ОКР(-10009.645,2) = -10"009.64;    --Если ключ не установлен
ОКР(-10009.645,2) = -10"009.65;    --Если ключ установлен
```

- Ключ **RoundDelta;**

  Командная строка имеет следующий вид:
  Tbw.exe /RoundDelta=<число>

  При этом задается точность округления. Допустимые значения данного параметра от 0 до 18. По умолчанию значение Delta равно 10.

- Ключи **RoundAbs и RoundDelta;**

  Командная строка имеет следующий вид:
  Tbw.exe /RoundAbs RoundDelta=<число>

  При этом отрицательные числа округляются по правилу: Res := -Окр(-x) с заданной точностью.

Функция Отбр

Ж Б ф К

Синоним:

Trunc

Формат описания:

Отбр (Выр : Ч | С [; ЧислоЗн : Ц]) : Ч;

Входные параметры:

Выр — любое числовое выражение или его строковое представление

ЧислоЗн — количество знаков после десятичной точки

Назначение:

Отбрасывает или обнуляет соответствующие разряды числа в исходном выражении. Если в заданном строковом представлении числа содержится ошибка, то результат функции будет равен нулю.

Примечание:

Если второй параметр не задан, то отбрасывается вся дробная часть числа. Если второй параметр отрицателен, то начиная с разряда, равного модулю значения второго параметра, все значащие цифры обнуляются, а дробная часть отбрасывается. Если второй параметр положителен, то в дробной части результирующего числа сохраняется столько знаков, сколько указано во втором параметре.

Примеры:

```
Число = Отбр (49.125); -- Число = 49
Число = Отбр ("49.125"); -- Число = 49
Число = Отбр (49.125,2); -- Число = 49.12
Число = Отбр (49.12,-1); -- Число = 40
Число = Отбр ("49.12S"); -- Число = 0
```

Функция Цел

Ж | Б | ф | К

Синоним:

Int

Формат описания:

Цел (Выр : Ч | С) : Ц;

Входные параметры:

Выр — любое числовое выражение или его строковое представление

Назначение:

Преобразует числовое выражение или его строковое представление в целое число. Если в заданном строковом представлении числа содержится ошибка, то результат функции будет равен нулю.

Примеры:

```
Число = Цел (49.125); -- Число = 49
Число = Цел ("49.125"); -- Число = 49
Число = Цел ("49.12S"); -- Число = 0
```

Функция Эксп

Ж Б ф К

Синоним:

Exp

Формат описания:

Эксп (Выр : Ч) : Ч;

Входные параметры:

Выр — любое числовое выражение

Назначение:

Вычисляет экспоненту (показательную функцию с основанием *e*) от заданного числового выражения.

Примеры:

```
Число = Эксп (1); -- Число = 2.7182819
Число = Эксп (0); -- Число = 1
```

Строковые функции

Функция Бол

Синоним:

Up

Формат описания:

Бол (Строка : С) : С;

Входные параметры:

Строка — любая строка

Назначение:

Переводит все символы в заданной строке в верхний регистр, т.е. заменяет все строчные буквы на прописные.

Пример:

```
Строка = Бол ("Турбо Бухгалтер"); -- Строка = "ТУРБО БУХГАЛТЕР"
```

Функция Вставить

Синоним:

Insert

Формат описания:

Вставить (Подстрока : С; Текст : С; Индекс : Ц) : С;

Входные параметры:

Подстрока — подстрока, которая будет вставляться в текст

Текст — любой текст

Индекс — позиция в тексте, с которой будет вставляться подстрока

Назначение:

Вставляет в текст подстроку, начиная с позиции *Индекс*.

Пример:

```
var S1 : String;
  S1 = Insert(" плюс", "Двадцать один", 9);
-- S1 содержит "Двадцать плюс один"
```

Функция ВыделитьСлова

Ж Б ф

Синоним:

ExtractWords

Формат описания:

ВыделитьСлова (Слово : Ц; Количество : Ц; Текст : С; Разделитель : С) : С;

Входные параметры:

Слово — номер слова в тексте

Количество — количество выделяемых слов

Текст — любой текст

Разделитель — заданный разделитель слов в тексте

Назначение:

Выделяет в тексте заданное количество слов, начиная со слова с заданным по порядку номером и через заданный разделитель.

Примеры:

```
var S1 : String;
 S1 = ExtractWord(2, 2, "один, два три", " ");
-- S1 содержит "два три"
```

Функция ВыделитьСлово

Ж Б ф

Синоним:

ExtractWord

Формат описания:

ВыделитьСлово (Слово : Ц; Текст : С; Разделитель : С) : С;

Входные параметры:

Слово — номер слова в тексте

Текст — любой текст

Разделитель — заданный разделитель слов в тексте

Назначение:

Выделяет в тексте слово, с заданным номером по порядку и через заданный разделитель.

Примеры:

```
var S1 : String;
  S1 = ExtractWord(3, "один, два три", " ");
-- S1 содержит "три"
```

Функция Длина

Ж Б Ф

Синоним:

Length

Формат описания:

Длина (Строка : С) : Ц;

Входные параметры:

Строка — любая строка

Назначение:

Возвращает длину заданной строки.

Пример:

```
Дл = Длина ("Турбо Бухгалтер"); -- Дл = 15
```

Функция Заменить

Ж Б Ф

Синоним:

Replace

Формат описания:

Заменить (Подстрока : С; Текст : С; Индекс : Ц) : С;

Входные параметры:

Подстрока — подстрока, которой будет производиться замена в тексте

Текст — любой текст

Индекс — позиция в тексте, с которой будет производиться замена

Назначение:

Заменяет в тексте подстрокой, начиная с позиции *Индекс*.

Пример:

```
var S1 : String;
```

```
  S1 = Replace("два", "Двадцать один", 10);
-- S1 содержит "Двадцать два"
```

Функция КодСимвола

Ж Б ф

Синоним:

Ascii

Формат описания:

КодСимвола (Символ : С) : Ц;

Входные параметры:

Символ — строка из одного символа

Назначение:

Определяет код строки из одного символа.

Примеры:

```
var vRes: Integer;
 vRes = Ascii("X"); -- vRes = 88
```

Функция КоличествоСлов

Ж Б ф

Синоним:

WordsCount

Формат описания:

КоличествоСлов (Текст : С; Разделитель : С) : Ц;

Входные параметры:

Текст — любой текст

Разделитель — заданный разделитель слов в тексте

Назначение:

Возвращает количество слов в тексте через заданные разделители.

Примеры:

```
var X : Integer;
X = WordsCount("Налоговая декларация","а");
-- X равно 5
```

Функция Мал

Синоним:

Lo

Формат описания:

Мал (Строка : С) : С;

Входные параметры:

Строка — любая строка

Назначение:

Переводит все символы в заданной строке в нижний регистр (заменяет все прописные буквы на строчные).

Пример:

```
Строка = Мал ("Турбо Бухгалтер"); -- Строка = "турбо бухгалтер"
```

Функции Отрезать, ОтрезатьСлева, ОтрезатьСправа

Синонимы:

Trim, TrimLeft, TrimRight

Формат описания:

Отрезать (Строка : С) : С;

ОтрезатьСлева (Строка : С) : С;

ОтрезатьСправа (Строка : С) : С;

Входные параметры:

Строка — любая строка

Назначение:

Отрезать — отрезает лидирующие и финальные пробелы в строке.

ОтрезатьСлева — отрезает лидирующие пробелы в строке.

ОтрезатьСправа — отрезает финальные пробелы в строке.

Пример:

```
func Trim (s : String) :String;
```

```
  -- удаляет из строки ведущие и замыкающие пробелы
    s = TrimLeft(s);
    s = TrimRight(s);
    return s;
  end; -- Trim
```

Функция Повтор

Ж Б ф

Синоним:

RepStr

Формат описания:

Повтор (Строка : С; ЧислоПовт : Ц) : С;

Входные параметры:

Строка — любая строка

ЧислоПовт — число копий

Назначение:

Возвращает строку, полученную из заданной путем добавления в конец определенного числа копий заданной строки.

Примеры:

```
Строка = Повтор ("ТБ", 3); -- Строка = "ТБТБТБ"
Строка = Повтор ("ТБ", -1); -- Строка = ""
```

Функция Подстр

Ж Б ф

Синоним:

SubStr

Формат описания:

Подстр (Строка : С; НачПоз : Ц; Кол : Ц) : С;

Входные параметры:

Строка — любая строка

НачПоз — начальная позиция

Кол — количество символов

Назначение:

Выделяет подстроку из строки с заданным количеством символов, начиная с заданной позиции.

Примечания:

1. При выделении подстроки до конца строки в качестве значения параметра *Кол* можно подставить значение функции **Длина (***Строка***)**.

2. Если начальная позиция задана меньшей или равной нулю, то переменная *НачПоз* принимает значение 1.

Примеры:

```
Строка = "Турбо Бухгалтер";
Строка1 = Подстр (Строка, 1, 5); -- Строка1 = “Турбо”
Строка2 = Подстр (Строка, 7, Длина (Строка)); -- Строка2 = "Бухгалтер"
```

Функция Поз

Ж Б Ф

Синоним:

Pos

Формат описания:

Поз (Подстрока : С; Строка : С) : Ц;

Входные параметры:

Подстрока — подстрока заданной строки

Строка — любая строка

Назначение:

Возвращает позицию подстроки в заданной строке. В случае отсутствия данного параметра функция возвращает нуль.

Пример:

```
Позиция = Поз (“Бухгалтер”, “Турбо Бухгалтер” ); -- Позиция = 7
```

Функция Символ

Б

Синоним:

CHR

Формат описания:

Символ (Код : Ц) : С;

Входные параметры:

Код — символьный код

Назначение:

Возвращает символ по заданному коду и может использоваться для формирования текстовых файлов со специальными символами, в частности: для разделения полей символом табуляции Символ(9), для разделения полей символом перевода строки Символ(13) и т.д.

Пример:

```
Проц Р1;
 Перем  Знак : Строка;
 Перем Код : Целое;
  Для Код = 1 .. 256 Цикл
   Знак = Символ(Код);
   Трассировка(Знак);
  Конец;
Конец;
```

Примечание

При использовании данной функции с кодом 13 [chr(13) - вставляет в строку символ перевода строки] надо обратить внимание на следующее: если такой символ должен быть первым в строке, то ему должна предшествовать пустая строка.

Пример:

```
S : String = "" + chr(13) + "string..." + chr(13) + " ... ";
```

Функция Сло

Ж Б К

Синоним:

Slo

Формат описания:

Сло (Выражение : Ц | Ч) : С;

Входные параметры:

Выражение — выражение целого или числового типа

Назначение:

Возвращает строку, в которой заданная выражением денежная сумма написана прописью на русском языке. В качестве копеек берутся десятые и сотые доли числа. Если учет ведется только в рублях, то копейки не выдаются.

Пример:

```
Сумма = Сло (25.211 ); -- Сумма = "Двадцать пять рублей 21 копейка"
```

Функция Сло_

Ж Б К

Синоним:

Slo_

Формат описания:

Сло_ (Выражение : Ц | Ч [; Язык : С]) : С;

Входные параметры:

Выражение — выражение целого или числового типа

Язык — язык, на котором должен быть записан результат

Назначение:

Возвращает строку, в которой указанная денежная сумма написана прописью на заданном языке. Если параметр *Язык* опущен, то используется язык по умолчанию, заданный в файле прописей текущего плана бухгалтерии.

Примеры:

```
Сумма = Сло_ (25.211); -- Сумма = "Двадцать пять рублей 21 копейка"
Сумма = Сло_ (25.211,' '); -- Сумма = "Двадцать пять рублей 21 копейка"
Сумма = Сло_ (25.211,'USD'); -- Сумма = "Twenty fife dollars 21 cent"
```

Функция СловоВСтроке

Ж Б ф

Синоним:

WordInString

Формат описания:

СловоВСтроке (Слово : С; Текст : С; Разделитель : С) : С;

Входные параметры:

Слово — искомое слово

Текст — любой текст

Разделитель — разделитель слов в тексте

Назначение:

Возвращает "ИСТИНА", если заданное слово есть в тексте через заданный разделитель.

Пример:

```
var S1 : Logical;
  S1 = WordInString("один", "один, два,три", ",");
-- TRUE
```

Функция Соотв

Синоним:

Match

Формат описания:

Соотв (Образец : С; Маска : С) : Л;

Входные параметры:

Образец — любое числовое выражение

Маска — маска

Назначение:

Возвращает истину, если введенный образец соответствует заданной маске.

Пример:

```
Лог = Соотв ( "68.НДС", "68!"); -- Лог = Истина
```

Функция Стр

Синоним:

Str

Формат описания:

Стр (Выражение : Ц | Ч [; ЧислоЗн : Ц]) : С;

Входные параметры:

Выражение — выражение целого или числового типа

ЧислоЗн — число знаков после запятой

Назначение:

Преобразует число к его строковому представлению с заданным числом знаков после запятой. Если второй параметр пропущен, то он считается равным нулю.

Примечание:

Для обратного преобразования строкового представления числа к целому или числовому используются функции **Цел** и **Окр**.

Примеры:

```
Строка = Стр (25.211 ); -- Строка = “25”
Строка = Стр (25.211, 2); -- Строка = “25.21”
```

Функция Стр16

Ж Б Ф

Синоним:

Str16

Формат описания:

Стр16 (Выражение : Ц | Ч [; ЧислоЗн : Ц]) : С;

Входные параметры:

Выражение — выражение целого или числового типа

ЧислоЗн — задает нужное число знаков

Назначение:

Преобразует целое число к его 16-тиричному представлению с заданным числом знаков. Если второй параметр пропущен, то он считается равным 8.

Примеры:

```
Строка = СТР16(511) – дает 000001FF
Строка = СТР16(511,2) – дает FF
Строка = СТР16(511,10) – дает 00000001FF
```

Функция Удалить

Ж Б Ф

Синоним:

Delete

Формат описания:

Удалить (Текст : С; Индекс : Ц; Длина : Ц) : С;

Входные параметры:

Текст — любой текст

Индекс — позиция в тексте, с которой будет производиться удаление подстроки

Длина — длина подстроки

Назначение:

Удаляет из текста подстроку заданной длины, начиная с позиции *Индекс*.

Пример:

```
var S1 : String;
  S1 = Delete("Двадцать один", 9, 1);
-- S1 содержит "Двадцатьодин"
```

Функция Формат

Синоним:

Format

Формат описания:

Формат (ФорматСтрока : С; Аргумент1 [,..., АргументN] : Ц | Ч | С) : С;

Входные параметры:

ФорматСтрока — строка форматирования

Аргумент1,..., АргументN — аргументы форматирования целого, вещественного или строкового типа.

Строка форматирования содержит 2 типа объектов простые символы и спецификаторы формата. Простые символы копируются дословно в результирующую строку.

Спецификаторы формата приносят аргументы от списка параметров и применяют форматирование к ним.

Спецификаторы формата могут принимат следующую форму:

"%" [index ":"] ["-"] [width] ["." prec] type

Спецификатор формата начинается с "%". После "%" следует в порядке:

- дополнительный спецификатор индекса аргумента, [index ":"]
- дополнительный левый индикатор выравнивания, ["-"]
- дополнительный спецификатор ширины, [width]
- дополнительный спецификатор точности, ["." prec]
- конверсионный символ типа, type.

Следующая таблица обобщает возможные значения для конверсионного символ типа, type:

d — десятичное число. Аргументом должна быть значение целого числа. Значение преобразована к строка десятичных цифр. Если строка формата содержит спецификатор точности, это указывает, что получающаяся строка должна содержать по крайней мере указанное число цифр; если значение имеет меньше цифр, получающаяся строка лево-дополнена с нолями.

e — научный. Аргументом должна быть значение с плавающей запятой. Значение преобразована к строка формы "-d.ddd… E+ddd". Получающаяся строка начинается с минус признак, если число отрицательно. Одна цифра всегда предшествует десятичному пункту. Общее количество цифр в получающейся строка (включая один перед десятичным пунктом) дается спецификатором точности в строка формата - точность по умолчанию 15 принята, не присутствует никакой спецификатор точности. "E" символ образца в получающейся строка всегда сопровождается плюс или минус признак и по крайней мере три цифры.

f — фиксированный. Аргументом должна быть значение с плавающей запятой. Значение преобразована к строка формы "-ddd.ddd …". Получающаяся строка начинается с минус признак, если число отрицательно. Число цифр после десятичного пункта дается спецификатором точности в строка формата - по умолчанию 2 десятичных цифр, если не присутствует никакой спецификатор точности.

g — основной. Аргументом должна быть значение с плавающей запятой. Значение преобразована к самой короткой десятичной строка, используя установленный или научный формат. Число существенных цифр в получающейся строка дается спецификатором точности в строка формата - точность по умолчанию 15 принята, если не присутствует никакой спецификатор точности. Тянущиеся ноли удалены от получающейся вереницы, и десятичный пункт кажется только если необходимым. Установленный формат пункта использований получающейся строке, если число цифр налево от десятичного пункта в ценности меньше чем или равно указанной точности, и если значение больше чем или равна 0.00001. Иначе получающаяся строка использует научный формат.

n — число. Аргументом должна быть значение с плавающей запятой. Значение преобразована к строке формы "-d, ddd, ddd.ddd …". Формат "n" соответствует формату "f", за исключением того, что получающаяся строка содержит тысячу сепараторов.

m — деньги. Аргументом должна быть значение с плавающей запятой. Значение преобразована к строке, которая представляет количество валюты. Преобразованием управляет от Формата Валюты в Международной секции Пульта управления Windows. Если вереница формата содержит спецификатор точности, это отвергает значение, данную глобальной настройке.

s — строка. Аргументом должен быть строка. Строка вставлены вместо спецификатора формата. Спецификатор точности, если

указан в строке формата, определяет максимальную длину получающейся строки. Если аргумент - строка, которая более длинна чем этот максимум, строка обрезана.

x — шестнадцатеричный. Аргументом должна быть значение целого числа. Значение преобразована к строке шестнадцатеричных цифр. Если строка формата содержит спецификатор точности, это указывает, что получающаяся строка должна содержать по крайней мере указанное число цифр; если ценность имеет меньше цифр, получающаяся строка лево-дополнена с нолями.

Конверсионные символы могут быть определены в прописных буквах так же как в строчных буквах — оба производят те же самые результаты.

Для всех форматов с плавающей запятой, использовали фактические символы, поскольку десятичное число и тысяча сепараторов получены от глобальных настроек.

Индекс (index), ширина (width), и спецификаторы точности (prec) могут быть определены, непосредственно используя десятичную строку цифры (например "%10d"), или косвенно используя символ звездочки (например "% *.*f").

Используя звездочку, следующий аргумент в списке параметров (который должен быть значением целого числа) становится значением, которая фактически используется.

Например,

```
Format("% *.*f", 8, 2, 123.456)
```

является тем же самым как:

```
Format("%8.2f", 123.456).
```

Спецификатор ширины устанавливает минимальную полевую ширину для преобразования. Если получающаяся строка короче чем минимальная полевая ширина, это дополнено с пробелами, чтобы увеличить полевую ширину. По умолчанию должен оправдать правом результат, добавляя пробелы перед значением, но если спецификатор формата содержит индикатор лево-оправдания ("-" символ, предшествующий спецификатору ширины), результат лево-оправдан, добавляя пробелы после значения.

Спецификатор индекса устанавливает текущий индекс списка параметров в указанную значение. Индекс первого аргумента в списке параметров — 0. Используя спецификаторы индекса, возможно форматировать тот же самый аргумент многократные времена.

Назначение:

Функция для форматирования строки.

Пример:

```
Format("%d %d %0:d %1:d", 10, 20);
-- вернет строку '10 20 10 20.
```

Бухгалтерские функции

Функция Автообработка

Б

Синоним:

Autorefresh

Формат описания:

Автообработка : Л;

Назначение:

Возвращает текущее значение флага **Автообработка** в диалоге "Прочие настройки". При значении "ИСТИНА" режим автообработки включен, при значении "ЛОЖЬ" — выключен.

Пример:

```
ПРОЦ Р1;
   -- Определяем текущее значение флага Автообработка
   If (Автообработка=Истина) :
      Сообщение ("Установлен режим автообработки");
    Истина :
      Сообщение ("Режим автообработки выключен");
   Конец;
Конец;
```

Функция БазоваяВалюта

Ж Б

Синоним:

BaseCurrency

Формат описания:

БазоваяВалюта : С;

Назначение:

Возвращает строку с кратким названием базовой валюты системы, например, "РУБ".

Пример:

```
Проц Р1(Объект : трока);
 Перем Курс:Число;
 Перем КурсВал:Строка;
 Для Валюты  КурсВал = "*"  Цикл
```

```
Если (Бол(КурсВал)<>Бол(БазоваяВалюта)):
  Курс=Общ_Пер(Курс. + КурсВал, 24.11.04);
    Трассировка("Курс " + КурсВал + " = " + Стр(Курс,2));
  Илсе;
Лкиц;
Конец;
```

Функция Бал_Пров_Нул

Б

Синоним:

Bal_Check_Nul

Формат описания:

Бал_Пров_Нул (УслВалАнал : С; Дата : Д [; Коэф : Ч]) : Ч;

Входные параметры:

УслВалАнал — условие отбора на валюту и/или аналитику, записанное в кавычках

Дата — дата составления баланса

Коэф — масштабирующий коэффициент (необязательный параметр)

Назначение:

Возвращает остаток нулевого (“00”) счета или группы счетов, заданных условием отбора, на указанную дату. В случае неравенства остатка нулю выдается сообщение об ошибке. Эта функция, наряду с функцией **Бал_Строка**, позволяет не генерировать бланк В, используемый для построения баланса.

Пример:

```
Проц Р1
 Перем SNULL, FNULL :Число;
 SNULL = Бал_Пров_Нул ("USD{ОС.1}", 01.01.99, 1000.0);
 FNULL = Бал_Пров_Нул ("USD{ОС.1}", 01.07.99, 1000.0);
Конец;
```

Функция Бал_Строка

Б

Синоним:

Bal_Line

Формат описания:

Бал_Строка (УслВалАнал : С; БалСтр : С; Дата : Д
[; Коэф : Ч [; Валюта : С]]) : Ч;

Входные параметры:

УслВалАнал — условие отбора на валюту и/или аналитику, записанное в кавычках

БалСтр — имя строки баланса

Дата — дата составления баланса

Коэф — масштабирующий коэффициент (необязательный параметр)

Валюта — валюта, в которой получается результат (необязательный параметр)

Назначение:

Возвращает остаток счетов, заданных условиями отбора аналитических признаков и валют, на указанную дату, относящихся к заданной строке баланса. Эта функция, наряду с функцией **Бал_Пров_Нул**, позволяет не генерировать бланк B, используемый для построения баланса.

Пример:

```
Проц P1
 Перем S100, F200 :Число;
 S100 = Бал_Строка ("USD{ОС.1}","100",14.08.99,1.0,"USD");
 S100 = Бал_Строка ("{ОС.1}","100",14.08.99,1.0,"РУБ");
 F200 = Бал_Строка ("","100",14.08.99);
Конец;
```

Переменные типа S100, S200 и т.д. используются для вычисления остатка счетов на начало отчетного периода на заданную строку баланса (100, 200); переменные типа F100, F200 и т.д. — на конец отчетного периода. Номер строки баланса указывается в имени переменной после букв S или F. Например, для вычисления остатка на 100-й строке баланса применяются переменные S100 и F100. Эти переменные должны быть описаны в бланке баланса.

Функция ВзятьГруппуСчетов

Б

Синоним:

GetGroupAccounts

Формат описания:

ВзятьГруппуСчетов (УслОтбСч : С ; Счета[ ] : С ; НомерГруппы1 : Ц [, НомерГруппы2 : Ц , ... , НомерГруппыN : Ц) : Ц;

Входные параметры:

УслОтбСч — строка с условием отбора счетов

Счета[ ] — строковый массив, в который будет формироваться список счетов. Массив очищается автоматически

НомерГруппы1, ..., НомерГруппыN — перечисленные через запятую числа, которые содержат номера групп

Назначение:

Формирует в массиве группы счетов, удовлетворяющих условию отбора и списку номеров групп. Возвращает количество элементов в массиве.

Пример:

Есть пять счетов:

188.0302.0010000.239.1.101.02.310

188.0302.0010000.239.1.101.02.410

188.0302.0010000.239.1.101.07.410

188.0312.0810000.220.1.503.01.820

188.0312.0810000.220.1.503.01.810

```
var vAccGrp[]: String;
var vCount: Integer;
  vCount = GetGroupAccounts(«188.*», vAcc, 5, 6, 7);
  vCount = Alternate("Группа счетов:", vAccGrp, 25);
   -- В диалоге «Альтернатива» будет три строчки:
   -- 1.101.02
   -- 1.101.07
   -- 1.503.01
```

Функция ВключаетПризнак

Б Ф

Синоним:

IncludesSign

Формат описания:

ВключаетПризнак (Образец : С; ИмяПоля : С) : Л;

Входные параметры:

Образец — строка образец

ИмяПоля — имя (идентификатор) поля картотеки или табличного журнала

Назначение:

Возвращает значение "ИСТИНА", если введенный образец содержится в заданном поле картотеки или табличного журнала.

Примеры:

Если в табличном журнале отбор записей проводится по аналитическим признакам или по типу валюты, то выражение для фильтра будет иметь вид:

```
ВключаетПризнак ("Мат.01", Аналитика); и ВключаетПризнак ("Руб", Валюта);
```

Для того чтобы в картотеке физических лиц оставить клиентов, имеющих имя "Иван", следует установить фильтр вида:

```
ВключаетПризнак ("Иван", ФИО);
```

Функция ВыборЖурнала

Б

Синоним:

PickJournal

Формат описания:

ВыборЖурнала : С;

Назначение:

Открывает список журналов и возвращает имя выбранного журнала или пустую строку — в случае отказа от выбора.

Пример:

```
Жур = ВыборЖурнала;
```

Функция ВыборКоррСчета

Б

Синоним:

PickCorrAccount

Формат описания:

ВыборКоррСчета (Счет : С; ПоКредиту : Л) : С;

Входные параметры:

Счет — имя счета

ПоКредиту — логическое выражение, равное значению "ИСТИНА" для выбора счетов, корреспондирующих с заданным *по кредиту*, и равное значению "ЛОЖЬ" для выбора счетов, корреспондирующих с заданным *по дебету*

Назначение:

Открывает на экране список счетов, корреспондирующих с заданным по дебету или по кредиту, и возвращает идентификатор выбранного счета — в случае отказа от выбора.

Пример:

```
-- Выбираем счет среди счетов, корреспондирующих с 51 счетом по дебету
Счет = ВыборКоррСчета ('51',ИСТИНА);
```

Функция ВыборПризнака

Б

Синоним:

AnaElection

Формат описания:

ВыборПризнака ([УслПризн: С [; ТолькоСправочники: Л [; ВыбратьГруппу: Л [; Позиция : С]]]) : С;

Входные параметры:

УслПризн — условие отбора аналитических признаков

ТолькоСправочники — логическое выражение по умолчанию равное значению "ЛОЖЬ". Если указано значение "ИСТИНА", то в списке аналитических признаков будут присутствовать только аналитические справочники по условию отбора один из которых можно будет выбрать.

ВыбратьГруппу — логический параметр, если его значение равно "ИСТИНА", то из диалога аналитики можно выбрать как сам признак так и группу. Если — "ЛОЖЬ", то только признак. Если параметр не указан, то его значение интерпретируется как "ЛОЖЬ". Если выбрана группа аналитики, то в результирующем значении будет содержаться точка в конце строки.

Позиция — строковый параметр, позволяющий спозиционироваться на указанный признак или группу признаков. Если параметр не задан, то он по умолчанию принимает значение " " и позиционирования не происходит.

Назначение:

Открывает список аналитических признаков, удовлетворяющих введенному условию отбора, и возвращает идентификатор выбранного признака или пустую строку — в случае отказа от выбора.

Примеры:

```
Пр = ВыборПризнака ('*5');
-- Выбираем признак среди тех, которые оканчиваются на 5
```

```
Glossary = ВыборПризнака(«*», True);
-- Открывает список всех аналитических справочников

Glossary = ВыборПризнака(«П*», True);
-- Открывает список аналитических справочников начинающихся с «П»
```

Функция ВыборСчета

Б

Синоним:

AccountElection

Формат описания:

ВыборСчета ([УслСчет : С [; ВыборГруппы : Л [; Позиция : С]]]) : С;

Входные параметры:

УслСчет — условие отбора счетов

ВыборГруппы — логический параметр, позволяющий выбрать группу или счет. Если параметр не задан, то он по умолчанию принимает значение “ЛОЖЬ”, в этом случае из диалога можно выбрать только счет. Если задано значение “ИСТИНА”, то разрешается выбрать как сам счет, так и группу счетов.

Позиция — строковый параметр, позволяющий спозиционироваться на указанный счет или группу счетов. Если параметр не задан, то он по умолчанию принимает значение “ ” и позиционирования не происходит. Если выбрана группа счетов, то в результирующем значении функции будет содержаться точка в конце строки.

Назначение:

Открывает список счетов, удовлетворяющих введенному условию отбора, и возвращает идентификатор выбранного счета — в случае отказа от выбора.

Пример:

```
-- Выбираем счет среди счетов, начинающихся с 5
Счет = ВыборСчета ('5*');
```

Функция Единица

Б

Синоним:

Unit

Формат описания:

Единица (ИмяПризнака : С) : С;

Входные параметры:

ИмяПризнака — имя аналитического признака (простой идентификатор) или группы признаков (квалифицированный идентификатор)

Назначение:

Возвращает имя количественной единицы для аналитического признака или группы признаков.

Пример:

```
ПРОЦ Р1;
 ПЕРЕМ ИмяЕд, Признак : Строка;
 -- Получаем имя признака
 Признак = ВыборПризнака;
 -- Получаем имя единицы
 ИмяЕд = Единица (Признак);
 -- Выводим сообщение на экран
 Сообщение (Признак + ' измеряются в ' + ИмяЕд);
Конец;
```

Функция ЕстьПроводки

Б

Синоним:

ExistsTransacton

Формат описания:

ЕстьПроводки (Условие : С; ДатаНач : Д, ДатаКон : Д) : Л;

Входные параметры:

Условие — строка с условием отбора проводок

ДатаНач, *ДатаКон* — даты начала и конца периода

Назначение:

Возвращает значение "ИСТИНА", если существуют проводки удовлетворяющие условию отбора за указанный период.

Пример:

```
ЕстьПроводки ("20{П.1}",01.01.05, 01.04.05);
```

Процедура ЗадатьАвтообработку

Б

Синоним:

ToggleAutorefresh

Формат описания:

ЗадатьАвтообработку (НовоеЗначение : Л);

Входные параметры:

НовоеЗначение — значение флага **Автообработка**

Назначение:

Задает новое значение флага **Автообработка** в диалоге "Прочие настройки". Если задано значение "ИСТИНА", то режим автообработки включен, иначе — выключен.

Пример:

```
Если (Автообработка=ЛОЖЬ);
   ЗадатьАвтообработку (ИСТИНА);
Конец;
Сообщение ("Установлен режим автообработки");
```

Процедура ЗадатьОбработкуРеп

Б

Синоним:

ToggleRefreshRep

Формат описания:

ЗадатьОбработкуРеп (НовоеЗначение : Л);

Входные параметры:

НовоеЗначение — значение флага значение флага **Обязательная обработка перед отчетами и раскрытием**

Назначение:

Задает новое значение флага **Обязательная обработка перед отчетами и раскрытием** в диалоге "Прочие настройки". При значении "ИСТИНА" данный режим включен, иначе — выключен.

Пример:

```
Проц P1(Объект : Строка);
  If not ОбработкаРеп :
```

```
    ЗадатьОбработкуРеп(Истина);
  end;
   Сообщение(«Установлен режим обязательной обработки перед отчетами и
     раскрытием»);
end;
```

Процедура ЗакрытьПлан

Б

Синоним:

ClosePlan

Формат описания:

ЗакрытьПлан;

Назначение:

Закрывает текущий план бухгалтерии.

Пример:

```
ПРОЦ ПоОткрытию (Отправитель : Строка);
    ПЕРЕМ Код, Рез : Целое;
    Рез = Ввод (Код, “Введите пароль”) ;
    Если ( Рез <> кмдОк) :
        ЗакрытьПлан;
   Истина :
        Если (Код <> КодБланка) :
            ЗакрытьПлан;
        Конец;
    Конец;
Конец;
```

Функция ИмяЖурнала

Б

Синоним:

JournalName

Формат описания:

ИмяЖурнала (НомерЖурнала : Ц) : С;

Входные параметры:

НомерЖурнала — номер журнала в списке журналов текущей бухгалтерии

Назначение:

По заданному номеру в списке журналов возвращает:

- имя файла с расширением JUR (для текстового журнала);
- имя jdf-файла (для журнала-картотеки);
- имя раздела (для табличного журнала);
- имя файла с переменными (для несуществующих журналов).

Примечания:

1. Нумерация журналов в списке журналов начинается с нуля.
2. Данная функция является обратной к функции **НомерЖурнала**.

Пример:

```
Проц Р1;
    Перем Строка :Строка;
    Перем Колич, Номер :Целое;
    Колич = ЧислоЖурналов;
    Для Номер = 0 .. Колич -1 Цикл
        Строка = " ";
        Если ТипЖурнала(ИмяЖурнала(Номер)) = "КАРТ":
            Строка = КартотекаЖурнала(ИмяЖурнала(Номер));
        Конец;
        Трассировка(Стр(Номер)+" - "ИмяЖурнала(Номер)+" - "+Строка);
    Конец;
Конец;
```

Функция Корреспонденция

Ж Б

Синоним:

Correspond

Формат описания:

Корреспонденция (ИмяСчета1 : С; ИмяСчета2 : С) : Л;

Входные параметры:

ИмяСчета1, ИмяСчета2 — имена счетов

Назначение:

Возвращает значение "ИСТИНА", если заданные счета являются корреспондирующими.

Пример:

```
Если не (Корреспонденция("50",КоррСчет)) :
 Сообщение("Счета не могут корреспондировать!");
Конец;
```

Функция МаскаПризнаков

Б

Синоним:

SignsMask

Формат описания:

МаскаПризнаков (Счет : С; Обязат : Л) : С;

Входные параметры:

Счет — имя счета

Обязат — логическое выражение, равное значению "ИСТИНА" для отбора справочников, *обязательных* для заданного счета, и равное значению "ЛОЖЬ" для отбора справочников, *связанных* с ним

Назначение:

Возвращает условие отбора признаков из справочников, обязательных для данного счета или связанных с ним.

Пример:

```
-- Условие отбора признаков, обязательных для счета 60
sAna = МаскаПризнаков ("60",ИСТИНА);
-- Окрытие списка признаков, обязательных для счета 60
sAna = ВыборПризнака (sAna);
```

Функции Наим_Б

Ж Б

Синоним:

Name_B

Формат описания:

Наим_Б (ИмяБланка : С) : С;

Входные параметры:

ИмяБланка — имя бланка

Назначение:

Возвращает наименование бланка по его имени, пробел или пустую строку.

Примечания:

1. Если в качестве параметра задано имя бланка с перечислением всех групп, в которые он входит, то функция по уникальному

имени признака возвращает его наименование, если оно указано в дереве бланков, иначе — пустую строку.

2. Если указан несуществующий бланк, то функция возвращает пробел.

Пример:

```
ПРОЦ Р1;
 Перем Назв : Строка;
 Назв = Наим_Б ("Документы.Банк.Поручн"); -- Назв = "Платежное поручение"
 Назв = Наим_Б ("Поручн"); -- Назв = " ", функция возвращает пробел, т.к.
 -- неправильно задано имя бланка
 Назв = Наим_Б ("Тесты.ПользовБланк"); -- Назв = "", функция возвращает
 -- пустую строку, т.к. данный бланк не имеет наименования
Конец;
```

Функция Наим_В

Ж Б

Синоним:

Name_C

Формат описания:

Наим_В (Вал : С) : С;

Входные параметры:

Вал — краткое описание (код) валюты из текущей структуры учёта

Назначение:

Возвращает наименование валюты по её коду. Фактически, это комментарий в строке с кодом валюты в структуре учёта.

Пример:

```
Trace(Name_C("Руб"));
-- Выводит «Российский рубль», т.к. в структуре учёта есть строка «Руб  !2
-- Российский рубль»
```

Функция Наим_Е

Ж Б

Синоним:

Name_U

Формат описания:

Наим_Е (ИмяЕд : С) : С;

Входные параметры:

ИмяЕд — имя единицы измерения

Назначение:

Возвращает наименование единицы измерения по ее имени.

Примечание:

Функция **Наим_Е** используется для совместимости с данными предыдущих версий и возвращает наименования единиц, описанных в разделе ЕДИНИЦЫ структуры учета.

Пример:

```
Назв = Наим_Е ("Бут33"); -- Назв = "Бутылки по 0,33 литра"
```

Функция Наим_О

Ж Б

Синоним:

Name_O

Формат описания:

Наим_О (ИмяОп : С) : С;

Входные параметры:

ИмяОп — имя операции

Назначение:

Возвращает наименование операции по ее уникальному имени, пустую строку или пробел.

Примечания:

1. Если в качестве параметра задано имя типовой операции с перечислением всех групп, в которые она входит, то по имени операции функция возвращает ее наименование, если оно указано в перечне типовых операций, иначе — пустую строку.
2. Если указана несуществующая операция, то функция возвращает пробел.

Пример:

```
ПРОЦ Р1;
 Перем Назв : Строка;
 Назв = Наим_О ("К60.Д08.4"); -- Назв="ОПРИХОДОВАНИЕ капиталовложений на ОС"
 Назв = Наим_О ("РАБ.ОФ.ОСТАТКИ"); -- Назв = "", операция не имеет
 -- наименование, поэтому функция возвращает пустую строку
 Назв =  Наим_О ("ИЗНОС"); -- Назв=" ", неправильно задано имя операция,
 -- функция возвращает пробел
Конец;
```

Функция Наим_П

Б

Синоним:

Name_A

Формат описания:

Наим_П (ИмяПр : С) : С;

Входные параметры:

ИмяПр — имя аналитического признака или пустая строка

Назначение:

Возвращает наименование аналитического признака, пустую строку или пробел.

Примечания:

1. Если в качестве параметра задано имя аналитического признака с перечислением всех групп, в которые он входит, то функция по уникальному имени признака возвращает его наименование, если оно существует в списке признаков, иначе — пустую строку.
2. Если в качестве параметра задан несуществующий признак, то функция возвращает пробел. Это можно использовать для определения наличия данного признака в текущем плане бухгалтерии.
3. Если в функцию передать группу аналитического признака с точкой на конце, то функция вернет наименование для группы, если группа существует в дереве признаков, иначе — пустую строку. Если указана несуществующая группа аналитического признака, то функция фозвращает пробел.

Примеры:

```
ПРОЦ Р1;
 ПЕРЕМ Назв :Строка;
 Назв = Наим_П ("ПОСТ.Ю.06"); -- Назв = "ЗАО "Поле чудес"
 Назв = Наим_П ("ПОСТ.1");  -- Назв = " ", т.к. признак отсутствует
                            -- в текущем плане бухгалтерии
 Назв = Наим_П ("НДС.20");  -- Назв = "", т.к. у данного признака
                            -- отсутствует наименование
Конец;
```

Функция Наим_С

Ж Б

Синоним:

Name_S

Формат описания:

Наим_С (ИмяСчета : С) : С;

Входные параметры:

ИмяСчета — имя счета

Назначение:

Возвращает наименование счета по его уникальному имени, пустую строку или пробел.

Примечания:

1. Если в качестве параметра задано имя счета с перечислением всех групп, в которые он входит, то функция по его уникальному имени возвращает наименование, если оно указано в плане счетов, иначе — пустую строку.
2. Если указан несуществующий счет, то функция возвращает пробел.

Примеры:

```
ПРОЦ Р1;
 Перем Назв,Имя :Строка;
 Имя = ВыборСчета ("01*"); -- пользователь выбрал счет 01.1
 Назв = Наим_С (Имя);
 -- Назв = "Земельные участки и объекты природопользования"
 Назв = Наим_С ("НДСПО"); -- Назв = "", отсутствует наименование, поэтому
 -- функция возвращает пустую строку
 Назв = Наим_С ("01.4"); -- Назв = " ", в плане счетов такого счета нет,
 -- поэтому функция возвращает пробел
Конец;
```

Функция НомерЖурнала

Б

Синоним:

JournalNumber

Формат описания:

НомерЖурнала (ИмяЖурнала : С) : Ц;

Входные параметры:

ИмяЖурнала — имя файла с журналом (полное или сокращенное)

Назначение:

Возвращает целое число (номер журнала в списке журналов) или -1, если журнал не найден.

Примечание:

Если в качестве имени файла с журналом задано сокращенное имя (без указания пути), то файл ищется в каталоге плана бухгалтерии.

Примеры:

```
Ном1 = НомерЖурнала ("Расчет итогов квартала");
Ном2 = НомерЖурнала ("c:\Tbw65p\Wrk\Бухгалтерия\Приход товара.jdf");
```

Функция Обо

Ж Б

Синоним:

Tur

Формат описания:

Обо (УсловиеОтбора; Дата : Д [; Показатель : С]) : Ч;

Входные параметры:

УсловиеОтбора — условие отбора

Дата — начальная дата

Показатель — название показателя

Назначение:

Вычисляет оборот счета или группы счетов, заданных введенным условием отбора, за период от задаваемой начальной даты до даты последней проводки в журнале (если функция используется в операциях) или до системной даты (если она используется в бланке). Необязательный параметр *Показатель* определяет, в каком показателе должен быть получен результат.

Примечания:

1. Если в условии отбора указано несколько валют (например, "`USD|Руб"), то параметр *Показатель* обязателен и сумма вычисляется в соответствии с заданным показателем.
2. Если в условии отбора задана одна валюта, то параметр *Показатель* может отсутствовать, а его значение выбирается из условия отбора. Например, если в условии отбора записано "USD", то результат будет вычисляться в долларах.

3. Если в параметре *Показатель* задана пустая строка, то сумма вычисляется в базовой валюте. При наличии непустого значения в параметре *Показатель* возможны следующие варианты его задания:

- имя валюты — сумма вычисляется в иностранной валюте;
- имя твердого эквивалента — сумма вычисляется в твердом эквиваленте;
- "КОЛ" — вычисляется количественная сумма в заданных единицах измерения.

Пример:

```
Сумма = Обо (70{Ф.001}, 01.01.97);
-- переменная 'Сумма' равна обороту 70 счета по признаку Ф.001 с начала
-- года до даты последней проводки в журнале
```

Функция Оборот

Ж Б

Синоним:

Turn

Формат описания:

Оборот (УсловиеОтбора : С; ДатаНач : Д; ДатаКон : Д [; Показатель : С]) : Ч;

Входные параметры:

УсловиеОтбора — условие отбора

ДатаНач — дата начала периода

ДатаКон — дата конца периода

Показатель — название показателя

Назначение:

Вычисляет оборот счета или группы счетов, заданных условием отбора, за период от начальной до конечной даты. Необязательный параметр *Показатель* определяет, в каком показателе должен быть получен результат.

Примечания:

1. Если в условии отбора указано несколько валют (например, "`USD|Руб"), то параметр *Показатель* обязателен и сумма вычисляется в соответствии с заданным показателем.
2. Если в условии отбора задана одна валюта, то параметр *Показатель* может отсутствовать, а его значение выбирается из условия отбора. Например, если в условии отбора записано "USD", то результат будет вычисляться в долларах.

3. Если в параметре *Показатель* задана пустая строка, то сумма вычисляется в базовой валюте. При наличии непустого значения в параметре *Показатель* возможны следующие варианты его задания:

- имя валюты — сумма вычисляется в иностранной валюте;
- имя твердого эквивалента — сумма вычисляется в твердом эквиваленте;
- "КОЛ" — вычисляется количественная сумма в заданных единицах измерения.

Пример:

```
Сумма = Оборот ( "20{П.1}", 01.01.97, 01.01.98);
-- Переменная Сумма равна обороту 20 счета с учетом признака П.1 за период
-- от 01.01.97 по 31.12.97 включительно
```

Процедура Обработать

Б

Синоним:

Refresh

Формат описания:

Обработать [(ОбработатьВсе : Л [; КонечнаяДата : Д])];

Входные параметры:

ОбработатьВсе — логический параметр, определяющий режим обработки: при значении "ИСТИНА" производится обработка *всех* файлов текущего плана бухгалтерии, при значении "ЛОЖЬ" — обрабатываются только *измененные* файлы текущего плана бухгалтерии, влияющие на построение отчетов и на результаты вычисления остатков и оборотов (по умолчанию, значение параметра равно "ЛОЖЬ")

КонечнаяДата — дата, *до* которой производится обработка проводок от начальной даты, не включая ее

Назначение:

Компилирует журналы хозяйственных операций.

Примечание:

Если дата не указана, то при значении параметра *Обработать Все*="ИСТИНА" производится обработка *всех* проводок во *всех* журналах, иначе — только в *измененных*.

Пример:

```
ПРОЦ  Р1;
```

```
  var oldRefr :l;
  oldRefr = АВТООБРАБОТКА;
 -- сохраняем старое значение флага "Режим автообработки"
  Попытка
     ЗадатьАвтообработку(False);
 -- снимаем автообработку, чтобы не компилировать журналы после изменения
 -- каждой проводки
     Трассировка(Str(Остаток("50",01.02.99))); -- печатаем остаток
     Вставка("Расчет итогов квартала.jur",31.01.1999, " : 1111 50 51");
     -- меняем журнал
     Трассировка(Str(Остаток("50",01.02.99))); -- печатаем остаток,
     -- он прежний
     Обработать(true,02.02.99); -- обрабатываем журналы, включая нашу
     -- дату, но не до сегодняшней даты (в целях экономии времени)
     Trace(Str(Остаток("50",01.02.99))); -- печатаем изменившийся остаток
  Окончание
  ЗадатьАвтообработку(oldRefr); -- восстанавливаем значение флага
 -- "Режим автообработки"
  end;
Конец;
```

Функция ОбработкаРеп

Б

Синоним:

RefreshRep

Формат описания:

ОбработкаРеп : Л;

Назначение:

Возвращает значение, соответствующее текущему состоянию флага Обязательная обработка перед отчетами и раскрытием в диалоге «Прочие настройки». При значении "ИСТИНА" данный режим включен, при значении "ЛОЖЬ" — выключен.

Пример:

```
Proc P1(Объект : Строка);
-- Определяем текущее состояние флага «Автообработка»
 If ОбработкаРеп :
   Сообщение(«Установлен режим обязательной обработки перед отчетами и
      раскрытием»);
 else
   Сообщение(«Режим обязательной обработки перед отчетами и раскрытием
      выключен»);
 end;
end;
```

Функция Общ_Пер

Ж Б

Синоним:

Com_Var

Формат описания:

Общ_Пер (ИмяПерем : С [; Дата : Д]) : Ч;

Входные параметры:

ИмяПерем — имя общей переменной, имеющей числовой тип

Дата — дата, на которую должно быть взято значение переменной (необязательный параметр)

Назначение:

Возвращает значение общей переменной числового типа с именем *ИмяПерем*. (К общим переменным числового типа можно отнести, например, курсы валют, ставки налогов и т.п.)

Примечание:

Если параметр *Дата* не задан, то значение берется согласно дате последней проводки в журнале (если функция используется в операциях), или системной дате (если она используется в бланках).

Пример:

```
КурсДоллара = Общ_Пер ("Курс.USD", 01.09.98); -- КурсДоллара = 9.3301
```

Функция Общ_Пер_С

Ж Б

Синоним:

Com_Var_S

Формат описания:

Общ_Пер_С (ИмяПерем : С [; Дата : Д]) : С;

Входные параметры:

ИмяПерем — имя общей переменной, имеющей строковый тип

Дата — дата, на которую должно быть взято значение переменной (необязательный параметр)

Назначение:

Возвращает значение общей переменной строкового типа с именем *ИмяПерем*. (К общим переменным строкового типа можно отнести,

например, названия отчетных периодов: “За 1-й квартал”, “За 1-е полугодие”, “За 9 месяцев”, “За год”.)

Примечание:

Если параметр *Дата* не задан, то значение берется согласно дате последней проводки в журнале (если функция используется в операциях), или системной дате (если она используется в бланках).

Пример:

```
ОтчПериод = Общ_Пер_С (“Период”, 31.09.98); -- ОтчПериод = “За 9 месяцев”
```

Функция Общ_Пер_Файл

Синоним:

Com_Var_File

Формат описания:

Общ_Пер_Файл (ИмяПерем : С) : С;

Входные параметры:

ИмяПерем — строка с именем общей переменной

Назначение:

Возвращает строку с именем файла, в котором описана общая переменная.

Пример:

```
НалогСПродаж = Общ_Пер_Файл (“НалогСПродаж”);
-- “Общ.var”
```

Функция Обязат

Синоним:

Oblig

Формат описания:

Обязат (ИмяПризнака : С; ИмяСчета : С) : Л;

Входные параметры:

ИмяПризнака, ИмяСчета — имя аналитического признака и имя счета, соответственно

Назначение:

Проверяет, является ли заданный аналитический признак обязательным для заданного счета, и возвращает значение “ИСТИНА” при положительном результате проверки.

Пример:

```
Если (не Обязат ('Б', '68.НДС') ) :
 Сообщение ('Признак Б не обязательный для счета 68.НДС');
Конец;
```

Функция Ост

Ж Б

Синоним:

Sal

Формат описания:

Ост (УсловиеОтбора [; Показатель : С]) : Ч;

Входные параметры:

УсловиеОтбора — условие отбора

Показатель — название показателя

Назначение:

Вычисляет остаток по группе счетов, заданных условием отбора, на момент совершения операции в журнале, либо на системную дату в описаниях бланков. Необязательный параметр *Показатель* определяет, в каком показателе должен быть получен результат.

Примечания:

1. Если в условии отбора указано несколько валют (например, “`USD|Руб”), то параметр *Показатель* обязателен и сумма вычисляется в соответствии с заданным показателем.
2. Если в условии отбора задана одна валюта, то параметр *Показатель* может отсутствовать, а его значение выбирается из условия отбора. Например, если в условии отбора записано “USD”, то результат будет вычисляться в долларах.
3. Если в параметре *Показатель* задана пустая строка, то сумма вычисляется в базовой валюте. При наличии непустого значения в параметре *Показатель* возможны следующие варианты его задания:

- имя валюты — сумма вычисляется в иностранной валюте;
- имя твердого эквивалента — сумма вычисляется в твердом эквиваленте;

- "КОЛ" — вычисляется количественная сумма в заданных единицах измерения.

Примеры:

```
А = Ост (20{П.1}); -- А = остатку 20 сч. с учетом аналит. признака П.1
А = Ост (68*); -- А = остатку 68 сч. и всех его субъектов
```

См. также:

Функции **Обо, Оборот, Остаток, Пров_*, Разд_*_Ост**

Функция Остаток

Ж Б

Синоним:

Saldo

Формат описания:

Остаток (УсловиеОтбора : С; Дата : Д [; Показатель : С]) : Ч;

Входные параметры:

УсловиеОтбора — условие отбора

Показатель — название показателя

Дата — дата конца периода

Назначение:

Вычисляет остаток по группе счетов, заданных условием отбора, на указанную дату. Необязательный параметр *Показатель* определяет, в каком показателе должен быть получен результат.

Примечания:

1. Если в условии отбора указано несколько валют (например, "`USD|Руб"), то параметр *Показатель* обязателен и сумма вычисляется в соответствии с заданным показателем.
2. Если в условии отбора задана одна валюта, то параметр *Показатель* может отсутствовать, а его значение выбирается из условия отбора. Например, если в условии отбора записано "USD", то результат будет вычисляться в долларах.
3. Если в параметре *Показатель* задана пустая строка, то сумма вычисляется в базовой валюте. При наличии непустого значения в параметре *Показатель* возможны следующие варианты его задания:

- имя валюты — сумма вычисляется в иностранной валюте;
- имя твердого эквивалента — сумма вычисляется в твердом эквиваленте;

- "КОЛ" — вычисляется количественная сумма в заданных единицах измерения.

4. В отличие от функции **Ост,** входной параметр *УсловиеОтбора* является выражением типа Строка и должен заключаться в кавычки.

Примеры:

```
А = Остаток ('20{П.1}',10.05.2000); -- А = остатку 20 сч. с учетом признака П.1
А = Остаток ('68*',10.05.2000); -- А = остатку 68 сч. и всех его субъектов
```

Процедура ОткрытьПлан

Б

Синоним:

OpenPlan

Формат описания:

ОткрытьПлан (ПутьКФайлу : С);

Входные параметры:

ПутьКФайлу — полный путь к файлу с планом бухгалтерии

Назначение:

Открывает заданный файл с планом бухгалтерии.

Пример:

```
Проц СменитьБухгалтерию (Кнопка:Строка);
  ОткрытьПлан ("d:\tbw69p\wrk\пример\менеджер.pro");
Конец;
```

Функция ПолучитьИмяСправ

Б

Синоним:

GetVocName

Формат описания:

ПолучитьИмяСправ (Идентификатор : С) : С;

Входные параметры:

Идентификатор — идентификатор аналитического справочника

Назначение:

Возвращает полное имя аналитического справочника по его идентификатору.

Пример:

```
Проц ТЕСТ
 Перем ИмяСправ : строка;
  ИмяСправ = ПолучитьИмяСправ("ТМЦ");
  Трассировка("Справочник " + ИмяСправ + " " + РольСправочника(ИмяСправ));
Конец
```

Функция ПравильныйПризнак

Б

Синоним:

ValidSign

Формат описания:

ПравильныйПризнак (Стр : С) : Л;

Входные параметры:

Стр — исходное выражение

Назначение:

Проверяет, может ли строковое выражение *Стр* являться идентификатором аналитического признака. При положительном результате возвращает значение "ИСТИНА", иначе — "ЛОЖЬ".

Пример:

```
Проц Р1;
 Перем Идент : Строка; -- введите идентификатор и нажмите кнопку "Ввод"
 Если Не(Ввод(Идент, "Введите идентификатор") = КмдВерно):
  Выход;
 Иначе
  Если (ПравильныйПризнак(Идент)) :
   Сообщение("Идентификатор " + Идент + " соответствует синтаксису!");
  Истина :
   Сообщение("Идентификатор " + Идент + "  не соответствует синтаксису!");
  Илсе;
 Конец;
Конец;
```

Функция ПризнакИспользован

Б

Синоним:

SignUsed

Формат описания:

ПризнакИспользован (ИмяПризнака : С) : Л;

Входные параметры:

ИмяПризнака — имя (идентификатор) аналитического признака

Назначение:

Возвращает значение “ИСТИНА”, если заданный аналитический признак используется в проводках журналов хозяйственных операций.

Пример:

```
Трассировка ("Признак Мат.01"+Если(ПризнакИспользован("Мат.01"),
                "использовался ","не использовался "+"в проводках "));
```

Функции Пров_Деб, Пров_Кре, Пров_Нул

Синонимы:

Check_Deb, Check_Cre, Check_Nul

Формат описания:

Пров_* (УсловиеОтбора : С; Дата : Д [; Показатель : С]) : Ч;

Входные параметры:

УсловиеОтбора — условие отбора

Дата — дата конца периода

Показатель — название показателя

Назначение:

Функции вычисляют остаток счета или группы счетов, заданных условием отбора, на указанную дату; при этом проверяют, является ли остаток дебетовым **(Пров_Деб)**, кредитовым **(Пров_Кре)** или нулевым **(Пров_Нул)**. В случае ошибки выдается соответствующее сообщение, а результат функций будет равен нулю. Необязательный параметр *Показатель* определяет, в каком показателе должен быть получен результат.

Примечания:

1. Если в условии отбора указано несколько валют (например, “`USD|Руб”), то параметр *Показатель* обязателен и сумма вычисляется в соответствии с заданным показателем.
2. Если в условии отбора задана одна валюта, то параметр *Показатель* может отсутствовать, а его значение выбирается из условия отбора. Например, если в условии отбора записано “USD”, то результат будет вычисляться в долларах.
3. Если в параметре *Показатель* задана пустая строка, то сумма вычисляется в базовой валюте. При наличии непустого значения в данном параметре возможны следующие варианты его задания:

- имя валюты — сумма вычисляется в иностранной валюте;
- имя твердого эквивалента — сумма вычисляется в твердом эквиваленте;
- "КОЛ" — вычисляется количественная сумма в заданных единицах измерения.

Пример:

```
0 = Пров_Нул("20{П.1}",01.01.99); -- 0 = остатку 20 счета на 01.01.99
-- если остаток не нулевой, то выдается сообщение об этом и 0 = 0
```

Функции Разд_Деб_Ост, Разд_Кре_Ост

Синонимы:

Sep_Deb_Sal, Sep_Cre_Sal

Формат описания:

Разд_*_Ост (УсловиеОтбора : С; Дата : Д; Группа : С [; Показатель : С]) : Ч;

Входные параметры:

УсловиеОтбора — условие отбора

Показатель — название показателя

Дата — дата конца периода

Группа — название группы аналитических признаков

Назначение:

Вычисляет дебетовый (кредитовый) остаток по группе счетов, заданных условием отбора, разделенный по аналитическим признакам указанной группы или групп, на указанную дату. Необязательный параметр *Показатель* определяет, в каком показателе должен быть получен результат.

Примечания:

1. Если в условии отбора указано несколько валют (например, "`USD|Руб"), то параметр *Показатель* обязателен и сумма вычисляется в соответствии с заданным показателем.
2. Если в условии отбора задана одна валюта, то параметр *Показатель* может отсутствовать, а его значение выбирается из условия отбора. Например, если в условии отбора записано "USD", то результат будет вычисляться в долларах.
3. Если в параметре *Показатель* задана пустая строка, то сумма вычисляется в базовой валюте. При наличии непустого значения в параметре *Показатель* возможны следующие варианты его задания:

- имя валюты — сумма вычисляется в иностранной валюте;
- имя твердого эквивалента — сумма вычисляется в твердом эквиваленте;
- "КОЛ" — вычисляется количественная сумма в заданных единицах измерения.

Пример:

```
А = Разд_Кре_Ост ('60',ОИ.НАЧ,"Поставщик"); -- разделенное сальдо по 60
-- счету по группе "Поставщик" на дату, заданную переменной НАЧ, бланка "ОИ"
```

Функция РольСправочника

Б

Синоним:

VocRole

Формат описания:

РольСправочника (ИмяСправочника : С) : С;

Входные параметры:

ИмяСправочника — имя аналитического справочника

Назначение:

Возвращает экономическую роль аналитического справочника по его имени.

Пример:

```
Трассировка(Справочник+ИмяСправ+ +РольСправочника(ИмяСправ));
```

Функция СверткаЖурналов

Б

Синоним:

TujReport

Формат описания:

СверткаЖурналов (ИмяФайла : С; ДатаН : Д; ДатаК : Д; УслСчетов : С; УслАналит : С; УслСжатия : С; ПризнДеб : С; ПризнКре : С) : Ц;

Входные параметры:

ИмяФайла — имя текстового файла для записи свертки журналов

ДатаН — начальная дата старого (сворачиваемого) периода

ДатаК — начальная дата нового периода

УслСчетов — условие отбора счетов

УслАналит — условие отбора аналитических признаков

УслСжатия — условие сжатия аналитики

ПризнДеб — имя нового признака, заменяющего сжатые признаки, относящиеся к счету дебета

ПризнКре — имя нового признака, заменяющего сжатые признаки, относящиеся к счету кредита

Назначение:

Выполняет свертку журналов по заданным условиям, записывает ее в указанный файл и возвращает число обработанных проводок. Одновременно со сверткой журналов возможно сжатие аналитики.

Примечания:

1. Если файл для записи свертки журналов не пуст, то полученные данные дописываются в конец файла.
2. *Условие сжатия аналитики* представляет собой последовательность масок признаков из одного аналитического справочника, разделенных вертикальной чертой.

   Например, условие “СВ.ОТД1|СВ.ОТД2*” означает, что будут свертываться все группы справочника СВ, которые начинаются с “СВ.ОТД1”, а также с “СВ.ОТД2”, “СВ.ОТД21”, “СВ.ОТД22” и т.д. В результате такой свертки будут объединяться проводки, имеющие признаки СВ.ОТД1.БРИГ1.ИВАНОВ, СВ.ОТД1.БРИГ3.СМИРНОВ и т.д. и совпадающие по счетам, другим признакам и валюте (если она указана в проводках). У объединенной проводки будет стоять новый признак [+/-]СВ.ОТД1.*ПризнДеб* (или *ПризнКре*), где “+”/“-” означает отношение признака к счету дебета или кредита соответственно.

   Уровень, до которого сжимается аналитика, определяется количеством указанных в условии сжатия точек, разделяющих имена групп. Например, если в условии сжатия присутствует только одна точка (“СВ.ОТД1”), то сжатие происходит до групп второго уровня, т.е. новый признак будет выглядеть как “СВ.ОТД1.*ПризнДеб*/*ПризнКре*”. При маске “СВ.ОТД1.*” сжатие выполняется до групп третьего уровня, и новый признак будет записываться как “СВ.ОТД1.БРИГ1.*ПризнДеб*/*ПризнКре*”, “СВ.ОТД1.БРИГ2.*ПризнДеб*/*ПризнКре*” и т.д.
3. Если параметр *УслСжатия* не задан (пустая строка), то свертка журналов выполняется без сжатия аналитики.
4. Если параметры *ПризнДеб* и *ПризнКре* не заданы (пустые строки), то при свертке будут полностью игнорироваться признаки из справочников, упомянутых в параметре *УслСжатия*, и создаваться объединенные проводки без учета этой аналитики.

Пример:

```
NUM = СверткаЖурналов ("JUR1.jur",01.01.98,01.01.99,"","",
      "СВ.ОТД1:СВ.ОТД2*","ПРДЕБ","ПРКРЕ"); -- свертка за 1998 г.
```

Процедура СверткаЖурналовКонец

Б

Синоним:

TujReportEnd

Формат описания:

СверткаЖурналовКонец;

Назначение:

Удаляет временные структуры данных, сформированные процедурой **СверткаЖурналовНачало.**

Пример:

```
СверткаЖурналовНачало
("JUR1.jur",01.01.98,"","","СВ.ОТД1:СВ.ОТД2*","ПРДЕБ","ПРКРЕ");
-- готовит данные для свертки с 01.01.98
...
NUM = СверткаЖурналовСлед (01.01.99); -- свертка за 1998 г.
...
СверткаЖурналовКонец; -- конец свертки
```

Процедура СверткаЖурналовНачало

Б

Синоним:

TujReportBegin

Формат описания:

СверткаЖурналовНачало (ИмяФайла : С; ДатаН : Д; УслСчетов : С; УслАналит : С; УслСжатия : С; ПризнДеб : С; ПризнКре : С);

Входные параметры:

ИмяФайла — имя текстового файла для записи свертки журналов

ДатаН — начальная дата старого (сворачиваемого) периода

УслСчетов — условие отбора счетов

УслАналит — условие отбора аналитических признаков

УслСжатия — условие сжатия аналитики

ПризнДеб — имя нового признака, заменяющего сжатые признаки, относящиеся к счету дебета

ПризнКре — имя нового признака, заменяющего сжатые признаки, относящиеся к счету кредита

Назначение:

Формирует начальные данные для свертки.

Примечания:

См. примечания к функции **СверткаЖурналов**

Пример:

См. пример для процедуры **СверткаЖурналовКонец**

Функция СверткаЖурналовСлед

Б

Синоним:

TujReportNext

Формат описания:

СверткаЖурналовСлед (ДатаК : Д) : Ц;

Назначение:

Выполняет свертку данных на интервале времени [*ДатаН*; *ДатаК*] и возвращает число обработанных проводок. Если это первый вызов функции **СверткаЖурналовСлед**, то *ДатаН* берется из предшествующего вызова процедуры **СверткаЖурналовНачало**. Иначе — *ДатаН* = *ДатаК* из предыдущего вызова **СверткаЖурналовСлед**.

Пример:

См. пример для процедуры **СверткаЖурналовКонец**

Функция Связан

Ж Б

Синоним:

Relat

Формат описания:

Связан (ИмяПризнака : С, ИмяСчета : С) : Л;

Входные параметры:

ИмяПризнака — имя аналитического признака

ИмяСчета — имя счета

Назначение:

Проверяет, связан ли заданный аналитический признак с конкретным счетом. Возвращает значение “ИСТИНА” при положительном результате проверки.

Пример:

```
Проц Р1(Объект : Строка);
 Перем Пр : Строка;
 Если(Не Связан("ПОК.Ю.8", "68.НДС")) :
  Сообщение("Признак ПОК.Ю.8 не связан со счетом 68.НДС");
 Илсе;
Конец;
```

Функция СправИмеетКоличПризнак

Б

Синоним:

HaveVocQuaSign

Формат описания:

СправИмеетКоличПризнак (ИмяСправочника : С) : Л;

Входные параметры:

ИмяСправочника — имя аналитического справочника

Назначение:

Возвращает значение “ИСТИНА”, если справочник имеет количественный учет, “ЛОЖЬ” — если не имеет.

Пример:

```
Proc Проверка;
 var a : Logical;
 a = HaveVocQuaSign ("ТМЦ");
  -- a = True;
end
```

Функция СчетАктивный

Б

Синоним:

IsAccActive

Формат описания:

СчетАктивный (НомерСчета : С);

Входные параметры:

НомерСчета — номер счета

Назначение:

Возвращает значение одной из предопределенных констант (**AccIsActive** — счет активный, **AccIsPassive** — счет пассивный, **AccIsActPas** — счет активно-пассивный, **AccIsZero** — счет нулевой) по заданному номеру счета.

Пример:

```
A = IsAccActive (20_1);
if A = AccIsActive : Message ("Счет активный");
if A = AccIsPassive : Message ("Счет пассивный");
```

Функция СчетБалансовый

Б

Синоним:

AccIsBalance

Формат описания:

СчетБалансовый (НомерСчета : С) : Л;

Входные параметры:

НомерСчета — номер счета

Назначение:

Возвращает значение “TRUE”, если счет с заданным номером — балансовый.

Пример:

```
Трассировка(Счет+НомерСчета+ +Если(СчетБалансовый(НомерСчета),
                                   является ,не является )+балансовым);
```

Функция СчетИспользован

Б

Синоним:

AccountUsed

Формат описания:

СчетИспользован (НомерСчета : С [; ДатаНач : Д; ДатаКон : Д]) : Л;

Входные параметры:

Номер счета — номер счета

ДатаНач, *ДатаКон* — начальная и конечная даты периода, за который проводки проверяются на предмет использования в них счета. Если период не указан, то проверка на наличие счета производится по всем проводкам.

Назначение:

Возвращает значение "Истина", если счет используется в проводках, иначе — "Ложь".

Пример:

```
Проц Р1(Объект:Строка);
 Если СчетИспользован(ТСС.ПОСТ):
   Трассировка(Счет используется в проводках);
 Конец;
 if AccountUsed(ТСС.ПОСТ, 1.1.2004, 1.1.2005):(ТСС.ПОСТ):
    Trace(Счет используется в проводках за период с 1.1.2004, 31.12.2004);
 end;
Конец;
```

Функция ТипЖурнала

Б

Синоним:

JournalType

Формат описания:

ТипЖурнала (ИмяЖурнала : С) : С;

Входные параметры:

ИмяЖурнала — имя файла с журналом (полное или сокращенное)

Назначение:

Возвращает строку с названием типа журнала ("ТЕКСТ", "РАЗД" или "КАРТ") или пустую строку, если журнал не найден.

Примечание:

Если в качестве имени файла с журналом задано сокращенное имя (без указания пути), то файл ищется в каталоге плана бухгалтерии.

Пример:

```
...
jur = ВыборЖурнала;
...
ТипЖур = ТипЖурнала (jur);
...
```

Функция Точность

Б

Синоним:

Accuracy

Формат описания:

Точность [(Имя : С)] : Ц;

Входные параметры:

Имя — имя количественной единицы, валюты, аналитического признака или группы аналитических признаков (с точкой на конце в последнем случае)

Назначение:

Возвращает точность заданного параметра. Если входной параметр не задан, то возвращает точность базовой валюты.

Пример:

```
ПРОЦ Р1;
 ПЕРЕМ Точн : Ц;
 Точн = Точность;
 Сообщение ( 'Точность равна ' + Стр (Точн) );
Конец;
```

Функция ЧислоЖурналов

Б

Синоним:

JournalsCount

Формат описания:

ЧислоЖурналов : Ц;

Назначение:

Возвращает общее число журналов, содержащихся в текущем плане бухгалтерии.

Пример:

```
Колич = ЧислоЖурналов;
```

Календарные функции

Функция Время

Б ф

Синоним:

Time

Формат описания:

Время : Ц;

Назначение:

Возвращает число секунд от начала суток.

Пример:

```
В = Время; -- В = 360060
```

Функция Год

Ж Б ф К

Синоним:

Year

Формат описания:

Год (Дата : Д) : Ц;

Входные параметры:

Дата — некоторая дата

Назначение:

Возвращает по введенной дате порядковый номер ее года с учетом века.

Пример:

```
Г = Год (01.01.99); -- Г = 1999
```

Функция Дат

Ж Б ф К

Синоним:

Dat

Формат описания:

Дат (День : Ц; Мес : Ц; Год : Ц) : Д;

Входные параметры:

День, Мес, Год — день, месяц и год некоторой даты

Назначение:

Возвращает дату по заданным дню, месяцу и году.

Пример:

```
Дата = Дат (1, 1, 1999); -- Дата = 01.01.1999
```

Функция День

Ж Б ф К

Синоним:

Day

Формат описания:

День (Дата : Д) : Ц;

Входные параметры:

Дата — некоторая дата

Назначение:

Возвращает по дате порядковый номер ее дня в месяце.

Пример:

```
Д = День (01.01.99); -- Д = 1
```

Функция ДеньНедели

Б

Синоним:

DayOfWeek

Формат описания:

ДеньНедели (Дата : Д) : Ц;

Входные параметры:

Дата — дата

Назначение:

Возвращает по дате номер дня недели: 1- пн., 2-вт., 3 – ср., 4 – чт., 5 – пт., 6 – сб., 7 – воср.

Пример:

```
Trace(Str(DayOfWeek(ToDay))); -- Сегодняшний день недели
```

Функция ДобавитьМес

Ж Б ф К

Синоним:

AddMon

Формат описания:

ДобавитьМес (День : Д [; Добавить : Ц]) : Д;

Входные параметры:

День — некоторая дата, к которой необходимо прибавить месяц или несколько месяцев

Добавить — число месяцев, которое нужно добавить

Назначение:

Прибавляет к некоторой дате, заданной первым параметром, один месяц, если второй параметр опущен, или же то количество месяцев, что указано во втором параметре. Второй параметр может быть как положительным, так и отрицательным.

Пример:

```
proc Example;
var X : Date;
  X = AddMon(Today, 3); – через три месяца от сего дня
end;
```

Функция КартотекаКалендарь

Б

Синоним:

CardFileCalendar

Формат описания:

КартотекаКалендарь (Картотека : С; ПолеДаты : С; ДеньНедели : С; ТипДня : С; , СписокДнейНедели : С; СписокТиповДня : С; Дата : Д): Д;

Входные параметры:

Картотека — строка с именем картотеки содержащий характеристики дней;

ПолеДаты — строка с именем поля содержащим дату;

ДеньНедели — строка с именем поля содержащим день недели;

ТипДня — строка с именем поля содержащим тип дня;

СписокДнейНедели — строка со списком значений дней недели следующего формата:

“<ИмяДня>=<Значение>[|<ИмяДня>=<Значение>]”,

где:

ИмяДня — строковое название дня недели;

Значение — целое значение дня недели, которое берется и записывается в поле *ДеньНедели*.

СписокТиповДня — строка со списком значений типов дней следующего формата:

“<ИмяТипа>=<Значение>=<Цвет>[|<ИмяТипа>=<Значение>=<Цвет>]”,

где:

ИмяТипа — строковое название типа дня;

Значение — целое значение типа дня, которое берется и записывается в поле ТипДня;

Цвет — целое значение цвета типа дня, которое отображается на календаре.

Дата — дата в календаре.

Назначение:

Функция предназначена для вызова календаря с привязкой к картотеке, содержащей характеристики дней. Для изменения картотеки через стандартный диалог календаря. Возвращает дату, выбранную пользователем или дату, которая была на входе функции в параметре *Дата*.

Пример:

```
vDate = CardFileCalendar(“ХарактеристикиДня”, ’ДАТАДНЯ’, ’ДЕНЬНЕДЕЛИ’, ’ТИПДНЯ’,
    “Понедельник=1¦Вторник=2¦Среда=3¦Четверг=4¦Пятница=5¦Суббота=6¦Воскресенье=7”,
        “Рабочий=0=” + Str(coBlack) + “¦Предпраздничный=1=” +
        Str(coAqua) + “¦Суббота=2=” + Str(coRed) + “¦Воскресенье=4=” +
        Str(coRed) + “¦Праздничный=8=” + Str(coBlue), ToDay);
```

Функция Мес

Ж Б ф К

Синоним:

Mon

Формат описания:

Мес (Дата : Д) : Ц;

Входные параметры:

Дата — некоторая дата

Назначение:

Возвращает по некоторой дате порядковый номер ее месяца в году.

Пример:

```
M = Мес (01.01.99); -- M = 1
```

Функция Сегодня

Ж Б ф К

Синоним:

Today

Формат описания:

Сегодня : Д;

Назначение:

Возвращает системную дату (в процедуре бланка, фильтре картотеки, калькуляторе) или дату бухгалтерской проводки (в типовой операции в журнале).

Примеры:

```
-- В фильтре картотеки:
CreareTime = Сегодня; -- просмотр записей картотеки, созданных сегодня
-- В процедуре бланка:
Проц Р1;
   Перем Дата : Дата;
   Дата = Сегодня; -- Дата = 24.01.2000
Конец;
```

Функции доступа к проводкам

Функция ПроводАналитика

Б

Синоним:

TransSigns

Формат описания:

ПроводАналитика (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает список аналитических признаков проводки. Если возвращается пустая строка, то это говорит о неправильном номере проводки.

Пример:

```
Проц P1(С:Строка);
-- Построение бланка с помощью цикла по проводкам
 Перем Пров:Целое;
  ОчиститьПеременную(Пров);

 Для проводки t = " ", 01.01.2000 01.01.2001 Do
  ВставитьРамку(Пров, Пров+1);
  tDate[Пров] = ПроводДата(t); -- дата проводки
  tDeb[Пров] = ПроводДеб(t); -- дебет проводки
  tCre[Пров] = ПроводКре(t); -- кредит проводки
  tAna[Пров] = "{" + ПроводАналитика(t) + "}"; -- аналитика проводки
  tComm[Пров] = ПроводКоммент(t); -- комментарий к проводке
  tSum1[Пров] = ПроводСумма1(t); -- первая сумма проводки
  tSum2[Пров] = ПроводСумма2(t); -- вторая сумма проводки
  tCur1[Пров] = ПроводВалюта1(t); -- валюта первой суммы проводки
  tCur2[Пров] = ПроводВалюта2(t); -- валюта второй суммы проводки
  tCur[Пров] = ПроводВалюта(t); -- валюта проводки
  tBalSum[Пров] = ПроводСумма(t, "руб"); -- сумма проводки в рублях
  tValSum[Пров] = ПроводСумма(t, "ВАЛ"); -- сумма проводки в валюте
  tKolSum[Пров] = ПроводСумма(t, "кол"); -- количественная сумма проводки
  tKol[Пров] = ПроводЕдиница(t); -- имя единицы количественного учета
  tAccur[Пров] = ПроводТочность(t) -- точность единицы количественного учета
  tKolSign[Пров] = ПроводКоличПризнак(t); -- признак с количественным учетом
  ПроводОткрытьЖурнал(t); -- дата совершения операции
 Конец;
Конец;
```

Функция ПроводВалюта

Б

Синоним:

TransCurrency

Формат описания:

ПроводВалюта (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает идентификатор валюты проводки. Если проводка рублевая, то возвращает имя базовой валюты. Если в проводке две суммы, то возвращает имя иностранной валюты.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводВалюта1

Б

Синоним:

TransCurrency1

Формат описания:

ПроводВалюта1 (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает идентификатор валюты, в которой задана первая сумма проводки. Если валюта первой суммы в журнале опущена, то возвращается идентификатор базовой валюты. Если возвращается пустая строка, то это говорит о неправильном номере проводки.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводВалюта2

Б

Синоним:

TransCurrency2

Формат описания:

ПроводВалюта2 (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает идентификатор валюты, в которой задана вторая сумма проводки. Если валюта второй суммы в журнале опущена, то возвращается идентификатор базовой валюты. Если в проводке единственная сумма, то возвращается пустая строка.

Примечание:

Сравнивая значение, возвращаемое данной функцией, с пустой строкой, можно определить число сумм в проводке.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводДата

Б

Синоним:

TransDate

Формат описания:

ПроводДата (НомерПр : Ц) : Д;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает дату проводки. Если возвращается пустая дата (01.01.100), то это говорит о неправильном номере проводки.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводДеб

Б

Синоним:

TransDeb

Формат описания:

ПроводДеб (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает идентификатор счета дебета. Если возвращается пустая строка, то это говорит о неправильном номере проводки.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводЕдиница

Б

Синоним:

TransMeasure

Формат описания:

ПроводЕдиница (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает имя количественной единицы, если у проводки есть количественный показатель, либо пустую строку — в противном случае.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводЖурнал

Б

Синоним:

TransJurNum

Формат описания:

ПроводЖурнал (НомерПр : Ц) : Ц;

Входные параметры:

НомерПр — номер проводки

Назначение:

Функция по номеру проводки (переменная цикла по проводкам) дает номер журнала, из которого эта проводка получена.

Пример:

```
Проц ФормированиеЖурнала(Об:строка);
VAR Т:INTEGER;
VAR Проводка:string;
VAR Коммент:string;
VAR ПрЖур:string;
VAR ИмяФайла:string;
VAR ДатН:date;
VAR ДатК:date;
  ДатН=01.01.2001;
   ДатК=01.07.2001;
    ИмяФайла="Журнал.Jur";
    For transaction Т="*",ДатН,ДатК do
      ...
      ...
    ПрЖур="["+Str(TransJurNum(Т))+"," 
+SubStr(RecordToString(TransRecord(Т)),2,Length(RecordToString(TransRecord(Т)))-
2)+"]";
     Коммент="- +TransComment(Т)+" "+ПрЖур;
     Проводка=": "+str(TransSum(Т),2)+" "+TransDeb(Т)+» «+TransCre(Т)+»
"+TransSigns(Т)+" "+Коммент;
     WriteLn(ИмяФайла,Проводка);
  Конец;
Конец;
```

Функция ПроводЗапись

Б

Синоним:

TransRecord

Формат описания:

ПроводЗапись (НомерПр : Ц) : З;

Входные параметры:

НомерПр — номер проводки

Назначение:

По указанному номеру проводки возвращает уникальный ключ записи {UKEY}, если проводка размещается в журнале-картотеке или табличном журнале, иначе — константу **Пусто/Nil.**

Пример:

```
Проц Р1;
    Перем R: Запись;
    Перем t: Целое;
    Перем ИмяКарт: Строка;
    Для проводки t = "*", 01.01.99, Сегодня Цикл
        ИмяКарт = ПроводКартотека(t);
        R = ПроводЗапись(t);
        Трассировка("Имя картотеки: "+ИмяКарт+" ;запись:
"+ЗаписьВСтроку(R));
    Конец;
Конец;
```

См. также:

Функцию **ПроводКартотека**

Функция ПроводИмяПризн

Б

Синоним:

TransSignName

Формат описания:

ПроводИмяПризн (НомПроводки : Ц; НомерПризн : Ц) : С;

Входные параметры:

НомПроводки — номер проводки

НомерПризн — номер аналитического признака проводки

Назначение:

Возвращает идентификатор аналитического признака по его номеру в заданной проводке.

Пример:

```
ПРОЦ Р1;
 ПЕРЕМ t, i : Целое;
```

```
ДЛЯ ПРОВОДКИ t = *, 01.01.98, 01.03.98 ЦИКЛ
 ДЛЯ i = 1..ПроводПризнаков(t) ЦИКЛ
  Сообщение ("Имя аналитического признака" + Стр(i) "в проводке" +
                              Стр(t) + " - " + ПроводИмяПризн(t,i));
 Конец;
Конец;
Конец;
```

Функция ПроводКартотека

Б

Синоним:

TransCardFile

Формат описания:

ПроводКартотека (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Данная функция позволяет работать с табличным журналом и журналами-картотеками при обработке проводок в цикле по проводкам и по заданному номеру проводки возвращает:

- имя (идентификатор) картотеки, если проводка размещается в журнале-картотеке;
- строку "Table_OPER", если проводка размещается в табличном журнале;
- иначе — пустую строку.

Пример:

```
Проц P1;
   Перем R: Запись;
   Перем t: Целое;
   Перем ИмяКарт: Строка;
   Для проводки t = "*", 01.01.99, Сегодня Цикл
      ИмяКарт = ПроводКартотека (t);
      R = ПроводЗапись (t);
      Трассировка ("Имя картотеки: " + ИмяКарт + " ;запись: "
                                        + ЗаписьВСтроку (R));
   Конец;
Конец;
```

Функция ПроводКоличПризнак

Б

Синоним:

TransQuantSign

Формат описания:

ПроводКоличПризнак (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает идентификатор аналитического признака, для которого задано количество, либо пустую строку, если у данной проводки нет количественного показателя.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводКоммент

Б

Синоним:

TransComment

Формат описания:

ПроводКоммент (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает комментарий к проводке.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводКре

Б

Синоним:

TransCre

Формат описания:

ПроводКре (НомерПр : Ц) : С;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает идентификатор счета кредита. Если возвращается пустая строка, то это говорит о неправильном номере проводки.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводНомерПризнака

Б

Синоним:

TransNumSign

Формат описания:

ПроводНомерПризнака (НомерПр : Ц; Признак: Строка): Целое;

Входные параметры:

НомерПр — номер проводки

Признак — идентификатор признака

Назначение:

Возвращает номер признака в проводке. 0 — если признака нет в проводке. Если признак задад как справочник (с точкой на конце), то возвращается номер признака из этого справочника.

Пример 1:

```
for transaction t = «“*{ЗАРПЛ!}», 01.01.2003, 01.01.2007 do
    i = TRANSNUMSIGN(t, “-ЗАРПЛ.”);
-- Получить номер признака в проводке принадлежащий справочнику ЗАРПЛ и
привязанный к счету дебет
      if i > 0:
-- Есть признак, выводим о нём информацию
       vSign = TransSignName(t, i);
      Trace(vSign + “ – “ + НАИМ_П(Substr(vSign, 2, Length(vSign))));
     end;
end;
```

Пример 2:

```
for transaction t = «“*{ЗАРПЛ!}», 01.01.2003, 01.01.2007 do
    i = TRANSNUMSIGN(t, “+ЗАРПЛ.НАЧИСЛ.019”);
-- Получить номер признака в проводке привязанный к счету кредета
```

```
        if i > 0:
-- Есть признак, выводим о нём информацию
      vSign = TransSignName(t, i);
     Trace(vSign + " - " + НАИМ_П(Substr(vSign, 2, Length(vSign))));
      end;
end;
```

Пример 3:

```
for transaction t = «"*{ТМЦ!}», 01.01.2003, 01.01.2007 do
  i = TRANSNUMSIGN(t, "ТМЦ");
-- Получить номер признака в проводке не привязанный к счету
  if i > 0: - Есть признак, выводим о нём информацию
    vSign = TransSignName(t, i);
    Trace(vSign + " - " + НАИМ_П(vSign));
  end;
end;
```

См. также:

Функцию **ПроводПризнаков**

Процедура ПроводОткрытьЖурнал

Б

Синоним:

TransOpenJur

Формат описания:

ПроводОткрытьЖурнал (НомерПр : Ц);

Входные параметры:

НомерПр — номер проводки

Назначение:

В немодальном окне открывает журнал (текстовый, табличный или журнал-картотеку), в котором записана данная проводка, и делает текущей соответствующую запись этого журнала.

Пример:

См. пример для функции **ПроводАналитика**.

Функция ПроводПризнаков

Б

Синоним:

TransSignCount

Формат описания:

ПроводПризнаков (НомПроводки : Ц) : Ц;

Входные параметры:

НомПроводки — номер проводки

Назначение:

Возвращает количество аналитических признаков заданной проводки.

Пример:

```
ПРОЦ Р1;
 ПЕРЕМ t : Целое;
 ДЛЯ ПРОВОДКИ t = *, 01.01.98, 01.03.98 ЦИКЛ
 Сообщение ("Проводка" + Стр(t) + "содержит" + ПроводПризнаков(t) + "признаков";
 Конец;
Конец;
```

Функция ПроводПризнКол

Б

Синоним:

TransSignQuant

Формат описания:

ПроводПризнКол (НомПроводки : Ц; НомПризн : Ц) : Л;

Входные параметры:

НомПроводки — номер проводки

НомПризн — номер аналитического признака проводки

Назначение:

Возвращает значение "ИСТИНА", если в проводке с номером *НомПроводки* аналитический признак с номером *НомПризн* содержит количественную сумму, иначе — "ЛОЖЬ".

Пример:

```
ПРОЦ Р1;
 ПЕРЕМ t, i :Целое;
 ДЛЯ ПРОВОДКИ t = *, 01.01.98, 01.03.98 ЦИКЛ
  ДЛЯ i = 1..ПроводПризнаков(t) ЦИКЛ
   Если (ПроводПризнКол(t, i)) :
    Сообщение ("В проводке" + Стр(t) + " аналитический признак" +
              ПроводИмяПризн (t, i) + " содержит количественную сумму");
  ПРЕРВАТЬ;
    Конец;
  Конец;
 Конец;
Конец;
```

Функция ПроводСумма

Б

Синоним:

TransSum

Формат описания:

ПроводСумма (НомерПр : Ц [, ТипПоказ : С]) : Ч;

Входные параметры:

НомерПр — номер проводки

ТипПоказ — тип показателя, по которому запрашивается сумма:

- имя базовой валюты — для получения балансовой суммы
- "ВАЛ" — для получения суммы в валюте
- "КОЛ" — для получения количества
- имя твердого эквивалента — для получения суммы в твердом эквиваленте

Любое другое значение параметра *ТипПоказ* приводит к возникновению ошибки.

Назначение:

Возвращает сумму проводки по заданному показателю. Если проводка рублевая, то валютная сумма равна нулю.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводСумма1

Б

Синоним:

TransSum1

Формат описания:

ПроводСумма1 (НомерПр : Ц) : Ч;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает величину первой суммы проводки.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводСумма2

Б

Синоним:

TransSum2

Формат описания:

ПроводСумма2 (НомерПр : Ц) : Ч;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает величину второй суммы проводки или ноль, если в проводке одна сумма.

Пример:

См. пример для функции **ПроводАналитика.**

Функция ПроводТочность

Б

Синоним:

TransAccuracy

Формат описания:

ПроводТочность (НомерПр : Ц) : Ц;

Входные параметры:

НомерПр — номер проводки

Назначение:

Возвращает точность количественной единицы, в которой была выполнена проводка, или ноль — если в проводке нет количественной суммы.

Пример:

См. пример для функции **ПроводАналитика.**

Процедуры и функции доступа к отчетам

Процедура ОтчВосстановить

Б

Синоним:

RepReset

Формат описания:

ОтчВосстановить (Отчет : О);

Входные параметры:

Отчет — переменная типа Отчет

Назначение:

Устанавливает отчет в начальное состояние, т.е. делает текущей строкой строку итогов.

Пример:

```
ПРОЦ Р1 (С:Строка);
 ПЕРЕМ Отч : Отчет;
 Отч = ОтчОткрыть (Отч1, , , 01.01.97, 01.01.98);
 -- операторы
 ОтчВосстановить (Отч); -- восстановить отчет
 Отч = ОтчЗакрыть (Отч);
Конец;
```

Функция ОтчЗакрыть

Б

Синоним:

RepClose

Формат описания:

ОтчЗакрыть (Отчет : О) : О;

Входные параметры:

Отчет — переменная типа Отчет

Назначение:

Закрывает структуру типа Отчет, созданную функцией **ОтчОткрыть**. Возвращаемое этой функцией значение необходимо присвоить переменной, которая передавалась в нее в качестве параметра. Это обеспечивает обнуление этой переменной.

Пример:

```
ПРОЦ Р1 (С:Строка);
-- Построение бланка на основании отчета
   ПЕРЕМ Отч : Отчет;
   ПЕРЕМ Раб : Логич;
   Отч = ОтчОткрыть (Отч1, , , 01.01.97, 01.01.98);
   Попытка
      ОчиститьПеременную (Секц);
      Раб = ОтчСледующий (Отч); -- переход к первой строке отчета
      ПОКА Раб ЦИКЛ
         ВставитьРамку (Секц, Секц+1);
         ID[Секц] = ОтчНазвание (Отч); -- идентификатор текущей строки отчета
         COMM[Секц] = ОтчКоммент (Отч); -- комментарий текущей строки отчета
         SDS[Секц] = ОтчОстатокН (Отч, отчДеб, РУБ); -- сальдо на начало периода
         SCS[Секц] = ОтчОстатокН (Отч, отчКре, РУБ); -- сальдо на начало периода
         TD[Секц] = ОтчОборот (Отч, отчДеб, РУБ); -- оборот по строке отчета
         TC[Секц] = ОтчОборот (Отч, отчКре, РУБ); -- оборот по строке отчета
         SDF[Секц] = ОтчОстатокК (Отч, отчДеб, РУБ); -- сальдо на конец периода
         SCF[Секц] = ОтчОстатокК (Отч, отчКре, РУБ); -- сальдо на конец периода
         Раб = ОтчСледующий (Отч); -- переход к следующей строке отчета
      Конец;
      SDST = ОтчИтогОстатокН (Отч, отчДеб, РУБ); -- сальдо на начало периода
      -- по отчету в целом
      SCST = ОтчИтогОстатокН (Отч, отчКре, РУБ); -- сальдо на начало периода
      -- по отчету в целом
      TDT = ОтчИтогОборот (Отч, отчДеб, РУБ); -- оборот по отчету в целом
      TCT = ОтчИтогОборот (Отч, отчКре, РУБ); -- оборот по отчету в целом
      SDFT = ОтчИтогОстатокК (Отч, отчДеб, РУБ); -- сальдо на конец периода
      -- по отчету в целом
      SCFT = ОтчИтогОстатокК (Отч, отчКре, РУБ); -- сальдо на конец периода
      -- по отчету в целом
   Окончание
      Отч = ОтчЗакрыть (Отч); -- закрыть отчет
   Конец;
Конец;
```

Функция ОтчИтогОборот

Б

Синоним:

RepTotalTurn

Формат описания:

ОтчИтогОборот (Отчет : О; ДебКре : Ц [; ТипПоказ : С]) : Ч;

Входные параметры:

Отчет — переменная типа Отчет

ДебКре — вид требуемой суммы. Допустимы следующие значения:

- 0 (константа *отчСверн*) — свернутое сальдо
- 1 (константа *отчДеб*) — дебетовое сальдо
- 2 (константа *отчКре*) — кредитовое сальдо

ТипПоказ — тип показателя, по которому запрашивается сумма:

- имя базовой валюты — для получения балансовой суммы
- "ВАЛ" — для получения суммы в валюте
- "КОЛ" — для получения количества
- имя твердого эквивалента — для получения суммы в твердом эквиваленте

Другие значения параметров *ДебКре* и *ТипПоказ* приводят к ошибке.

Назначение:

Возвращает оборот по отчету в целом.

Примечание:

Параметр *ТипПоказ* может быть опущен или быть равным пустой строке. В этом случае вычисляется оборот в базовой валюте.

Пример:

См. пример для функции **ОтчЗакрыть.**

Функция ОтчИтогОстатокК

Б

Синоним:

RepTotalSaldoF

Формат описания:

ОтчИтогОстатокК (Отчет : О; ДебКре : Ц [; ТипПоказ : С]) : Ч;

Входные параметры:

Отчет — переменная типа Отчет

ДебКре — вид требуемой суммы. Допустимы следующие значения:

- 0 (константа *отчСверн*) — свернутое сальдо
- 1 (константа *отчДеб*) — дебетовое сальдо
- 2 (константа *отчКре*) — кредитовое сальдо

ТипПоказ — тип показателя, по которому запрашивается сумма:

- имя базовой валюты — для получения балансовой суммы

- "ВАЛ" — для получения суммы в валюте
- "КОЛ" — для получения количества
- имя твердого эквивалента — для получения суммы в твердом эквиваленте

Другие значения параметров *ДебКре* и *ТипПоказ* приводят к ошибке.

Назначение:

Возвращает сальдо на конец периода по отчету в целом.

Примечание:

Параметр *ТипПоказ* может быть опущен или быть равным пустой строке. В этом случае вычисляется остаток в базовой валюте.

Пример:

См. пример для функции **ОтчЗакрыть.**

Функция ОтчИтогОстатокН

Б

Синоним:

RepTotalSaldoS

Формат описания:

ОтчИтогОстатокН (Отчет : О; ДебКре : Ц [; ТипПоказ : С]) : Ч;

Входные параметры:

Отчет — переменная типа Отчет

ДебКре — вид требуемой суммы. Допустимы следующие значения:

- 0 (константа *отчСверн*) — свернутое сальдо
- 1 (константа *отчДеб*) — дебетовое сальдо
- 2 (константа *отчКре*) — кредитовое сальдо

ТипПоказ — тип показателя, по которому запрашивается сумма:

- имя базовой валюты — для получения балансовой суммы
- "ВАЛ" — для получения суммы в валюте
- "КОЛ" — для получения количества
- имя твердого эквивалента — для получения суммы в твердом эквиваленте

Другие значения параметров *ДебКре* и *ТипПоказ* приводят к ошибке.

Назначение:

Возвращает сальдо на начало периода по отчету в целом.

Примечание:

Параметр *ТипПоказ* может быть опущен или быть равным пустой строке. В этом случае вычисляется остаток в базовой валюте.

Пример:

См. пример для функции **ОтчЗакрыть.**

Функция ОтчКоммент

Б

Синоним:

RepComment

Формат описания:

ОтчКоммент (Отчет : О) : С;

Входные параметры:

Отчет — переменная типа Отчет

Назначение:

Возвращает комментарий текущей строки отчета.

Пример:

См. пример для функции **ОтчЗакрыть.**

Функция ОтчНазвание

Б

Синоним:

RepIdent

Формат описания:

ОтчНазвание (Отчет : О) : С;

Входные параметры:

Отчет — переменная типа Отчет

Назначение:

Возвращает идентификатор текущей строки отчета, т.е. значение крайнего левого столбца.

Пример:

См. пример для функции **ОтчЗакрыть.**

Функция ОтчОборот

Б

Синоним:

RepTurn

Формат описания:

ОтчОборот (Отчет : О; ДебКре : Ц [; ТипПоказ : С]) : Ч;

Входные параметры:

Отчет — переменная типа Отчет

ДебКре — вид требуемой суммы. Допустимы следующие значения:

- 0 (константа *отчСверн*) — свернутое сальдо
- 1 (константа *отчДеб*) — дебетовое сальдо
- 2 (константа *отчКре*) — кредитовое сальдо

ТипПоказ — тип показателя, по которому запрашивается сумма:

- имя базовой валюты — для получения балансовой суммы
- "ВАЛ" — для получения суммы в валюте
- "КОЛ" — для получения количества
- имя твердого эквивалента — для получения суммы в твердом эквиваленте

Другие значения параметров *ДебКре* и *ТипПоказ* приводят к ошибке.

Назначение:

Возвращает оборот по строке отчета.

Примечание:

Параметр *ТипПоказ* может быть опущен или быть равным пустой строке. В этом случае вычисляется оборот в базовой валюте.

Пример:

См. пример для функции **ОтчЗакрыть.**

Функция ОтчОстатокК

Б

Синоним:

RepSaldoF

Формат описания:

ОтчОстатокК (Отчет : О; ДебКре : Ц [; ТипПоказ : С]) : Ч;

Входные параметры:

Отчет — переменная типа Отчет

ДебКре — вид требуемой суммы. Допустимы следующие значения:

- 0 (константа *отчСверн*) — свернутое сальдо
- 1 (константа *отчДеб*) — дебетовое сальдо
- 2 (константа *отчКре*) — кредитовое сальдо

ТипПоказ — тип показателя, по которому запрашивается сумма:

- имя базовой валюты — для получения балансовой суммы
- "ВАЛ" — для получения суммы в валюте
- "КОЛ" — для получения количества
- имя твердого эквивалента — для получения суммы в твердом эквиваленте

Другие значения параметров *ДебКре* и *ТипПоказ* приводят к ошибке.

Назначение:

Возвращает сальдо на конец периода по строке отчета.

Примечание:

Параметр *ТипПоказ* может быть опущен или быть равным пустой строке. В этом случае вычисляется остаток в базовой валюте.

Пример:

См. пример для функции **ОтчЗакрыть.**

Функция ОтчОстатокН

Б

Синоним:

RepSaldoS

Формат описания:

ОтчОстатокН (Отчет : О; ДебКре : Ц [; ТипПоказ : С]) : Ч;

Входные параметры:

Отчет — переменная типа Отчет

ДебКре — вид требуемой суммы. Допустимы следующие значения:

- 0 (константа *отчСверн*) — свернутое сальдо
- 1 (константа *отчДеб*) — дебетовое сальдо
- 2 (константа *отчКре*) — кредитовое сальдо

ТипПоказ — тип показателя, по которому запрашивается сумма:

- имя базовой валюты — для получения балансовой суммы
- "ВАЛ" — для получения суммы в валюте
- "КОЛ" — для получения количества
- имя твердого эквивалента — для получения суммы в твердом эквиваленте

Другие значения параметров *ДебКре* и *ТипПоказ* приводят к ошибке.

Назначение:

Возвращает сальдо на начало периода по строке отчета.

Примечание:

Параметр *ТипПоказ* может быть опущен или быть равным пустой строке. В этом случае вычисляется остаток в базовой валюте.

Пример:

См. пример для функции **ОтчЗакрыть.**

Функция ОтчОткрыть

Б

Синоним:

RepOpen

Формат описания:

ОтчОткрыть (ИмяОтч : С; УслНаСч : С; УслНаПризн : С; ДатаНач : Д; ДатаКон : Д [; ИмяВал : С] [; Справ : С] [; ОбрПорядок : Л]) : О;

Входные параметры:

ИмяОтч — имя отчета

УслНаСч — условие на счета

УслНаПризн — условие на признаки

ДатаНач — начальная дата

ДатаКон — конечная дата

ИмяВал — имя валюты

Справ — имя справочника (при заданном разбиении на строки на аналитику)

ОбрПорядок — вывод строк в отчете в обратном порядке (“ИСТИНА”) и в прямом порядке (“ЛОЖЬ” — значение по умолчанию).

Назначение:

Строит отчет и возвращает ссылку на соответствующую внутреннюю структуру, которая запоминается в переменной типа Отчет. Все установки для построения отчета, за исключением передаваемых в качестве параметров, берутся: при коротком имени *ИмяОтч* (без указания пути) — из списка внутренних отчетов, при полном имени *ИмяОтч* (с указанием пути) — из файла с настройками внутренних отчетов. Если отчета с таким именем нет или его параметры таковы, что на его основе нельзя построить структуру типа Отчет, то генерируется сообщение об ошибке.

Примечания:

1. В первом параметре можно указать имя отчета как с заданием пути, так и без него. Допускается указывать как полный путь к отчету, так и путь относительно папки с текущей бухгалтерией. В качестве разделителя между путем и именем отчета используется “.” (точка) или вертикальная черта “|” (*см. примеры*). Если в качестве разделителя используется точка, то выражение для первого параметра заключается в одинарные кавычки, а путь и имя отчета записываются в двойных кавычках.
2. При задании в первом параметре только названия отчета (без папки) поиск его настроек выполняется последовательно по всем папкам отчетов в том порядке, в котором они перечислены в списке диалога “Внутренние отчеты”.
3. Если вместо параметров *УслНаСч* и/или *УслНаПризн* указаны пустые строки, то используются соответствующие значения из списка внутренних отчетов. Для задания пустых условий отбора следует использовать пробелы.
4. Невозможно открыть структуру типа Отчет для:

- отчетов по проводкам;
- отчетов с разбиением на таблицы;
- отчетов с разбиением на колонки.

5. Если необязательные параметры отсутствуют или равны пустой строке, то валюта и справочник берутся из настроек отчета, иначе — параметры настроек меняются на заданные значения.

Примеры:

```
rep = ОтчетОткрыть ("По оборотам.Lst","Оборотная ведомость","","",
01.01.98,01.01.99,"USD","ПОСТ");
rep = ОтчОткрыть ("C:\tbw65\Wrk\Пример\По оборотам.lst","Мой отчет",
"","",01.01.98,01.01.99); -- справочник и валюта из настроек
rep = ОтчОткрыть ("Мой отчет","","",01.01.98,01.01.99,"USD");
-- валюта из настроек
rep = ОтчОткрыть ("Мой отчет","","",01.01.98,01.01.99,"","Пост");
-- справочник из настроек
```

Функция ОтчСледующий

Б

Синоним:

RepNext

Формат описания:

ОтчСледующий (Отчет : О) : Л;

Входные параметры:

Отчет — название структуры типа Отчет

Назначение:

Делает текущей строкой следующую строку отчета. Если таковая строка есть, возвращает значение "ИСТИНА". Если к моменту выполнения данной процедуры все строки отчета были исчерпаны, то выполняет процедуру **ОтчВосстановить** и возвращает "ЛОЖЬ".

Пример:

См. пример для функции **ОтчЗакрыть.**

Функция ОтчЧислоСтрок

Б

Синоним:

RepRowsCount

Формат описания:

ОтчЧислоСтрок (Отчет : О) : Ц;

Входные параметры:

Отчет — название структуры типа Отчет

Назначение:

Возвращает число строк в отчете.

Примечание:

Значение вычисляется только при первом вызове функции для данного отчета, поэтому многократный вызов данной функции не сказывается на быстродействии.

Пример:

```
Проц Тест (С:Строка);
 Перем Отч : Отчет;
 Перем Кол :Ц;
 Отч = ОтчОткрыть("Отч1", "", "", 01.01.2002, 01.01.2003);
 Кол = ОтчЧислоСтрок (Отч);
Конец
```

Процедуры и функции выдачи запросов на экран

Функция Альтернатива

Б

Синоним:

Alternate

Формат описания:

Альтернатива (Заголовок : С ; Строки[ ] : С [; Ширина : Ц; [Позиция : Ц [РазмерШрифта : Ц]]]) : Ц;

Входные параметры:

Заголовок — заголовок окна

Строки[ ] — массив строк

Ширина — ширина окна в знакоместах

Позиция — номер текущей (выбранной) строки в окне списка

РазмерШрифта — размер шрифта в окне

Назначение:

Выводит окно с заданным заголовком и списком строк определенной ширины и с заданным размером шрифта. В окне осуществляется выбор одной из строк. Возвращает номер выбранной строки или ноль при отказе от выбора (нажатии клавиши *Esc*).

Примечание:

Если параметр *Ширина* отсутствует или его значение меньше нуля, то ширина окна будет настроена автоматически.

Пример:

```
БланкСШаблоном "Тестовый бланк";
   Номер :Целое = [1];
   МассивСтрок [] :Строка;
   Проц ВыборНДС (НДС : Строка);
      МассивСтрок [1] = " НДС= 16,67%Сумма";
      МассивСтрок [2] = " НДС= 20*Сумма/120";
      МассивСтрок [3] = " НДС= 9,09%Сумма";
      МассивСтрок [4] = " НДС= 10*Сумма/110";
      МассивСтрок [5] = " Без НДС";
      Номер = Альтернатива ("НДС платежа", МассивСтрок, -1, Номер);
      -- После открытия окна со списком строк курсор размещается на
      -- строке, которая была выбрана в предыдущее открытие
```

```
        -- Ширина окна устанавливается автоматически
        Конец;
Конец
```

Функция Ввод

Б

Синоним:

Input

Формат описания:

Ввод (ИмяПерем; Приглашение : С) : Ц;

Входные параметры:

ИмяПерем — имя любой доступной переменной любого типа

Приглашение — приглашение на ввод (любая строка)

Назначение:

Выводит модальное окно с заданным заголовком для ввода значения в определенную переменную. Возвращает *кмдВерно*, если была нажата кнопка **ОК** (при этом присваивает значение переменной), или *кмдОтказ* при нажатии кнопки **Отказ**.

Примечание:

Если тип введенного выражения не соответствует типу переменной, то будет сгенерирована ошибка.

Пример:

```
БланкСШаблоном "Некоторый бланк";
ПРОЦ ПоОткрытию (Отправитель : Строка);
   ПЕРЕМ Код, Рез : Целое;
   Рез = Ввод (Код, "Введите пароль") ;
   Если ( Рез <> кмдОк) :
      ЗакрытьПрог;
   Истина :
      Если (Код <> КодБланка) :
         ЗакрытьПрог;
       Конец;
    Конец;
 Конец;
Конец;
```

Функция ВопрДаНетОтказ

Б

Синоним:

EnqYesNoCancel

Формат описания:

ВопрДаНетОтказ (Вопрос : С [; ТипСообщения : Ц]) : Ц;

Входные параметры:

Вопрос — любая строка с сообщением или вопросом

ТипСообщения — тип сообщения может принимать следующие значения:

тсПредупреждение (mtWarning) — предупреждающее сообщение

тсОшибка (mtError) — сообщение об ошибке

тсИнформация (mtInformation) — инормационное сообщение

тсПодтверждение (mtConfirmation) — подтверждающее сообщение, является значением по умолчанию, если параметр *ТипСообщения* не указан

тсПроизвольное (mtCustom) — произвольное сообщение (картинки на диалоге не будет)

Назначение:

Выводит модальное окно с заданным сообщением или вопросом. Возвращает *кмдДа*, если была нажата кнопка **Да**, *кмдНет* — если кнопка **Нет**, *кмдОтказ* — если кнопка **Отказ**.

Пример:

```
...
Если ( ВопрДаНетОтказ ("Вы будете обедать ?") = кмдДа ) :
 Сообщение ("Приятного аппетита");
Конец;
```

Функция ВопрДаОтказ

Б

Синоним:

EnqOkCancel

Формат описания:

ВопрДаОтказ (Вопрос : С [; ТипСообщения : Ц]) : Ц;

Входные параметры:

Вопрос — любая строка с сообщением или вопросом

ТипСообщения — тип сообщения может принимать следующие значения:

тсПредупреждение (mtWarning) — предупреждающее сообщение

тсОшибка (mtError) — сообщение об ошибке

тсИнформация (mtInformation) — инормационное сообщение

тсПодтверждение (mtConfirmation) — подтверждающее сообщение, является значением по умолчанию, если параметр *ТипСообщения* не указан

тсПроизвольное (mtCustom) — произвольное сообщение (картинки на диалоге не будет)

Назначение:

Выводит модальное окно с заданным сообщением или вопросом. Возвращает *кмдВерно*, если была нажата кнопка **ОК** или *кмдОтказ* — если кнопка **Отказ**.

Пример:

```
...
Если ( ВопрДаОтказ ("Вы будете обедать ?") = кмдВерно ) :
 Сообщение ("Приятного аппетита");
Конец;
```

Функция Вопрос

Б

Синоним:

Enquiry

Формат описания:

Вопрос (Заголовок : С; Вопрос : С; ТекстКнопок : С; ТипДиалога : Ц) : Ц;

Входные параметры:

Заголовок — строка с заголовком диалогового окна

Вопрос — строка с сообщением или вопросом

ТекстКнопок — строка с текстом, выводящихся на кнопках разделеным символом ";"

ТипДиалога — тип диалога может принимать следующие константы:

mtWarning — предупреждающее сообщение

mtError — сообщение об ошибке

mtInformation — информационное сообщение

mtConfirmation — подтверждающее сообщение

mtCustom — произвольное сообщение (в диалоге не будет пиктограммы)

Назначение

Выводит модальное окно с заданным сообщением или вопросом, в нижней части которого расположено несколько кнопок. Возвращает номер нажатой кнопки или 0, если окно было закрыто (или нажата клавиша *Esc*).

Если первый параметр *Заголовок* — пустая строка, то окно выводится с заголовком по типу сообщения (*ТипДиалога*).

Пример:

```
proc P1;
 if Вопрос(«» , «Вы будете обедать?», «Несомненно;Может быть;Не знаю»,
mtConfirmation) = 1:
   Сообщение(«Приятного аппетита»);
 end;
end; --P1
```

Функция ДеревоАльтернатив

Б

Синоним:

AlternateTree

Формат описания:

ДеревоАльтернатив (Заголовок : С ; Строки[ ] : С; Ширина : Ц; Индекс : Ц; РазмерШрифта : Ц; Разделитель : С; Заголовок1 : С; Заголовок2 : С [; ШиринаКол1: Ц [; ВыборГруппы : Л]]) : Ц;

Входные параметры:

Заголовок — заголовок окна

Строки[ ] — массив строк

Ширина — ширина окна в знакоместах

Индекс — индекс в массиве текущей (выбранной) строки в окне с иерархией

РазмерШрифта — размер шрифта в окне

Разделитель — строка с символом разделителем групп

Заголовок1 — строка с заголовком первой колонки с иерархией

Заголовок2 — строка с заголовком второй колонки с коментарием

ШиринаКол1 — ширина первой колонки в знакоместах. Если параметр отсутствует или его значение меньше нуля, то ширина первой колонки будет настроена автоматически;

ВыборГруппы — логический параметр, если его значение "ИСТИНА", то пользователь может выбрать группу. Выбор группы осуществляется двойным кликом мыши на 2-ой колонке или клавишей *Enter*. Если значение параметра равно "ЛОЖЬ" или не задано, то выбор группы запрещен.

Назначение:

Выводит окно с заданным заголовком и деревом строк в первой заданой колонке и коментарием во второй заданой колонке, ширина которого задана автоматически или пользователем, с заданым размером шрифта или по умолчанию.

Возвращает 0 при отказе от выбора или номер индекса в массиве выбранной в окне строки (выбор осуществляется двойным щелчком на требуемой строке).

Примечание:

Если параметр *Ширина* отсутствует или его значение меньше нуля, то ширина окна будет настроена автоматически.

Пример:

```
var vKBK[]: String;
func SelectKBK: String;
 var vIndex: Integer;
  ClearVariable(vKBK);
   vKBK[1] = "188 -- Министерсво внутренних дел РФ";
   vKBK[2] = "188.0000";
   vKBK[3] = "188.0000.0000000";
   vKBK[4] = "188.0000.0000000.000";
   vKBK[5] = "188.0302" -- Органы внутренних дел;
   vKBK[6] = "188.0302.0000000";
   vKBK[7] = "188.0302.0000000.000";
   vKBK[8] = "188.0302.0010000
-- Руководство и управление в сфере установленных функций";
   vKBK[9] = "188.0302.0010000.239
-- Военный персонал и сотр. правоохранительных органов, имеющие спец. звания";
   vKBK[10] = "188.0302.0010000.240 -- Гражданский персонал";
   vIndex = AlternateTree("Список КБК", vKBK, -1, 0, 0, ".", "КБК",
"Коментарий");
    if vIndex = 0 :
      Return("");
    else
    Return(vKBK[vIndex]);
end; -- SelectKBK
```

Процедура ОкноВыполненияПоказать

Б

Синоним:

ShowProgress

Формат описания:

ОкноВыполненияПоказать (Макс : Ц, Имя : С);

Входные параметры:

Макс — параметр целого типа, определяющий максимальное количество заполненных ячеек в полосе;

Имя — строковый параметр надписи на окне выполнения.

Назначение:

Выводит на экран окно, отображающее ход выполнения задачи в виде движущейся светлой полосы.

Пример:

```
ОкноВыполненияПоказать (100, "Выполнение задания"); -- окно выполнения
с надписью "Выполнение задания" выводится на экран
```

Процедура ОкноВыполненияПоложение

Б

Синоним:

ProgressPos

Формат описания:

ОкноВыполненияПоложение (Поз : Ц);

Входные параметры:

Поз — параметр целого типа, определяющий сколько ячеек полосы отобразить заполненными;

Назначение:

В окне выполнения задания продвигает движущуюся полосу на заданную позицию.

Примечание:

Параметр *Поз* не может быть больше параметра *Макс*, заданного процедурой **ОкноВыполненияПоказать**.

Пример:

```
ОкноВыполненияПоложение (80); -- продвижение движущейся полосы на позицию 80.
```

Процедура ОкноВыполненияЗакрыть

Б

Синоним:

CloseProgress

Формат описания:

ОкноВыполненияЗакрыть;

Назначение:

Закрывает окно выполнения задания на экране.

Пример:

```
ОкноВыполненияЗакрыть; -- окно выполнения задания закрыто
```

Процедура ОчиститьТрассировку

Б

Синоним:

ClearTrace

Формат описания:

ОчиститьТрассировку [(ЗакрытьОкно : Л)];

Входные параметры:

ЗакрытьОкно — логический параметр, определяющий, закрывать ли окно сообщений или оставить его на экране

Назначение:

Удаляет все предупреждающие сообщения из окна сообщений.

Примечание:

Если параметр *ЗакрытьОкно* опущен, то он считается равным значению "ЛОЖЬ" (предупреждающие сообщения удаляются из окна сообщений, но оно не закрывается).

Пример:

```
ОчиститьТрассировку (True); -- окно сообщений закрывается на экране
```

Процедура Подсказка

Ж Б

Синоним:

Hint

Формат описания:

Подсказка (Подск : С);

Входные параметры:

Подск — любая строка с подсказкой

Назначение:

Выводит в строку сообщений заданную подсказку.

Пример:

```
ПРОЦ Р1;
 Подсказка ('Для помощи выберете пункт меню помощь');
Конец;
```

Процедура Проверка

Ж Б

Синоним:

Assert

Формат описания:

Проверка (ЛогВыр : Л, Сообщение : С);

Входные параметры:

ЛогВыр — любое логическое выражение

Сообщение — сообщение об ошибке (любая строка)

Назначение:

Выдает сообщение об ошибке, если заданное логическое выражение принимает значение "ЛОЖЬ", в противном случае — никаких действий не выполняет.

Пример:

```
ФУНК Дебет (Выр : Ч) : Ч;
 Проверка (Выр > 0, 'Дебетовый остаток < 0');
 ^Выр;
Конец;
```

Процедура Сообщение

Ж Б

Синоним:

Message

Формат описания:

Сообщение (Сообщение : С);

Входные параметры:

Сообщение — сообщение (любая строка)

Назначение:

Выводит модальное окно с заданным сообщением и кнопкой **ОК**.

Пример:

```
ПРОЦ Р1;
 ПЕРЕМ Стор1, Стор2 : Ч;
 ПЕРЕМ кмдРез : Ц;
 -- вводим сторону прямоугольника
 кмдРез = Ввод ( Стор1, "Введите длину первой стороны прямоугольника " );
 -- вводим другую сторону прямоугольника
 кмдРез = Ввод ( Стор2, "Введите длину второй стороны прямоугольника " );
 -- выводим длину диагонали
 Сообщение ("Длина диагонали прямоугольника равна " +
 Стр ( Sqrt ( Стор1*Стор1 + Стор2*Стор2 ) ) );
Конец;
```

Процедура Трассировка

Ж Б

Синоним:

Trace

Формат описания:

Трассировка (Сообщение : С);

Входные параметры:

Сообщение — сообщение (любая строка)

Назначение:

Выводит заданное сообщение в окно сообщений.

Примечания:

1. Данная процедура может использоваться для отладки как более удобный аналог процедуры **Сообщение**.

2. Если в диалоге "Прочие настройки" флаг **Выдавать предупреждающие сообщения** снят, то предупреждения не будут выдаваться.

Пример:

```
ПРОЦ Р1;
 -- операторы
 Трассировка ( “Данное предупреждение может быть проигнорировано” );
 -- операторы
Конец;
```

Процедуры и функции для работы с файлами

Функция ВыборПапки

Б

Синоним:

ChooseFolder

Формат вызова:

ВыборПапки (ИмяПапки : С; Заголовок : С) : Ц;

Входные параметры:

ИмяПапки — переменная строкового типа

Заголовок — заголовок окна стандартного файлового диалога

Назначение:

Открывает стандартный файловой диалог для выбора папки с заданным заголовком окна и возвращает константу **кмдВерно,** если была выбрана папка, иначе — **кмдОтказ.** Если пользователь выбрал папку, то в параметре *ИмяПапки* возвращается ее имя.

Пример:

```
Раб = ВыборПапки (ИмяПапки, "Открыть папку");
```

Функция ВыборФайла

Б

Синоним:

ChooseFile

Формат вызова:

ВыборФайла (ИмяФайла : С; Заголовок : С [; ТипФайлов : С [; DOSТекст : Л]]) : Ц;

Входные параметры:

ИмяФайла — переменная строкового типа

Заголовок — заголовок окна стандартного файлового диалога

ТипФайлов — строковое выражение для задания типа файлов

DOSТекст — переменная логического типа. Если данный аргумент указан при вызове функции, то в диалоге выбора файла появится флаг "DOS Текст". Флаг будет включен, если значение переменной

“ИСТИНА” и выключен, если значение переменной “ЛОЖЬ”. Если файл был выбран и функция возвращает **cmOK**, то в переменную **DOSТекст** будет записано значение “ИСТИНА”, если флаг “DOS Текст” включен и “ЛОЖЬ”, если данный флаг был выключен *(см. Пример 2)*.

Назначение:

Открывает стандартный диалог для выбора файла с заданными заголовком окна и типами файлов и возвращает константу **кмдВерно**, если был выбран файл, иначе — **кмдОтказ**. Если пользователь выбрал файл, то в параметре *ИмяФайла* возвращается его имя.

Примечания:

1. Если до открытия диалога явно указать имя файла, то в диалоге по умолчанию в поле **Имя файла** будет проставлено указанное имя.
2. В файловом диалоге в выпадающем списке **Тип файлов** показываются типы файлов, заданные в одноименном параметре. В простейшем случае задается наименование типа и через знак “|” маска файлов, например:

```
"Журналы (*.jur)|*.jur"
"Текстовые файлы (*.txt)|*.txt"
```

Можно задать сразу несколько типов файлов, например:

```
"Журналы (*.jur)|*.jur|Бланки (*.bln)|*.bln|Текстовые файлы (*.txt)|*.txt"
```

Можно также задать несколько масок для одного типа файлов, например:

```
"Рисунки|*.bmp; *.wmf; *.emf; *.jpg"
```

Пример 1:

```
ПРОЦ Р1;
   Перем ИмяФайла, ИмяПапки :Строка;
   ИмяФайла = “D:\TBW65P\Readme.txt”;
   Если ВыборФайла (ИмяФайла,“Открыть файл...”,
"Текстовые файлы (*.txt) ¦ *.txt”) = кмдВерно :
      ОткрытьРедактор (ИмяФайла);
   Конец;
Конец;
```

Пример 2:

```
func SelectAnyFile: String;
 var vFileName: String;
 var vResult: Integer;
 var vIsDosText: Logical;
  vResult = ChooseFile(vFileName, “Выберите файл...”, “”, vIsDosText);
   if vResult = cmOk:
   -- vIsDosTex = True, если пользователь в диалоге поставил галочку на
     -- флаг “DOS Текст”
     Return(vFileName);
```

```
    else
      Return("");
    end;
  end;
```

Процедура Дописать

Б

Синоним:

Append

Формат вызова:

Дописать (ИмяФайла : С; Выр : С [; Кодир : Л]);

Входные параметры:

ИмяФайла — имя файла, в который происходит запись

Выр — строковое выражение

Кодир — тип кодировки, в которой записывается строковое выражение: "Истина" — для кодировки в Windows, "Ложь" — для DOS

Назначение:

Записывает строку в конец указанного текстового файла в заданной кодировке без символа перевода строки.

Примечание:

Если параметр *Кодир* отсутствует, то он считается равным значению "Истина".

Пример:

```
Проц ТЕСТ (Объект:Строка);
-- проверка наличия файла "Прочти меня.txt"
 Если  (ЕстьФайл("Прочти меня.txt")) :
   ОткрытьРедактор("Прочти меня.txt", ЛОЖЬ);
   -- если файл существует, то он открывается в окне редактирования в
   -- кодировке DOS
 Илсе;
Конец
```

Функция ЕстьПапка

Б

Синоним:

ExistsFolder

Формат вызова:

ЕстьПапка (Имя папки : С) : Л;

Входные параметры:

ИмяПапки — имя папки

Назначение:

Проверяет, существует ли папка с заданным именем. В случае положительного результата возвращает значение “ИСТИНА”.

Пример:

```
Функ Р1(ИмяПапки:Строка);
  Если ЕстьПапка(ИмяПапки):
    Трассировка(ИмяПапки);
  ^Истина;
  Конец;
Конец;
```

Функция ЕстьФайл

Б

Синоним:

ExistFile

Формат вызова:

ЕстьФайл (ИмяФайла : С) : Л;

Входные параметры:

ИмяФайла — имя файла

Назначение:

Проверяет, существует ли файл с заданным именем. В случае положительного результата возвращает значение “ИСТИНА”.

Пример:

```
...
Если ( ЕстьФайл (“Прочти меня.txt”) ) :
 -- если файл “Прочти меня.txt” есть, то открываем его в окне редактора
 ОткрытьРедактор (“Прочти меня.txt”);
Конец;
```

Функция ЗакрытьРедактор

Б

Синоним:

CloseEditor

Формат вызова:

ЗакрытьРедактор (ИмяФайла : С; Сохранить : Л) : Л;

Входные параметры:

ИмяФайла — имя редактируемого файла

Сохранить — логический параметр, если равен “ИСТИНА”, то выдаётся запрос о сохранении модифицированного файла.

Назначение:

Функция закрывает окно редактора с заданным именем файла.

Пример:

```
Проц Р1(Объект:Строка);
Перем Закрыть : Логич;
  Закрыть = ЗакрытьРедактор («ReadMe.txt», Истина);
Конец;
```

Процедура Записать

Б

Синоним:

WriteLn

Формат вызова:

Записать (ИмяФайла : С; Выр : С [; Кодир : Л]);

Входные параметры:

ИмяФайла — имя файла

Выр — строковое выражение

Кодир — тип кодировки, в которой записывается строковое выражение: “ИСТИНА" — для Windows, “ЛОЖЬ" – для DOS

Назначение:

Записывает строковое выражение в конец указанного файла в заданной кодировке. Если параметр *Кодир* опущен, то он считается равным значению “ИСТИНА”, т.е. строковое выражение записывается в кодировке для Windows.

Пример:

```
...
Если ( ЕстьФайл ("Прочти меня.txt") ) :
 -- если файл "Прочти меня.txt" существует, то в него записывается текст
 Записать ( "Прочти меня.txt", "Привет!", Ложь ); -- текст будет записан
 -- в кодировке DOS
Конец;
```

Процедура ОткрытьРедактор

Б

Синоним:

OpenEditor

Формат вызова:

ОткрытьРедактор (ИмяФайла : С [; Кодировка : Л]);

Входные параметры:

ИмяФайла — имя файла для редактирования

Кодировка — тип кодировки

Назначение:

Открывает файл с заданным именем в окне редактора и в заданной кодировке, если задан второй необязательный параметр. Значение параметра *Кодировка* равное "ИСТИНА" соответствует кодировке Windows, "ЛОЖЬ" — кодировке DOS. Если параметр *Кодировка* отсутствует, то он считается равным значению "ИСТИНА"

Пример:

```
...
Если ( ЕстьФайл ("Прочти меня.txt") ) :
  ОткрытьРедактор ("Прочти меня.txt", ЛОЖЬ);
-- если файл "Прочти меня.txt" есть, то открываем его в окне редактора в
--  кодировке DOS
Конец;
```

Процедура ПереименоватьПапку

Б

Синоним:

RenameFolder

Формат вызова:

ПереименоватьПапку (Старое имя : С ; Новое имя : С);

Входные параметры:

СтароеИмя – старое имя папки

НовоеИмя – новое имя папки

Назначение:

Переименовывает папку с заданным именем.

Процедура ПереименоватьФайл

Б

Синоним:

RenameFile

Формат вызова:

ПереименоватьФайл (Старое имя : С; Новое имя : С);

Входные параметры:

СтароеИмя – старое имя файла

НовоеИмя – новое имя файла

Назначение:

Переименовывает файл с заданным именем.

Пример:

```
Проц Р1(Объект:Строка);
 Если ЕстьФайл("Прочти меня.txt):
  ПереименоватьФайл("Прочти меня.txt, "Read.me");
 Конец;
Конец;
```

Процедура ПечатьТекста

Б

Синоним:

PrintText

Формат вызова:

ПечатьТекста (ИмяФайла : С [; ИмяФайлаНастроек : С [; ИмяПринтера : С]]);

Входные параметры:

ИмяФайла — имя текстового файла

ИмяФайлаНастроек — имя файла с настройками печати

ИмяПринтера — имя принтера, на котором распечатывается документ

Назначение:

Распечатывает текстовый файл, не открывая его на экране.

Примечания:

1. Если путь к распечатываемому файлу указан локальный, то он ищется от каталога с планом бухгалтерии.

2. Если имя файла настроек отсутствует, то печать выполняется согласно стандартным настройкам.

3. Имя принтера следует записывать точно так же, как это указано в выпадающем списке “Принтер" диалога “Печать текста”.

4. Если принтер не задан, то печать происходит на том принтере, который используется по умолчанию.

Пример:

```
ПечатьТекста ("ПрочтиМеня.txt","","HP LaserJet 4P on \\Max24\hp");
```

Процедура СоздатьПапку

Б

Синоним:

CreateFolder

Формат вызова:

СоздатьПапку (ИмяПапки : С);

Входные параметры:

ИмяПапки — имя создаваемой папки

Назначение:

Создает папку с заданным именем.

Пример:

```
Проц Р1(Объект:Строка);
  СоздатьПапку("МояБухгалтерия");
Конец;
```

Процедура СоздатьФайл

Б

Синоним:

CreateFile

Формат вызова:

СоздатьФайл (ИмяФайла : С);

Входные параметры:

ИмяФайла — имя создаваемого файла

Назначение:

Создает файл с заданным именем.

Пример:

```
...
Если ( не ЕстьФайл (“Прочти меня.txt.”) ) :
 -- если файла “Прочти меня.txt” нет, то создаем его и
 -- записываем в него приветствие
 СоздатьФайл (“Прочти меня.txt”);
 Записать ( “Прочти меня.txt”, “Привет” );
Конец;
```

Процедура УдалитьПапку

Б

Синоним:

DeleteFolder

Формат вызова:

УдалитьПапку (ИмяПапки : С);

Входные параметры:

ИмяПапки — имя удаляемой папки

Назначение:

Удаляет папку с заданным именем.

Пример:

```
Проц Р1(Объект:Строка);
  УдалитьПапку("МояБухгалтерия");
Конец;
```

Процедура УдалитьФайл

Б

Синоним:

DeleteFile

Формат вызова:

УдалитьФайл (ИмяФайла : С);

Входные параметры:

ИмяФайла — имя удаляемого файла

Назначение:

Удаляет файл с заданным именем.

Пример:

```
...
Если (ЕстьФайл (“Прочти меня.txt.”) ) :
 -- прверка наличия файла “Прочти меня.txt”
 УдалитьФайл (“Прочти меня.txt”);
 Конец;
```

Процедуры и функции для работы с текстовыми файлами по их номерам

Процедура ТекстФайлВКонец

Б

Синоним:

TextFileGotoEnd

Формат описания:

ТекстФайлВКонец (НомерФайла : Ц);

Входные параметры:

НомерФайла — номер текстового файла

Назначение:

Устанавливает файловый указатель в конец файла.

Примечание:

Данную процедуру полезно использовать при дозаписи в конец существующего файла. Если не сделать этой операции, новые строки будут писаться поверх старых.

Пример:

```
Проц Р1;
   Перем НомерФайла : Целое;
   НомерФайла = ТекстФайлОткрытьНаЗапись ("Текст.txt"); -- открывает
                                          -- существующий файл на запись
   ТекстФайлВКонец (НомерФайла);
   ТекстФайлДописатьСтроку (НомерФайла, "Этот текст будет добавлен в
                                                конец файла");
   ТекстФайлЗакрыть (НомерФайла); -- закрытие файла
Конец;
```

Процедура ТекстФайлДописатьСтроку

Б

Синоним:

TextFileAppendLn

Формат описания:

ТекстФайлДописатьСтроку (НомерФайла : Ц; Строка : С);

Входные параметры:

НомерФайла — номер текстового файла

Строка — исходная строка, записываемая в файл

Назначение:

Записывает в файл строку, начиная с текущей позиции.

Пример:

См. пример для процедуры **ТекстФайлВКонец**

Процедура ТекстФайлЗакрыть

Б

Синоним:

TextFileClose

Формат описания:

ТекстФайлЗакрыть (НомерФайла : Ц);

Входные параметры:

НомерФайла — номер текстового файла

Назначение:

Закрывает файл, ранее открытый функциями **ТекстФайлСоздать, ТекстФайлОткрытьНаЧтение, ТекстФайлОткрытьНаЗапись.** (Обязательное действие для корректной работы, иначе — появляется сообщение об ошибке.)

Пример:

См. пример для процедуры **ТекстФайлВКонец**

Процедура ТекстФайлЗаписатьСтроку

Б

Синоним:

TextFileWriteLn

Формат описания:

ТекстФайлЗаписатьСтроку (НомерФайла : Ц; Строка : С [; EOL : Л]);

Входные параметры:

НомерФайла — номер текстового файла

Строка — исходная строка, записываемая в файл

EOL — необязательный логический параметр, по умолчанию равен значению “ЛОЖЬ”. Если задать значение “ИСТИНА”, то при записывании строки, производится анализ на закрытие предыдущей строки (предыдущие две позиции) символами возврата каретки. Если не закрыты, то они дописываются перед строчкой.

Назначение:

Записывает в файл строку, начиная с текущей позиции, и заканчивает ее символами возврата каретки.

Пример:

См. пример для процедуры **ТекстФайлВКонец**

Функция ТекстФайлКонецФайла

Б

Синоним:

TextFileEndOfFile

Формат описания:

ТекстФайлКонецФайла (НомерФайла : Ц) : Л;

Входные параметры:

НомерФайла — номер текстового файла

Назначение:

Возвращает признак достижения конца файла (обычно при чтении).

Пример:

```
Проц Р1;
   Перем ТекСтр : Строка;
   Перем НомерФайла : Целое;
   НомерФайла = ТекстФайлОткрытьНаЧтение (“Readme.txt”);
  -- открывает файл только на чтение
   Попытка
      Пока ТекстФайлКонецФайла (НомерФайла) = Ложь Цикл
         ТекСтр = ТекстФайлПрочитатьСтроку (НомерФайла);
      Конец;
      Окончание
      ТекстФайлЗакрыть (НомерФайла);
   Конец;
Конец;
```

Функция ТекстФайлОткрытьНаЗапись

Б

Синоним:

TextFileOpenWrite

Формат описания:

ТекстФайлОткрытьНаЗапись (ИмяФайла : С; [DOS : Л]) : Ц;

Входные параметры:

ИмяФайла — имя текстового файла

DOS — тип кодировки. Если он равен значению “ИСТИНА”, то файл записан в DOS-кодировке, если опущен или равен значению “ЛОЖЬ” — в Windows-кодировке

Назначение:

Открывает на запись существующий текстовый файл. Если такого файла нет, возбуждается исключение. Возвращает номер файла, который впоследствии может использоваться в других функциях.

Примечание:

Если текстовый файл был открыт данной функцией, то его необходимо закрыть процедурой **ТекстФайлЗакрыть**, иначе — при следующем обращении к файлу выдается сообщение об ошибке: “Нельзя открыть файл”. В этом случае следует закрыть программу и открыть ее снова.

Пример:

См. пример для процедуры **ТекстФайлВКонец**

Функция ТекстФайлОткрытьНаЧтение

Б

Синоним:

TextFileOpenRead

Формат описания:

ТекстФайлОткрытьНаЧтение (ИмяФайла : С; [DOS : Л]) : Ц;

Входные параметры:

ИмяФайла — имя текстового файла

DOS — тип кодировки. Если он равен значению “ИСТИНА”, то файл записан в DOS-кодировке, если опущен или равен значению “ЛОЖЬ” — в Windows-кодировке

Назначение:

Открывает на чтение (с запретом записи) существующий текстовый файл. Если такого файла нет, возбуждается исключение. Возвращает номер файла, который впоследствии может использоваться в других функциях.

Примечание:

Если текстовый файл был открыт данной функцией, то его необходимо закрыть процедурой **ТекстФайлЗакрыть**, иначе — при следующем обращении к файлу выдается сообщение об ошибке: "Нельзя открыть файл". В этом случае следует закрыть программу и открыть ее снова.

Пример:

```
Проц Р1;
 Перем ТекущаяСтрока : Строка;
 Перем НомерФайла : Целое;
 НомерФайла = ТекстФайлОткрытьНаЧтение ("Текст.txt");
 -- открывает файл только на чтение
    ТекущаяСтрока = ТекстФайлПрочитатьСтроку (НомерФайла);
    ТекстФайлЗакрыть (НомерФайла);
Конец;
```

Функция ТекстФайлПрочитать

Б

Синоним:

TextFileRead

Формат описания:

ТекстФайлПрочитать (НомерФайла : Ц; ЧислоСимволов : Ц) : С;

Входные параметры:

НомерФайла — номер текстового файла

ЧислоСимволов — максимальная длина считываемой строки (1..255)

Назначение:

Считывает заданное число символов в текущей строке из текстового файла и возвращает их в виде строки, не переходя на другую строку. Если количество символов в текущей строке меньше заданного, то функция вернет имеющееся в строке число символов. Если количество символов превышает заданное, то текущая строка не обрезается, как в функции **ТекстФайлПрочитатьСтроку**, а возвращается в виде нескольких строк (их количество зависит от длины текущей строки и заданного числаа считываемых символов).

Примечание:

В отличие от функции ТекстФайлПрочитатьСтроку, в данной функции нет ограничений на длину считываемой строки.

Пример:

```
Проц Р1 (Объект : Строка);
 Перем ТекущаяСтрока  : Строка;
 Перем НомерФайла : Целое;
 НомерФайла = ТекстФайлОткрытьНаЧтение ("C:\Readme.txt"); -- открывает
 -- файл только на чтение
 Попытка
    Пока Не ТекстФайлКонецФайла (НомерФайла) Цикл
       ТекущаяСтрока = ТекстФайлПрочитать (НомерФайла, 100); -- читать по
       --100 символов
       Если ТекущаяСтрока = "": -- дошли до конца строки
          Трассировка ("eoln");
          -- проверка, необходимая, для последней строки файла
          Если Не ТекстФайлКонецФайла (НомерФайла) : -- проверяем, не
          -- достигнут ли конец файла
             ТекущаяСтрока = ТекстФайлПрочитатьСтроку (НомерФайла);
             -- переход на новую строку и возврат пустого значения
          Конец;
          Трассировка (ТекущаяСтрока);
       Иначе
          Трассировка (ТекущаяСтрока);
       Конец;
    Конец;
 Окончание
 ТекстФайлЗакрыть (НомерФайла);
 Конец;
Конец;
```

См. также:

Функцию **ТекстФайлПрочитатьСтроку**

Функция ТекстФайлПрочитатьСтроку

Б

Синоним:

TextFileReadLn

Формат описания:

ТекстФайлПрочитатьСтроку (НомерФайла : Ц) : С;

Входные параметры:

НомерФайла — номер текстового файла

Назначение:

Читает строку, начиная с текущей позиции файлового указателя. При попытке чтения за концом файла возбуждает исключение.

Если строка длинее 255 символов, возвращает первые 255 символов строки.

Пример:

См. пример для функции **ТекстФайлПрочитать**

См. также:

Функцию **ТекстФайлПрочитать**

Функция ТекстФайлСоздать

Б

Синоним:

TextFileCreate

Формат вызова:

ТекстФайлСоздать (ИмяФайла : С; [DOS : Л]) : Ц;

Входные параметры:

ИмяФайла — имя текстового файла

DOS — тип кодировки. Если он равен значению “ИСТИНА”, то файл записан в DOS-кодировке, если опущен или равен значению “ЛОЖЬ” — в Windows-кодировке

Назначение:

Создает новый пустой текстовый файл. Если такой файл уже существовал, он очищается. Возвращает номер файла, который впоследствии может использоваться в других функциях.

Примечание:

Если текстовый файл был открыт данной функцией, то его необходимо закрыть процедурой **ТекстФайлЗакрыть**, иначе — при следующем обращении к файлу выдается сообщение об ошибке: “Нельзя открыть файл”. В этом случае следует закрыть программу и открыть ее снова.

Пример:

```
Проц Р1;
   Перем НомерФайла : Целое;
   НомерФайла = ТекстФайлСоздать (“Текст.txt”);
  -- создается файл с именем “Текст.txt”
   ТекстФайлВКонец (НомерФайла);
   ТекстФайлДописатьСтроку (НомерФайла, “ Этот текст будет добавлен”);
   ТекстФайлЗакрыть (НомерФайла);
Конец;
```

Процедуры и функции для работы с бланками

Функция БланкИмяФайла

Б

Синоним:

BlankFileName

Формат описания:

БланкИмяФайла (ИмяБланка : С) : С;

Входные параметры:

ИмяБланка — имя бланка

Назначение:

Возвращает по имени бланка имя bln-файла с его описанием.

Примечание:

Если *ИмяБланка* = “” (пустая строка), то возвращается имя файла с описанием бланка, вызвавшего эту функцию.

Пример:

```
proc TestBtnClick (Sender :String);
   Trace(BlankFileName());
   Trace(BlankFileName(Документы.Банк.Поручн));
end;
```

Функция БланкОткрыт

Б

Синоним:

IsBlankOpen

Формат описания:

БланкОткрыт (ИмяБланка : С) : Л;

Входные параметры:

ИмяБланка — имя бланка

Назначение:

Функция определяет открыт ли указанный бланк. Если бланк открыт, то возвращается значение “ИСТИНА”, иначе — “ЛОЖЬ”.

Пример:

```
var Log: logical;
    Log = IsBlankOpen("УПРАВЛЕНИЕ.ЦЕНТР_УПРАВЛЕНИЯ");
    if Log = true:
      Message("Бланк открыт");]
    end;
```

Функция БланкСуществует

Б

Синоним:

BlankExist

Формат описания:

БланкСуществует (ИмяБланка : С) : Л;

Входные параметры:

ИмяБланка — имя бланка

Назначение:

Функция определяет существует ли указанный бланк в текущем плане бухгалтерии. Если бланк существует, то возвращается значение "ИСТИНА", иначе — "ЛОЖЬ".

Пример:

```
...
Если(БланкСуществует(Test.Test1);
--если бланк существует, то открываем его в модальном окне
ОткрБланк(Test.Test1,Истина);
end;
```

Функция ВзятьГУИД

Б

Синоним:

GetGUID

Формат описания:

ВзятьГУИД : С;

Назначение:

Возвращает глобальный уникальный идентификатор.

Пример:

```
if GUId = " ":
   GUId = GetGuid;
end;
```

Функция ВзятьДатуНачалаЗамороженногоПериода Б

Синоним:

GetFreezDateBegin

Формат описания:

ВзятьДатуНачалаЗамороженногоПериода : Д;

Назначение:

Возвращает дату начала замороженного периода или нулевую дату, если период не заморожен.

Пример:

```
proc test(s:string);
  message("Дата начала замороженного периода:" +
DateToString(ВзятьДатуНачалаЗамороженногоПериода));
end;
```

Функция ВзятьДатуКонцаЗамороженногоПериода Б

Синоним:

GetFreezDateEnd

Формат описания:

ВзятьДатуКонцаЗамороженногоПериода : Д;

Назначение:

Возвращает дату конца замороженного периода или нулевую дату, если период не заморожен.

Пример:

```
proc test(s:string);
  message("Дата конца замороженного периода:" +
DateToString(ВзятьДатуКонцаЗамороженногоПериода));
end;
```

Функция ВзятьЖурналЗамороженногоПериода Б

Синоним:

GetFreezJurnal

Формат описания:

ВзятьДатуКонцаЗамороженногоПериода : С;

Назначение:

Возвращает имя раздела табличного журнала замороженного периода или пустую строку, если период не заморожен.

Пример:

```
proc test(s:string);
  message("Имя журнала" + ВзятьЖурналЗамороженногоПериода);
end;
```

Функция ВзятьИменаБланков

Б

Синоним:

GetBlankNames

Формат описания:

ВзятьИменаБланков (УсловиеОтбора : С; МассивБланков [ ] : С; Сорт : Л; СИменемБланка : Л; СИменемГруппы : Л) : Ц;

Входные параметры:

УсловиеОтбора — строка с условием отбора бланков *(см. функцию* ***Соотв (Match)****)*

МассивБланков — строковый массив, который заполняется именами бланков

Сорт — логический параметр. Если “ИСТИНА”, то бланки в массиве будут опорядочены по имени, иначе в том порядке, как в структуре учета

СИменемБланка — логический параметр. Если “ИСТИНА”, то в строке элемента массива после имени бланка через через комментарий “--” будет добавлено наименование бланка (см. вторую колонку диалога «Список бланков» в программе ТБ)

СИменемГруппы — логический параметр. Если “ИСТИНА”, то в массиве будут элементы с именами групп, к которой принадлежат бланки

Назначение:

Функция предназначена для заполнения строкового массива бланков именами бланков по условию отбора. Возвращает количество элементов в массиве (количество бланков). При выполнении функции массив предварительно очищается.

Пример 1:

```
var vBlanks[]: String;
var vCount: Integer;
Proc Test (Object : String);
   vCount = GetBlankNames("ДОКУМЕНТЫ.*", vBlanks, True, False, False);
    if vCount > 0 then
     vCount = Alternate("Список документов...", vBlanks);  - Диалог со
списком бланков
      if vCount > 0 then
    Trace(Format(«Выбран бланк %s», vBlanks[vCount]));
     end;
    end;
end;
```

Пример 2:

```
Proc Test (Object : String);
  vCount = GetBlankNames("ДОКУМЕНТЫ.*", vBlanks, True, True, True);
  if vCount > 0 then
    vCount = AlternateTree("Список документов...", vBlanks, -1, -1, -1,
".", "Имя бланка", "Наименование");
    -- Диалог с деревом бланков
     if vCount > 0 then
      Trace(Format("Выбран бланк %s", vBlanks[vCount]));
     end;
  end;
end;
```

Функция ВзятьИмяАктивногоБланка

Б

Синоним:

GetActiveBlankName

Формат описания:

ВзятьИмяАктивногоБланка : C;

Назначение:

Возвращает имя активного бланка. В случае, если активое окно не является окном с бланком, то возвращает пустую строку.

Пример:

```
proc test(s:string);
  message("Вы работаете с бланком:" + GetActiveBlankName);
end;
```

Функция ВзятьКакМассивSQL

Б

Синоним:

GetAsArraySQL

Формат описания:

ВзятьКакМассивSQL (РабОбласть: Ц; ИмяБланка: С; ИмяПоля: С) : Ц;

Входные параметры:

РабОбласть — номер рабочей области, возвращаемый функцией ***OpenWordAreaSQL***.

ИмяБланка — строка с именем бланка, в массивы которого будут записываться данные из полей. Если параметр пуст, то запись будет производиться в текущий бланк.

ИмяПоля — строка со списком массивов и полей разделённых вертикальной чертой, имеющая следующий формат: "<ПеременнаяМассива1>=<ИмяПоля1>[¦<ПеременнаяМассиваN>=<ИмяПоляN>]", где: *ПеременнаяМассива* — имя переменной массива в которую будут записываться значения из поля *ИмяПоля*. Если переменная массива указана неверно, то будет возбуждаться соответствующая ошибка. Если поле рабочей области указано неверно, то запись в соответствующий массив производиться не будет.

Назначение:

Функция предназначена для загрузки значений полей из рабочей области SQL в массивы бланка. Функция возвращает количество загруженных записей в массив. В начале работы функции и при возникновении ошибки указанные массивы очищаются.

Примечания:

1. В массив ссылочного типа можно загружать значения из целочисленных полей.
2. В ТБ.Локальная данная функция ничего не делает и всегда возвращает 0.

Пример:

Загрузка из рабочей области записей в массивы текущего бланка:

```
var Колич[]: Numeric;
var ЦенаУч[]: Numeric;
.................................................
 proc LoadToArray;
   var vWA, vCount: Integer;
    var vSQL: String;
      vSQL = "SELECT [МИНЗАПАС] AS BBB, [ОБЪЕМШТ] AS AAA FROM " +
GetRealCardFileName("ТМЦ") + " ORDER BY [UKEY] ASC";
```

```
        vWA = OpenWorkAreaSQL("", vSQL, "ТМЦ");
        try
         vCount = GetAsArraySQL(vWA, "",'Колич=AAA¦ЦенаУч=BBB');
        finally
         CloseWorkArea(vWA);
        end;
    end; - LoadToArray
```

Функция ВзятьВерсиюКартотеки

Б

Синоним:

GetCardFileVersion

Формат описания:

ВзятьВерсиюКартотеки (ИмяКартотеки : С) : Ч;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Позволяет получить по заданному номеру нужную версию картотеки.

Пример:

```
GetCardFileVersion ("СОТРУДНИКИ") : 1.0;
```

Функция ВзятьСправочникАналит

Б

Синоним:

GetAnaGlossary

Формат описания:

ВзятьСправочникАналит (Признак : С) : С;

Входные параметры:

Признак — аналитический признак

Назначение:

Возвращает наименование аналитического справочника по аналитическому признаку.

Пример:

```
ВзятьСправочникАналит("ОС_.ИЗНОС");
-- Вернёт «ОС_»
```

Функция ВключитьКоманды

Б

Синоним:

EnableCommands

Формат описания:

ВключитьКоманды (Разрешено: Л; НазваниеКоманды1; НазваниеКоманды2;....; НазваниеКомандыN);

Входные параметры:

Разрешено — логический параметр разрешающий включение команд при значении “ИСТИНА”, значение “ЛОЖЬ” выключает команды.

НазваниеКоманды1.....НазваниеКомандыN — список названий команд в виде констант

Назначение:

Функция предназначена для включения или выключения тех или иных команд в интерфейсе пользователя ТБ на время выполнения того или иного бланка, из которого она вызвана. Значение “ИСТИНА” — включает команды, “ЛОЖЬ” — выключает.

Пример:

```
proc ПриСчитывании(S: String);
 ВключитьКоманды(Ложь; cmFirstRec; cmPrevRec; cmNextRec; cmLastRec;
cmEditRec; cmDelRec; cmCancelRec; cmPostRec; cmInsertRec; cmDublRec);
 конец;
```

Данный пример демонстрирует выключение следующих команд в навигаторе картотек при открытии бланка:

```
cmFirstRec - на первую запись в навигаторе картотек;
cmPrevRec - на предыдущую запись в навигаторе картотек;
cmNextRec - на следующую запись в навигаторе картотек;
cmLastRec - на последнюю запись в навигаторе картотек;
cmEditRec - изменить текущую запись в навигаторе картотек;
cmDelRec - удалить текущую запись в навигаторе картотек;
cmCancelRec - отменить изменения в текущей записи в навигаторе картотек;
cmPostRec - подтвердить изменения в текущей записи в навигаторе картотек;
cmInsertRec - вставить новую запись в навигаторе картотек;
cmDublRec - дублировать запись в навигаторе картотек.
```

Функция ВремяМодификацииСек

Б

Синоним:

UpdateTimeSec

Формат описания:

ВремяМодификацииСек : Ц;

Назначение:

Возвращает временную составляющую из поля *UpdateTime* картотечной записи (число секунд с начала суток, в которые была выполнена последняя модификация текущей записи).

Примечание:

Данная функция может использоваться в бланках-редакторах и в фильтрах картотек.

Пример:

```
ВремяСМоментаРед = ВремяМодификацииСек;
```

Функция ВремяСозданияСек

Б

Синоним:

CreateTimeSec

Формат описания:

ВремяСозданияСек : Ц;

Назначение:

Возвращает временную составляющую из поля *CreateTime* записи (число секунд с начала тех суток, в которые была создана текущая редактируемая запись).

Примечание:

Данная функция может использоваться в бланках-редакторах и в фильтрах картотек.

Пример:

```
ВремяСек = ВремяСозданияСек;
```

Процедура ВставитьВМассив

Б

Синоним:

InsertInArray

Формат описания:

ВставитьВМассив (ИмяМассива : *; НомерЭлемента : Ц; НовыйЭлемент : *);

где * — тип нового элемента массива должен соответствовать типу данных переменной-массива.

Входные параметры:

ИмяМассива — имя переменной-массива

НомерЭлемента — номер позиции, на которую ставится новый элемент (все последующие элементы сдвигаются на 1 позицию вправо)

НовыйЭлемент — новый элемент, который вставляется в массив, его тип должен соответствовать типу элементов массива

Назначение:

Вставляет заданный элемент в массив на заданную позицию.

Пример:

```
proc Pr1;
 var C : Integer;
 var Schet : Numeric;
 var S1 : Numeric;
InsertInArray(Schet, 5, S1);
end;
```

Процедура ВставитьВМассивСорт

Б

Синоним:

InsertInArraySort

Формат описания:

ВставитьВМассивСорт (ИмяМассива : *; НовыйЭлемент : *);

где * — тип нового элемента массива должен соответствовать типу данных переменной-массива.

Входные параметры:

ИмяМассива — имя переменной-массива

НовыйЭлемент — новый элемент, который вставляется в массив, его тип должен соответствовать типу элементов массива

Назначение:

Вставляет элемент в упорядоченный массив с сохранением упорядоченности. Все элементы справа от места, в которое вставлен новый элемент сдвигаются на одну позицию вправо.

Пример:

```
БланкСШаблоном "PROBA";
```

```
 C : Integer;
 Schet[] : Numeric;
 S1 : Numeric;
proc СортПоСумм(НаправлениеСортировки: Logical);
  SortArray(Schet, НаправлениеСортировки);
  НаправлениеСортировки = Not НаправлениеСортировки;
end;
proc Pr1;
 InsertInArraySort(Schet, S1);
end;
Конец
```

Функция ВставитьВМассивСорт

Б

Синоним:

InsertInArraySort

Формат описания:

ВставитьВМассив (ИмяМассива : *; НовыйЭлемент : *) : Ц;

где * — тип нового элемента массива должен соответствовать типу данных переменной-массива.

Входные параметры:

ИмяМассива — имя переменной-массива

НовыйЭлемент — новый элемент, который вставляется в массив, его тип должен соответствовать типу элементов массива

Назначение:

Возвращает номер позиции, на которую ставится новый элемент в упорядоченный массив.

Пример:

```
БланкСШаблоном "PROBA";
 C : Integer;
 Schet[] : Numeric;
 S1 : Numeric;
 N  :Integer;
proc СортПоСумм(НаправлениеСортировки: Logical);
  SortArray(Schet, НаправлениеСортировки);
  НаправлениеСортировки = Not НаправлениеСортировки;
end;
proc Pr1;
 InsertInArraySort(Schet, S1);
  N=InsertInArraySort(Schet, S1);
end;
```

Процедура Вставка

Б

Синоним:

Insertion

Формат описания:

Вставка (ИмяФайла : С; ДатаОпер : Д; Опер : С [; Послед : Л] [; Откр : Л]);

Входные параметры:

ИмяФайла — имя файла с текстовым журналом хозяйственных операций

ДатаОпер — дата записи проводки/операции в журнал

Опер — текст проводки/операции (до 254 символов)

Послед — место вставки проводки за указанную дату: "ИСТИНА" — для последней проводки, "ЛОЖЬ" — для первой проводки за эту дату

Откр — режим работы с текстовым журналом: "ИСТИНА" — с открытием журнала на экране, "ЛОЖЬ" — без открытия

Назначение:

Вставляет проводку/типовую операцию в заданный текстовый журнал за указанную дату.

Примечания:

1. Если заданный файл не существует, то выдается предупреждающее сообщение и создается новый файл с указанным именем, в который будет вставлена проводка/операция.

2. Если параметр *ИмяФайла* не задан (пустая строка), то ищется открытый файл на рабочем столе программы под бланком и предлагается вставить в него проводку.

3. Если параметры *Послед* и *Откр* отсутствуют, то они считаются равными значению "ИСТИНА".

Пример:

```
ПРОЦ ПрихВКассу;
   ПЕРЕМ ТО: Строка; -- Вызов типовой операции
   ТО=": "+Стр(Сумма)+" РАБ.КАССА.ПРИХОДВКАССУ ПО=П.2 ПРИЗН=ТОВ.СТР_300"
                ЦЕЛЬ=ПОСТ.Ю.БАЗА1 "+"НДС= "+Стр(НДС,2)+" ФИО= Серов";
   Вставка ("Приход в кассу.jur", 30.09.1998, ТО); -- Вставляет типовую
   -- операцию в журнал "Приход в кассу.jur" за 30.09.98, делает ее
   -- последней за данную дату и открывает журнал на экране
   Вставка ("1998 год - валюта.jur", 03.12.98, ": 440  01 02 {ОС.СТ_1 П.1}"
   +" -- "+"1 вставка",ИСТИНА,ЛОЖЬ); -- Вставляет проводку в журнал
   -- "1998 год - валюта.jur" за 03.12.98, делает ее последней за данную дату
      -- и не открывает журнал на экране
   Вставка ("1998 год - валюта.jur", 03.12.98, ": 120  01 02 {ОС.СТ_2 П.1}"
```

```
    +" -- "+"2 вставка",ЛОЖЬ); -- Вставляет проводку в журнал
    -- "1998 год - Валюта.jur" за 03.12.98, делает ее первой за данную дату
    -- и открывает журнал на экране
КОНЕЦ;
```

Процедура ВставкаВТаблЖурнал

Б

Синоним:

InsertToTableJur

Формат описания:

ВставкаВТаблЖурнал (ИмяРаздела : С; ДатаОпер : Д; Опер : С; Видим : Л);

Входные параметры:

ИмяРаздела — имя раздела табличного журнала

ДатаОпер — дата записи хозяйственной операции в журнал

Опер — текст проводки/операции (до 254 символов, начиная с двоеточия)

Видим — режим работы с табличным журналом: "ИСТИНА" — с открытием журнала на экране, "ЛОЖЬ" — без открытия

Назначение:

Вставляет проводку (типовую операцию) в заданный раздел табличного журнал за указанную дату.

Примечание:

Если заданный раздел не описан в плане бухгалтерии, то при *Видим* = "ИСТИНА" вставки не будет, а при *Видим* = "ЛОЖЬ" будет создан раздел и произведена вставка.

Примеры:

```
...
ВставкаВТаблЖурнал ("1998 год",01.01.98,": 250 50 51"+" -- "+
      "мой комментарий",ИСТИНА);
      -- Вставляет проводку в раздел табличного журнала с именем "1998
год" и    -- открывает этот раздел на экране. Если такой раздел не описан в
ПБ, то    -- вставки не произойдет
ВставкаВТаблЖурнал ("1998 год",01.01.98,": 440 50 51"+" -- "+
      "мой комментарий",ЛОЖЬ);
-- Вставляет проводку в раздел табличного журнала с именем "1998 год",
   -- при этом раздел на экране не открывается. Если такой раздел не описан
   -- в ПБ, то он создается и вставка производится. Увидеть этот раздел и
   -- вставленные записи можно, описав раздел в ПБ
...
```

Процедура ВставитьРамку

Б

Синоним:

InsertFrame

Формат описания:

ВставитьРамку (УпрПер : Упр | Ц; НомерИт : Ц);

Входные параметры:

УпрПер — управляющая переменная

НомерИт — номер новой рамки

Назначение:

Вставляет рамку, состоящую из набора клеток, в повторяющуюся секцию с управляющей переменной *УпрПер*.

Примечания:

1. Итерации нумеруются с единицы.

2. Только в повторяющихся секциях каждая клетка соответствует одному полю, связанному с переменной-массивом. После выполнения указанной процедуры в каждый массив (количество которых равно количеству столбцов секции) будет добавлен элемент с нулевым значением, номер которого равен номеру рамки (итерации) повторяющейся секции.

Пример:

```
...
Для Проводки Пр = "*", 01.04.98, 01.05.98 Цикл
 ВставитьРамку (Секции, Секции+1);
 ПрДата [Секции] = ПроводДата (Пр);
 ПрСумма [Секции] = ПроводСумма (Пр);
Конец;
```

Функция ВставитьРамку

Б

Синоним:

InsertFrame

Формат описания:

ВставитьРамку (УпрПер : Упр | Ц; НомерИт : Ц) : Ц;

Входные параметры:

УпрПер — управляющая переменная

НомерИт — номер рамки, за которой вставляется новая рамка

Назначение:

Возвращает номер новой рамки, состоящей из набора клеток, в повторяющейся секции с управляющей переменной *УпрПер*.

Примечания:

1. Итерации нумеруются с единицы.

2. Позволяет вставлять рамку в отсортированную подтаблицу

3. *НомерИт* и возвращаемое значение функции соответствует сортировке

Пример:

```
...
Для Проводки Пр = "*", 01.04.98, 01.05.98 Цикл
 i = ВставитьРамку (Секции, Секции+1);
 ПрДата [i] = ПроводДата (Пр);
 ПрСумма [i] = ПроводСумма (Пр);
Конец;
```

Процедура ВыполнитьПроцедуру

Б

Синоним:

ExecuteProc

Формат описания:

ВыполнитьПроцедуру (ИмяПроц : С [; Арг1 : *, ..., АргN : *;]) : *;

Входные параметры:

ИмяПроц — строка с именем процедуры. Может содержать полное квалифицированное имя. В этом случае процедура будет вызвана из соответствующего бланка

Арг1, ..., АргN — произвольное число аргументов вызываемой процедуры

* — тип аргументов. Соответствует типу аргумента вызываемой процедуры

Назначение:

Процедура бланка для вызова процедуры по её имени.

Пример:

```
ExecuteProc("Button5_OnClick", "");
```

Процедура ВыполнитьФункцию

Б

Синоним:

ExecuteFunc

Формат описания:

ВыполнитьФункцию* (ИмяФунк : С [; Арг1 : **, ..., АргN : **;]) : ***;

Входные параметры:

* — суффикс функции (D, R, L, S, I, N), соответсвующий типу вызываемой функции.

ИмяФунк — строка с именем функции. Может содержать полное квалифицированное имя. В этом случае функция будет вызвана из соответствующего бланка

Арг1, ..., АргN — произвольное число аргументов вызываемой функции

** — тип аргументов. Соответствует типу аргумента вызываемой функции

*** — тип функции (Date, Record, Logical, String, Integer, Numeric), соответсвующий типу вызываемой функции.

Назначение:

Процедура бланка для вызова функции по её имени.

Пример:

```
var vRes: String;
  vRes := ExecuteFuncS("vStr_OnType", "aaa", 1, 2, 3, "Oops...");
```

Функция ДлинаМассива

Б

Синоним:

LengthOfArray

Формат описания:

ДлинаМассива (ИмяМассива : *) : Ц;

где * — тип переменной-массива

Входные параметры:

ИмяМассива — имя переменной-массива, описанной в бланке

Назначение:

Возвращает длину динамического или статического массива по заданному имени.

Пример:

```
proc ПриОткрытии(Sender: Logical);
 Var i:Integer;
   For i=1..LengthOfArray(Значения) do
   ЗначенияСтр[i]=Str(Значения[i]);
     End;
end;
```

Функция ЗагрузитьИнтернетФайл

Б

Синоним:

DownloadInternetFile

Формат описания:

ЗагрузитьИнтернетФайл (Адрес : С; ИмяФайла : С [; ПроксиЛогин : С; ПроксиПароль : С;]) : Л;

Входные параметры:

Адрес — строка с адресом в интернете

ИмяФайла — имя файла, в который будет сохранена загруженная страница

ПроксиЛогин — имя пользователя для авторизации на прокси-сервере

ПроксиПароль — пароль пользователя для авторизации на прокси-сервере

Назначение:

Возвращает значение “ИСТИНА”, если страница из интернета загружена успешно.

Пример:

```
if not DownloadInternetFile("http://www.dic.ru", "d:\temp\index.html"):
  Message("Сайт ДИЦ не доступен...");
end;
```

Процедура ЗагрузитьРаздел

Б

Синоним:

LoadJurTab

Формат описания:

ЗагрузитьРаздел(Раздел : С; Файл : С);

Входные параметры:

Раздел — имя раздела табличного журнала;

Файл — полное имя текстового файла. Если задано только имя файла, то файл находится в папке с текущей бухгалтерией.

Примечание:

Закоментированные проводки в заданный раздел табличного журнала не считываются.

Назначение:

Загрузка данных из текстового файла в заданный раздел табличного журнала.

Пример:

```
Проц Р1(Объект:Строка);
    ЗагрузитьРаздел("Остатки", "С:\tbw69p\Wrk\Пример\ДанныеТЖ");
    --задан полный путь к файлу
    ЗагрузитьРаздел("Остатки", "ТЖ"); -- задано только имя файла.
Конец;
```

Процедура Закрыть

Б

Синоним:

Close

Формат описания:

Закрыть [(ИмяБланка : С)];

Назначение:

Закрывает бланк, вызвавший данную процедуру.

Входные параметры:

ИмяБланка – имя бланка;

Примечания:

1. Вызов данной процедуры запрещается использовать в теле процедуры-обработчика событий шаблона *ПриОткрытии* и *ПриСчитывании (см. Руководство программиста Гл. XIII)*.

2. Ни в коем случае нельзя закрывать бланк, из которого вызван модальный бланк.

3. Нельзя закрывать закрытый бланк.

Пример:

```
ПРОЦ ЗакрытьБланк(ЗакрБл:Строка);
-- процедура-обработчик, выполняется по нажатию кнопки "ЗакрБл"
  Закрыть;
  -- бланк закрывается
Конец;
```

Функция КартотечныйФильтрBSQL

Б

Синоним:

CardFilterToSQL

Формат описания:

КартотечныйФильтрBSQL (ИмяКартотеки : С; ИмяБазыSQL : С; КартФильтр : С | Длинная Строка; SQLФильтр : С | Длинная Строка) : Л;

Входные параметры:

ИмяКартотеки — имя картотеки, для которой формируется SQL-запрос

ИмяБазыSQL — алиас, который будет добавляться ко всем полям в SQLFilter. Для запросов по нескольким таблицам.

КартФильтр — картотечный фильтр в синтаксисе ТБ. Может быть длинной строкой.

SQLФильтр — переменная в которой будет размещен SQL-фильтр. Может быть длинной строкой.

Назначение:

Функция предназначена для преобразования картотечного фильтра в SQL-фильтр. Возвращает значение “ИСТИНА”, если картотечный фильтр удалось полностью перевести в SQL-фильтр. Если — “ЛОЖЬ”, то в *SQLФильтр* будет только часть SQL-фильтра, которую удалось перевести. В ТБ.Локальная функция всегда возвращает “ЛОЖЬ”.

Примеры:

```
 vRes = CARDFILTERTOSQL("ТМЦ", "", "Признак <> «»', vSQLFilter);
-- [ПРИЗНАК] <> ""
 vRes = CARDFILTERTOSQL("ТМЦ", "A.", "Признак <> «»', vSQLFilter);
-- A.[ПРИЗНАК] <> ""
```

Процедура КопироватьМассив

Б

Синоним:

CopyArray

Формат описания:

КопироватьМассив (Массив1 : * ; Массив2 : * [;ИндексИсточ : Ц; ИндексПрием: Ц]);

где * – тип данных массива

Входные параметры:

Массив1 – идентификатор массива, из которого копировать

Массив2 – идентификатор массива, в который копировать

ИндексИсточ – индекс в массиве источника, начиная с которого копировать в массив приемника

ИндексПрием – индекс в массиве приемника, с которого начнется копирование из массива источника

Назначение:

Копирует содержимое одного массива в другой.

Если указаны параметры *ИндексИсточ* и *ИндексПрием*, то массив приемника НЕ очищается перед копированием и производтся частичное копирование массива источника от элемента с указанным индексом до конца.

Если эти параметры не указаны, то массив приемника предварительно ичищается перед копированием и копирование производтся полностью всего массива источника.

Пример 1:

```
-- Полное копирование одного массива в другой
var vArray1[]: Integer;
var vArray2[]: Integer;
proc CopyArrayExample;
   CopyArray(vArray1, vArray2);
end;
```

Пример 2:

```
-- Полное копирование одного массива в другой но без очистки массива
   приемника
CopyArray(vArray1, vArray2, 1, 1);
```

Пример 3:

```
-- Произвольное копирование одного массива в другой
CopyArray(vArray1, vArray2, 5, 10);
```

Процедура КопироватьФайл

Б

Синоним:

CopyFile

Формат описания:

КопироватьФайл (Источник : С ; Приемник : С);

Входные параметры:

Источник — путь к файлу (или папке) источника, из которого (которой) происходит копирование

Приемник — путь к файлу (или папке) приемника, в который происходит копирование

Назначение:

Копирует файл или папку с заданным именем.

Пример:

```
Проц Р1(Объект:Строка);
 КопироватьФайл(D:\tb69p\67to69.rtf, D:\Temp);
 -- Копирование файла "67to69.rtf" в папку "\Temp"
 CopyFile(D:\tb69d\*.*, D:\Temp\1);
 -- Копирование содержимого папки "\tb69d" в папку "\1" временной папки
 CopyFile(D:\tb69d, D:\Temp);
 -- Копирование папки "\tb69d" в папку "\Temp"
Конец;
```

Функция МаксВМассиве

Б

Синоним:

MaxInArray

Формат описания:

МаксВМассиве (ИмяМассива : *) : Ц;

где * — тип переменной-массива

Входные параметры:

ИмяМассива — имя переменной-массива, описанной в бланке

Назначение:

Возвращает индекс максимального элемента массива.

Примечание

Если максимальных элементов более одного, то функция возвращает индекс последнего из максимальных элементов массива.

Пример:

```
proc Pr;
 var H : Integer;
 var Schet : Numeric;
H = MaxInArray(Schet);
end;
```

Функция МинВМассиве

Б

Синоним:

MinInArray

Формат описания:

МинВМассиве (ИмяМассива : *) : Ц;

где * — тип переменной-массива

Входные параметры:

ИмяМассива — имя переменной-массива, описанной в бланке

Назначение:

Возвращает индекс минимального элемента массива.

Примечание

Если минимальных элементов более одного, то функция возвращает индекс последнего из минимальных элементов массива.

Пример:

```
proc Pr1;
 var C : Integer;
 var Schet : Numeric;
H = MinInArray(Schet);
end;
```

Функция МодальныйРежим

Б

Синоним:

ModalMode

Формат описания:

МодальныйРежим : Л;

Назначение:

Возвращает значение “ИСТИНА” в случае, если вызвавший ее бланк работает в модальном режиме.

Пример:

```
...
Если (МодальныйРежим) :
 Трассировка (Бланк работает в модальном режиме);
Конец;
```

Функция Модуль

Б

Синоним:

ABS

Формат описания:

Модуль (Значение : Ц | Ч) : Ц | Ч;

Входные параметры:

Значение — значение целого или числового типа.

Назначение:

Функция предназначена для взятия значения переменной по модулю и возвращает абсолютное значение переменной целого или числового типа. Тип возвращаемого значения функции должен совпадать с типом входной переменной.

Примеры:

```
Функ P1(ЦенаТов:Число) : Число;
  if  ЦенаТов<0:
  ЦенаТов=Модуль(ЦенаТов);
  fi;
  ^ЦенаТов;
Конец;

Proc P1(Ob:String);
var  : Integer;
```

```
Val=ABS(-100. 51);  -- Val=100. 51
Конец;
```

Функция НайтиОбъект

Б

Синоним:

FindObject

Формат описания:

НайтиОбъект (ИмяОбъекта : С ; ПеременнаяОбъекта) : Л;

Входные параметры:

ИмяОбъекта — имя объекта на шаблоне

ПеременнаяОбъекта — имя локальной переменной типа объекта, в которую будет сохранена ссылка на найденный объект

Назначение:

Функция предназначена для поиска объекта на шаблоне по его имени. Функция возвращает “ИСТИНА”, если объект на шаблоне найден и описан в бланке. Если объект не найден, то функция возвращает “ЛОЖЬ”.

Если в параметре *ИмяОбъекта* указать полное квалифицированное имя объекта другого бланка (не текущего), то функция возвратит ссылку на этот объект, при условии, что данный бланк открыт *(см. пример 2)*.

Пример 1:

```
БланкСШаблоном «Example for FindObject»;
...
Button4: Object;
Button5: Object;
...
Proc Button5_OnClick(Sender: String);
var vObject: Object;

 if not FindObject(“Button4”, vObject):
    Exit;
 end;
 if AsLogical(vObject.Enabled):
  vObject.Caption = “Disabled”;
  vObject.Enabled = False;
 else
  vObject.Caption = “Enabled”;
  vObject.Enabled = True;
 end;
end;
```

Пример 2:

```
FindObject("ИНФО.ТМЦ.Button1", myObject);
```

Процедура ОбработатьАналит

Б

Синоним:

RefreshAna

Формат описания:

RefreshAna;

Назначение:

Предназначена для обновления (перекомпиляции аналитики). В случае, когда отключена автообработка.

Пример:

```
Proc P1(S: String);
 var vCount: Numeric;
  if (Автообработка = False):
   RefreshAna;
   end;
 vCount = Остаток(Тсс.ТМЦ{ТМЦ.00101&П.10&(~Движ.5)}, Дат + 1, "Кол");
end;
```

Процедура ОткрБланк

Б

Синоним:

OpenBlank

Формат описания:

ОткрБланк (ИмяБланка : С [; ПризнакМодал : Л [; КлючЗаписи: З]]);

Входные параметры:

ИмяБланка — имя бланка

ПризнакМодал — признак модальности — необязательный параметр, определяющий тип открывающегося окна с бланком

КлючЗаписи — ключ записи (UKEY) — необязательный параметр, определяющий запись по ее уникальному номеру

Назначение:

Открывает бланк с заданным именем в модальном окне, если признак модальности не указан или его значение равно "ИСТИНА",

в противном случае — бланк открывается немодально. Если указан третий параметр — ключ записи (UKEY), то откроется бланк-редактор картотеки на этой записи (если она есть). Если до вызова данной функции, бланк-редактор был открыт, то он будет закрыт и открыт по-новой на указанную запись (если она есть).

Примечание:

Если последний необязательный параметр *КлючЗаписи* имеет значение NIL, то при открытии бланка-редактора картотеки создается новая запись.

Примеры:

```
Проц Р1;
 ОткрБланк("Инфо.Сотрудник"); --открывает бланк "Физлицо" модально.
 OpenBlank("Инфо.Контрагент", Ложь); -- открывает бланк "Юрлицо"
немодально.
 OpenBlank("Инфо.Подразделение", Истина, ID); -- открывает бланк
"Подразделение" в модальном окне на записи с Ukey = ID.
Конец;
```

Функция ОткрытьТабличныйЖурнал

Б

Синоним:

OpenTabJurnal

Формат описания:

OpenTabJurnal (Раздел : С; КлючЗаписи : З) : Л;

Входные параметры:

Раздел — строка с именем раздела табличного журнала. Раздел табличного журнала должен быть прописан в ПБ. В сетевой версии достаточно, чтобы раздел был в схеме доступа на ТБ.Сервере

Если параметр *Раздел* пустая строка и в ПБ или в схеме доступа (в сетевой версии) описаны разделы, то откроется первый попавшийся.

Если нет разделов в ПБ или в схеме доступа (в сетевой версии), или указан раздел, которого нет в ПБ или в схеме доступа (в сетевой версии), то будет выдано соответсвующее сообщение без возбуждения ошибки и табличный журнал не откроется, а функция вернет “ЛОЖЬ”.

КлючЗаписи — ключ записи (UKEY), на которую необходимо установить курсор. Если NIL или записи с таким ключем нет в разделе, то курсор будет установлен на ту запись, где стоял в последний раз перед закрытием раздела

Назначение:

Открывает раздел табличного журнала с заданным именем на заданной записи. Функция возвращает “ИСТИНА”, если удалось открыть раздел табличного журнала.

Пример:

```
var vOpened: Logical;
 vOpened := OpenTabJurnal(“Остатки”, nil);
   if not vOpened then
   -- Раздел не открылся. Надо что-то делать...
   end;
```

Процедура ОчиститьБланк

Б

Синоним:

ClearBlank

Формат описания:

ОчиститьБланк (ИмяБланка : С; ОчищатьПоля : Л);

Входные параметры:

ИмяБланка — имя бланка

ОчищатьПоля — режим очистки полей по умолчанию (“ИСТИНА” — очищать поля по умолчанию)

Назначение:

Очищает поля бланка с заданным именем.

Пример:

```
ОчиститьБланк (“ОИ”,ИСТИНА);
```

Процедура ОчиститьПеременную

Б

Синоним:

ClearVariable

Формат описания:

ОчиститьПеременную (ИмяПер : С);

Входные параметры:

ИмяПер — имя переменной

Назначение:

Очищает значение переменной с заданным именем.

Пример:

```
ОчиститьПеременную (Отправитель);
```

Функция ПеременныеМодифицированы

Б

Синоним:

VarsIsModified

Формат описания:

ПеременныеМодифицированы (ИмяБланка : С) : Л;

Входные параметры:

ИмяБланка — имя бланка

Назначение:

Возвращает значение “ИСТИНА”, если переменные бланка с заданным именем были изменены.

Пример:

```
...
Если ( ПеременныеМодифицированы (“ОИ”) ) :
 ОчиститьБланк ( “ОИ” );
Конец;
```

Функция ПеременнаяСуществует

Б

Синоним:

VariableExist

Формат описания:

ПеременнаяСуществует (ИмяПеременной : С) : Л;

Входные параметры:

ИмяПеременной — имя переменной бланка

Назначение:

Функция определяет существует ли указанная переменная в бланке. Функция возвращает “ИСТИНА”, если переменная в бланке есть. В случае, если бланк, в котором происходит поиск переменной

открыт и является редактором картотеки, то поиск производится и среди полей картотеки.

Пример:

```
Проц Р1(Объект:Строка);
Перем Сущест : Логич;
  Сущест = ПеременнаяСуществует ("Сумма");
Конец;
```

Процедура Печать

Б

Синоним:

Print

Формат описания:

Печать (ИмяБланка : С [; ИмяФайлаНастроек : С [; ИмяПринтера : С [; Книжн : Л [; НачСтр : Ц [; КонСтр : Ц [; Просмотр : Л]]]]]]);

Входные параметры:

ИмяБланка — имя бланка

ИмяФайлаНастроек — имя файла с настройками печати

ИмяПринтера — имя принтера, на котором распечатывается документ

Книжн — логическая переменная определяет ориентацию страницы при печати (если "ИСТИНА" — книжная, "ЛОЖЬ" — альбомная)

НачСтр — страница, с которой начинается печать бланка

КонСтр — страница, на которой заканчинается печать бланка

Просмотр — логический параметр, если он не указан, то его значение равно "ЛОЖЬ". Если его значение равно "ИСТИНА", то документ находится в режиме предварительного просмотра.

Назначение:

Печатает бланк с шаблоном с заданным именем.

Примечания:

1. Имя принтера следует записывать точно так же, как это указано в выпадающем списке "Принтер" диалога "Печать текста".
2. Если принтер не задан, то печать происходит на том принтере, который используется по умолчанию.
3. Если имя файла настроек отсутствует, то печать выполняется согласно стандартным настройкам.
4. Если заданы три последних необязательных параметра *Книжн*, *НачСтр* и *КонСтр*, то их значения перевешивают значения,

которые берутся из файла, указанного в параметре *ИмяФайлаНастроек*.

Пример:

```
proc кнПечать(S:String);
  Печать("Proba","Настройка печати п/п.cfg", "",true, 1, 2);
end;
```

Процедура ПечатьБланкаРедактора

Б

Синоним:

PrintCardBlank

Формат описания:

ПечатьБланкаРедактора (ИмяКартотеки : С; НомЗап : Запись [; ИмяФайлаНастроек : С [; ИмяПринтера : С]]);

Входные параметры:

ИмяКартотеки — имя картотеки

НомЗап — номер записи

ИмяФайлаНастроек — имя файла с настройками печати

ИмяПринтера — имя принтера, на котором должен распечататься документ

Назначение:

Печатает бланк-редактор картотеки по номеру записи.

Примечания:

1. Печать выполняется только для картотек с одним бланком-редактором, если таких бланков у картотеки более одного, то на экран будет выводиться диалоговое окно выбора бланка-редактора для печати.

2. Имя принтера следует записывать точно так же, как это указано в выпадающем списке “Принтер” диалога “Печать текста”.

3. Если принтер не задан, то печать происходит на том принтере, который используется по умолчанию.

4. Если имя файла настроек отсутствует, то печать выполняется согласно стандартным настройкам.

Пример:

```
Проц ПечатьБланкаРед(Печать : Строка);
 ПечатьБланкаРедактора("УчетКарточкаОС",current, "Настройка печатиОС.cfg");
Конец;
```

Функция ПоискВМассиве

Б

Синоним:

SearchInArray

Формат описания:

ПоискВМассиве (ИмяМассива : *; ЭлементМассива : * [; Сорт : Л] [; Нач : Ц]]) : Ц;

где * - тип элемента массива

Входные параметры:

ИмяМассива — имя переменной-массива, описанной в бланке или столбец подтаблицы

ЭлементМассива — искомый элемент массива

Сорт — логический параметр определяет отсортирован ли массив, по умолчанию равен “ЛОЖЬ”. Если массив, в котором ведется поиск всегда отсортирован, лучше указать значение “ИСТИНА, поскольку это может многократно ускорить поиск.

Нач — номер элемента, с которого начинать поиск. По умолчанию данный параметр равен 1.

Назначение:

Возвращает номер найденного элемента или значение 1, если элемент не найден.

Примечание:

Если тип элемента задан неверно, то для динамического массива он не проверится и ошибка будет сгенерирована уже в процессе, в отличие от статического, в котором данная ошибка генерируется на стадии компилляции.

Пример:

```
Проц Поиск ;
  Перем Обл[] : Строка;
  Перем А : Целое;
  Обл[1] = "3";
  Обл[2] = "2";
  Обл[3] = "11";
  А = ПоискВМассиве ( Обл, "11" );
    -- Y = 3
end;
```

Функция ПравоваяПоддержка

Б

Синоним:

LegalSupport

Формат описания:

ПравоваяПоддержка (Заголовок : С; ИмяФайла : С) : Л;

Входные параметры:

Заголовок — заголовок окна с документами

ИмяФайла — имя файла со списком юридических документов

Назначение:

Если на данном компьютере установлена информационная правовая система "Референт" или "Гарант", то функция возвращает значение "ИСТИНА" и открывает ту правовую систему, которая выбрана в диалоге настройки системы бланков. Если ни одна из правовых систем не установлена на компьютере, то система возвращает значение "ЛОЖЬ".

Пример:

```
ПРОЦ КнПрПоддПоНажатию (Отправ : Строка);
    Если ПравоваяПоддержка ("Платежные поручения","8_1") : Выход;
    ! ИСТИНА : Сообщение ("Ни одна правовая база данных не найдена.");
    Конец;
Конец;
```

Процедура РедакторОтменить

Б

Синоним:

EditorCancel

Формат описания:

РедакторОтменить;

Назначение:

Отменяет все изменения, сделанные бланком-редактором в соответствующей записи картотеки.

Примечание:

Может быть использована только в бланке-редакторе.

Пример:

```
ПРОЦ КнОтмПоНажатию (Отправ : Строка);
```

```
    РедакторОтменить;
Конец;
```

Процедура РедакторПрименить

Б

Синоним:

EditorApply

Формат описания:

РедакторПрименить;

Назначение:

Вносит все изменения, выполненные с помощью бланка-редактора, в картотеку.

Примечание:

Может быть использована только в бланке-редакторе.

Пример:

```
ПРОЦ КнПримПоНажатию (Отправ : Строка);
 РедакторПрименить;
Конец;
```

Процедура СнятьМодификацию

Б

Синоним:

ClearModify

Формат описания:

СнятьМодификацию (ИмяБланка : С);

Входные параметры:

ИмяБланка — имя бланка

Назначение:

Если переменные бланка с заданным именем были изменены, то функция **ПеременныеМодифицированы** возвратит значение “ИСТИНА”. Однако, если сначала выполнить процедуру **СнятьМодификацию**, то функция **ПеременныеМодифицированы** возвратит значение “ЛОЖЬ”.

Пример:

```
. . .
```

```
Если (ПеременныеМодифицированы ("ОИ")) :
 СнятьМодификацию ("ОИ") ;
Конец;
```

Процедура СортироватьМассив

Б

Синоним:

SortArray

Формат описания:

СортироватьМассив (ИмяМассива : * [; ПоВозр : Л]);

где * — тип переменной-массива

Входные параметры:

ИмяМассива — имя переменной-массива, описанной в бланке

ПоВозр — необязательный параметр, логическая переменная, если равна "ИСТИНА" (значение по умолчанию), то происходит сортировка по возрастанию, если "ЛОЖЬ" – по убыванию.

Назначение:

Сортирует массив с заданным именем по возрастанию, если параметр *НапрСорт* равен "ИСТИНА" или не задан, по убыванию — если данный параметр равен "ЛОЖЬ".

Пример:

```
proc СортПоНазв(X: Integer; Y: Integer; Double: Logical);
  SortArray(НаименДеталей, НаправлениеСортировки);
  НаправлениеСортировки = Not НаправлениеСортировки;
end;
proc СортИмеется(X: Integer; Y: Integer; Double: Logical);
  SortArray(Есть, True);
  НаправлениеСортировки = Not НаправлениеСортировки;
  message(Отсортированно);
end;
```

Процедура СортироватьСекцию

Б

Синоним:

SortSection

Формат описания:

СортироватьСекцию (УпрПер : Упр | ИмяСекции : С; Столбец : Ц | С [; ПоВозр : Л [УчитыватьРегистр : Л]]);

Входные параметры:

УпрПер — управляющая переменная;

ИмяСекции — строка с именем секции. Указывается, когда на бланке несколько секций с одной и той же управляющей переменной, а нужно отсортировать конкретную

Столбец — либо номер столбца (целое число), либо имя переменной, связанного с этим столбцом массива

ПоВозр — необязательный параметр, логическая переменная, если равна "ИСТИНА" (значение по умолчанию), то происходит сортировка по возрастанию, если "ЛОЖЬ" – по убыванию

УчитыватьРегистр — необязательный параметр, устанавливающий режим чувствительности к регистру для строковых значений при сортировки секции. Если параметр не задан или равен "ИСТИНА", то регистр учитывается при сортировке, иначе ("ЛОЖЬ") — не учитывается.

Назначение:

Сортирует секцию по одному из столбцов по возрастанию, если параметр *ПоВозр* равен "ИСТИНА" или не задан, по убыванию — если данный параметр равен "ЛОЖЬ".

Пример 1:

```
ДатаДок[]    :Дата;
Позиции      :Целое;
Проц ПриНажатии(Кнопка:Строка);
 СортироватьСекцию(Позиции, "ДатаДок");
   -- После этой сортировки док-ты будут расположены по дате возрастания
Конец;
```

Пример 2:

```
var Порядок: Логич;
    ...
proc СортировкаПриНажатии(Row: Integer; Col: Integer; Double: Logical);
  SortSection("сек2", Col, Порядок, False);
    Порядок := not Порядок;
end;
```

Процедура СохранитьРаздел

Б

Синоним:

SaveJurTab

Формат описания:

СохранитьРаздел(Раздел : Строка; Файл : Строка);

Входные параметры:

Раздел — имя раздела табличного журнала;

Файл — полное имя текстового файла. Если задано только имя файла, то файл находится в папке с текущей бухгалтерией.

Назначение:

Сохраняет информацию из раздела табличного журнала в заданном текстовом файле.

Пример:

```
Проц Р1(Объект:Строка);
   СохранитьРаздел("Остатки", "C:\tbw69p\Wrk\Пример\ДанныеТЖ");
   --задан полный путь к файлу
   СохранитьРаздел("Остатки", "ТЖ"); -- задано только имя файла.
Конец;
```

Функция СуммаМассива

Б

Синоним:

SumOfArray

Формат описания:

СуммаМассива (Массив : Ц|Ч [; Длина : Ц]) : Ц|Ч;

Входные параметры:

Массив — массив целого или числового типа, описанный локально в бланке, или поле в подтаблице

Длина — число суммируемых элементов массива

Назначение:

Возвращает сумму значений всех элементов массива (подтаблицы) при отсутствии второго параметра, иначе — возвращает сумму значений начальных элементов массива, заданных во втором параметре.

Примечание:

Если первым параметром является локальный массив бланка, то обязательно нужно указывать второй параметр, т.к. количество элементов у такого массива заранее неизвестно.

Примеры:

```
-- Использование функции в бланке-редакторе при работе с бланками
Сумма[ ] :Число = Позиции[$].Колич*Позиции[$].Цена; -- массив числового типа
СуммаН[ ] :Число = Сумма[$]+Сумма[$]*Позиции[$].НДС/100;
Всего :Число = СуммаМассива(СуммаН,Позиции); -- второй параметр обязателен
-- Использование функции при работе с подтаблицами
```

```
Проц Р1;
    Перем ВсегоКол :Число;
    ВсегоКол = СуммаМассива(Позиции[].Колич); -- Колич – поле подтаблицы
    -- “Позиции”; данная операция имеет смысл, если во всех позициях
    -- документа указана  одна и та же единица измерения
Конец;
```

Функция Текущая

Б

Синоним:

Current

Формат описания:

Текущая [(НомЗаписи : З)] : З;

Входные параметры

НомЗаписи — ссылка на номер записи, которую нужно сделать текущей для бланка-редактора, необязательный параметр.

Назначение:

Если указан необязательный параметр, ссылка на номер записи, которую нужно сделать текущей для бланка - редактора *(НомЗаписи)*, то после выполнения функция возвращает ссылку на текущий номер записи. Если удалось сделать текущей указанную запись, то возвращённое значение совпадет с входным значением.

Если данный параметр не указан, то функция возвращает ссылку на текущую запись, редактируемую бланком-редактором.

Примечание:

Может быть использована только в бланке-редакторе.

Пример:

```
Proc Проц1;
 var vRec: Record;
 vRec = Current;
 Trace(“Текущая запись: “ + RecordToString(vRec));
 vRec = Current(StringTorecord(“{30}”));
 Trace(“Теперь текущая запись: “ + RecordToString(vRec));
end;
```

Функция ТекущееСостояние

Б

Синоним:

CurrentState

Формат описания:

ТекущееСостояние : Ц;

Назначение:

Определяет состояние текущей записи в бланке-редакторе картотеки. Может возвращать одно из 4-х значений:

crdusUnmodified (картсостБезИзменений) — запись не изменялась;

crdusModified (картсостИзменена) — запись изменена;

crdusInserted (картсостВставлена) — запись вставлена;

crdusDeleted (картсостУдалена) — запись удалена;

crdusUndefined (картсостНеопределена) — запись не определена.

Примечание:

Состояние "запись неопределена" может возникать при закрытии бланка-редактора картотеки, если в картотеки ещё нет записей.

Пример:

```
proc ПриСчитывании;
  ПодборБазы(, 0);
  if CurrentState = crdusInserted or CurrentState = crdusUndefined :
    ВидЕд = if(РАБ.ТСС.БИБ.DEF.ВидЕдПоУмолч > 0 and
РАБ.ТСС.БИБ.DEF.ВидЕдПоУмолч < 2, РАБ.ТСС.БИБ.DEF.ВидЕдПоУмолч, 1);
  end;
end;
```

Процедура УдалитьИзМассива

Б

Синоним:

DeleteFromArray

Формат описания:

УдалитьИзМассива (ИмяМассива : *; НомерЭлемента : Ч);

где * — тип переменной-массива

Входные параметры:

ИмяМассива — имя переменной-массива, описанной в бланке

НомерЭлемента — номер позиции удаляемого элемента массива (все последующие элементы сдвигаются на 1 позицию влево)

Назначение:

Удаляет элемент с заданной позицией из массива.

Пример:

```
proc Pr1;
  var Schet : Numeric;
 DeleteFromArray(Schet, 5);
end;
```

Процедура УдалитьИзДинамическогоМассива

Б

Синоним:

DeleteFromDynamicArray

Формат описания:

УдалитьИзДинамическогоМассива (ИмяМассива : *; НомерЭлемента : Ч) [; НомерЭлемента1 : Ч];

где * — тип переменной-массива

Входные параметры:

ИмяМассива — имя переменной-массива, описанной в бланке

НомерЭлемента — номер позиции удаляемого элемента массива (все последующие элементы сдвигаются на 1 позицию влево)

НомерЭлемента1 — номер позиции последнего удаляемого элемента массива. При указании данного параметра будет происходить удаление элементов динамического массива начиная с номера, указанного во втором параметре *НомерЭлемента* по номер, указанный в данном параметре НомерЭлемента1 (все последующие элементы после удаляемой группы сдвигаются на 1 позицию влево).

Назначение:

Удаляет элемент с заданной позицией из динамического массива.

Пример:

```
Dynamic Schet [] : Numeric;
proc aa(s:string);
 DeleteFromDynamicArray(Schet, 5);
end;
```

Процедура УдалитьРамку

Б

Синоним:

DeleteFrame

Формат описания:

УдалитьРамку (УпрПер : Упр | Ц; НомерИт : Ц);

Входные параметры:

УпрПер — управляющая переменная

НомерИт — номер итерации

Назначение:

Удаляет рамку из повторяющейся секции с управляющей переменной *УпрПер*.

Примечание:

После выполнения этой процедуры будет также удален элемент из нужного места во всех массивах, используемых в данной повторяющейся секции.

Пример:

```
УдалитьРамку (УпрПерем, 1);
```

Процедуры и функции управления исключительными ситуациями в бланках

Функция БазисПользовательскихОшибок

Б

Синоним:

ErrUserBasis

Формат описания:

БазисПользовательскихОшибок : Ц;

Назначение:

Для установки исключительной ситуации необходимо присвоить ей код (номер) — целое число из определенного диапазона, чтобы не пересечься со стандартными кодами, присвоенными программистами фирмы "ДИЦ". Данная функция возвращает нижнюю границу диапазона, длина которого составляет 256 значений включительно.

Пример:

```
...
Если (Что-то не так) :
 УстОшибку (БазисПользовательскихОшибок + 1);
 -- устанавливаем свою искл. ситуацию
Конец;
```

Функция КодОшибки

Б

Синоним:

ErrorCode

Формат описания:

КодОшибки : Ц;

Назначение:

Возвращает уникальный номер произошедшей ошибки.

Примечание:

Может быть использована только в блоке **Исключение ... Конец.**

Пример:

```
Проц Р1;
 Перем Номер3, РабОбласть : Целое;
 РабОбласть = ОткрытьРабочуюОбласть("Сотрудники", "");
 -- открываем рабочую область для работы с картотекой "Сотрудники"
  Попытка
  НачатьИзменения(РабОбласть);
   Попытка
    Номер3 = ВставитьЗапись(РабОбласть);
    Номер3 = ЗаписатьПолеС(РабОбласть, Номер3, "ПолнИмя", "ТОО Лютики");
    ЗакончитьИзменения(РабОбласть);
   Исключение
    ОтменитьИзменения(РабОбласть);
    Возбудить;
   Конец;
  Исключение
     Если КодОшибки = 21807 :
      Сообщение("Картотека доступна только для чтения, поэтому запись в
              картотеку не вставлена");
      Истина :
     Возбудить;
  Конец;
Конец;
ЗакрытьРабочуюОбласть(РабОбласть);
Конец;
```

Функция СообщениеОшибки

Б

Синоним:

ErrorMessage

Формат описания:

СообщениеОшибки : С;

Назначение:

Возвращает строку сообщения возникшей ошибки и может быть использована только в блоке Except ... End.

Пример:

```
try
 -- код возбуждающий исключение
except
 Message(Format("Сообщение: %s^MКод: %d", ErrorMessage, ErrorCode), mtError);
end;
```

Процедура УстОшибку

Б

Синоним:

SetError

Формат описания:

УстОшибку (НомерОш : Ц [; Сообщение : С]);

Входные параметры:

НомерОш — номер ошибки

Сообщение — текст выдаваемого сообщения

Назначение:

Устанавливает исключительную ситуацию с заданным уникальный номером и сообщением, если задан второй параметр.

Пример:

```
....
ОшНельзяОткрытьРО :Целое = ErrUserBasis;
ОшРОНеОткрыта :Целое = ErrUserBasis+2;
.....
ФУНК ОткрытьРО (ИмяКарт : Строка, ИмяПоляУп :Строка) :Целое
 ПЕРЕМ РО : Целое;
 РО = 0;
 РО = ОткрытьРабочуюОбласть (ИмяКарт, ИмяПоляУп);
 Если РО = 0 :
  УстОшибку (ОшНельзяОткрытьРО, "Нельзя открыть рабочую область");
 Конец;
Конец;
...
ПРОЦ ЗакрытьРО (РО : Целое);
 Если РО = 0 :
  УстОшибку (ОшНельзяОткрытьРО, "Нельзя открыть рабочую область");
 Конец;
 ЗакрытьРабочуюОбласть (РО);
Конец;
...
```

Процедуры и функции для работы с картотеками

Функция ВзятьГрупповоеИмяПоля

Б

Синонимы:

GetGroupFieldName

Формат вызова:

ВзятьГрупповоеИмяПоля (ИмяКарт : С) : С;

Входные параметры:

ИмяКарт — имя иерархической картотеки

Назначение:

Возвращает имя поля в иерархической картотеке, которое хранит ссылку на запись группы.

Примечание:

Если такое поле не найдено, то возвращает пустую строку.

Пример:

```
ИмяПоляКарт = ВзятьГрупповоеИмяПоля ("Сделка");
```

Функция ВзятьГрупповоеИнфоИмяПоля

Б

Синонимы:

GetGroupInfoFieldName

Формат вызова:

ВзятьГрупповоеИнфоИмяПоля (ИмяКарт : С) : С;

Входные параметры:

ИмяКарт — имя иерархической картотеки

Назначение:

Возвращает информационное имя поля в иерархической картотеке.

Примечание:

Если такое поле не найдено, то возвращает пустую строку.

Пример:

```
ИмяПоляКарт = ВзятьГрупповоеИнфоИмяПоля ("Сделка");
```

Функция ВзятьКартотекуПоССылочномуПолю

Б

Синонимы:

GetCardFileByRefField

Формат вызова:

ВзятьКартотекуПоССылочномуПолю (ИмяКарт : С ; ИмяПоля : С) : С;

Входные параметры:

ИмяКарт — имя картотеки

ИмяПоля — имя поля в картотеке (поле может быть ссылочного, периодического типа или подтаблицей)

Назначение:

Возвращает строку с именем картотеки, на которую ссылается заданное поле в заданной картотеке.

Пример:

```
ИмяПоляКарт = ВзятьКартотекуПоССылочномуПолю ("ТМЦ","Сделка");
```

Функции ВзятьКартПоле (ВзятьКартПолеД, ВзятьКартПолеЗ, ВзятьКартПолеЛ, ВзятьКартПолеС, ВзятьКартПолеЦ, ВзятьКартПолеЧ)

Б

Синонимы:

GetCardFieldD, GetCardFieldR, GetCardFieldL, GetCardFieldS, GetCardFieldI, GetCardFieldN

Формат вызова:

ВзятьКартПоле* (ИмяКарт : С; Ссылка : З; ИмяПоля : С [; Период : Д]) : *

где * — один из типов: Д, З, Л, С, Ц, Ч

Входные параметры:

ИмяКарт — имя картотеки

Ссылка — ссылка на запись

ИмяПоля — имя поля картотеки

Период — необязательный параметр, используется для периодических полей

Назначение:

Возвращает содержимое заданного поля записи в указанной картотеке. Тип функции должен совпадать с ее суффиксом.

Примечание:

Если картотека или запись не найдены, то возбуждаются соответствующие исключения. Если тип поля не совпадает с типом возвращаемого значения, то возникает ошибка.

Если указан дополнительный параметр *Период* при чтении/записи непериодического поля, то возникает ошибка. Если не указан дополнительный параметр *Период* при чтении/записи периодического поля, то он считается равным системной дате.

Пример:

```
Сообщение (ВзятьКартПолеС ("ФизЛицо", Текущая, "Телефон");
```

Функция ВзятьПоляКартотеки

Б

Синоним:

GetCardFileFields

Формат описания:

GetCardFileFields (ИмяКарт : С ; ИмяМассива [ ] : С ; ТипПоля : Ц) : Ц;

Входные параметры:

ИмяКарт — имя картотеки;

ИмяМассива — имя строкового массива, в который передаются поля картотеки;

ТипПоля — переменная, которая может принимать следующие значения предопределенных констант:

mtlREAL — вещественное поле

mtlINTEGER – целое поле

mtlBOOLEAN — логическое

mtlSTRING — строковое

mtlDATE — поле даты

mtlSUBTAB — поля подтаблицы

mtlREF — ссылочные поля

mtlTimed — периодические поля

mtlAll — все поля картотеки.

Назначение:

Функция заполняет строковый массив полями картотеки указанного типа и возвращает длину массива, т.е. количество полей.

Пример:

```
.....
var СписокПолей[] :Строка;
.....
Проц P1(Объект:Строка);
 Перем Колич:Целое;
 -- Все поля картотеки ТМЦ
 Колич=GetCardFileFields(ТМЦ, СписокПолей, mtlAll);
 -- Строковые поля картотеки ТМЦ
 Колич=GetCardFileFields(ТМЦ, СписокПолей, mtlString);
 -- Все числовые поля картотеки ТМЦ
 Колич=GetCardFileFields(ТМЦ, СписокПолей, mtlReal + mtlInteger);
 -- Все поля картотеки ТМЦ кроме подтаблиц и периодических.
 Колич=GetCardFileFields(ТМЦ, СписокПолей, mtlAll - mtlSubTab - mtlTimed);
  Если Колич>0:
   Колич= Альтернатива(Выберите поле картотеки ТМЦ, СписокПолей);
  Конец;
Конец;
```

Функция ВзятьСхему

Б

Синонимы:

GetCheme

Формат вызова:

ВзятьСхему : С ;

Назначение:

Возвращает имя схемы доступа пользователя в сетевой версии или пустую строку в локальной версии.

Пример:

```
Перем Схема : Строка;
Схема = ВзятьСхему;
```

Функция ВзятьТипПоляКартотеки

Б

Синонимы:

GetCardFileFieldType

Формат вызова:

ВзятьГрупповоеИмяПоля (ИмяКарт : С ; ИмяПоля : С) : Ц;

Входные параметры:

ИмяКарт — строка с именем картотеки

ИмяПоля — строка с именем поля

Назначение:

Возвращает целое значение, которое соответствует следующим предопределенным константам:

mtlREAL — вещественное поле

mtlINTEGER — целое поле

mtlBOOLEAN — логическое

mtlSTRING — строковое

mtlDATE — поле даты

mtlSUBTAB — поля подтаблицы

mtlREF — ссылочные поля

mtlTimed — периодические поля

Пример 1:

```
var vType: Integer;
  vType = GetCardFileFieldType("ТМЦ", "Признак");
  -- vType = mtlString
```

Пример 2:

```
var vType: Integer;
  vType = GetCardFileFieldType("ТМЦ", "НДС");
   if vType = mtlTimed: - Периодическое поле
    vType = GetCardFileFieldType("ТМЦ@НДС", "Значение");
    -- vType = mtlReal (НДС[]: Real;)
    end;
```

Функция Видимая

Б

Синонимы:

IsVisible

Формат вызова:

Видимая (ИмяКартотеки : С) : Л ;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Возвращает значение “ИСТИНА” если в картотека с заданным именем видимая или значение “ЛОЖЬ”, если картотека невидимая. В локальной версии всегда возвращает “ИСТИНА”.

Пример:

см. пример фунции **МожноВставить**

Функция ВходитВГруппу

Б

Синонимы:

EntryCardGroup

Формат вызова:

ВходитВГруппу (ИмяКартотеки : С; Запись : З; Группа : З) : Л ;

Входные параметры:

ИмяКартотеки — имя картотеки

Запись — ссылка на запись картотеки (UKEY)

Группа — ссылка на запись группы картотеки (UKEY)

Назначение:

Служит для определения вхождения записи в группу иерархической картотеки. Возвращает значение “ИСТИНА”, если ссылка на запись в картотеке входит в группу по ссылке.

Примечание:

Если ссылка на запись группы = NIL, то функция вернёт значение “ИСТИНА”, в случае корректности ссылок в иерархии, иначе — “ЛОЖЬ”. Это позволяет производить проверку на корректность в иерархии картотеки.

Пример:

```
...
   if Сд <> nil and Док <> nil:
    ДокСд = GetCardFieldR(ДвижениеРесурсов, Док, Сд);
    if EntryCardGroup(СДЕЛКА, ДокСд, Сд) = False :
     message(Несовместимые между собой параметры отчета "Сделка" и
"Документ". Указанный Вами документ ссылается на другую сделку ... );
      return False;
    end;
  end;
  return True;
end;
```

Функция ВыборКартотеки

Б

Синонимы:

ChooseCardFile

Формат описания:

ВыборКартотеки (ИмяКартотеки : С) : Ц;

Входные параметры:

ИмяКартотеки — переменная, в которой будет сохранено значение выбранной картотеки.

Назначение:

Открывает диалог со списком картотек и возвращает константу **кмдВерно (cmOK)**, если картотека выбрана, иначе — **кмдОтказ (cmCancel)**. Если пользователь выбрал папку, то в параметре "ИмяКарт" возвращается имя выбранной картотеки.

Пример:

```
Проц Р1(Объект:Строка);
  Перем ИмяКарт:Строка;
  Если ВыборКартотеки(ИмяКарт) = кмдВерно:
    Трассировка("Имя картотеки = " + ИмяКарт);
  Конец;
Конец;
```

Функция Задан

Ф

Синоним:

Assigned

Формат вызова:

Задан (Поле : Ссылка) : Л;

Входные параметры:

Поле — ссылочное поле картотеки

Назначение:

Функция используется в фильтрах картотек для проверки ссылочных полей и возвращает значение "ИСТИНА", если ссылка определена, иначе — значение "ЛОЖЬ".

Пример:

```
ПРОЦ ПоНажатиюКнопки ( Отправитель : Строка );
 Перем НомРабОбл, k, НомЗап : Ц;
 НомРабОбл = ОткрытьРабочуюОбласть ("Сделка","Контрагент","Задан
(ПредстКонтр)");
 Попытка
  Для k=1 ..Записей(НомРабОбл) Цикл
  МассивСтрок [k] = ВзятьПолеС (НомРабОбл, k, "Контрагент.КорИмя");
 Конец;
  НомЗап = Альтернатива ( "Контрагенты", МассивСтрок, 40);
  Если НомЗап > 0 :
   Сообщение ("Представитель контрагента" + МассивСтрок [НомЗап] + " " +
                 ВзятьПолеС (НомРабОбл, НомЗап, "ПредстКонтр.ФИО" ));
  Илсе;
 Окончание
  ЗакрытьРабочуюОбласть (НомРабОбл);
 Конец;
Конец;
-- на экран выдается диалог со всеми именами контрагентов, для которых
--задан представитель.
-- для выбранного контрагента выдается имя его представителя.
```

Функция ЗаписьСуществует

Б

Синонимы:

RecordExists

Формат вызова:

ЗаписьСуществует (ИмяКартотеки : С; Ссылка : З) : Л;

Входные параметры:

ИмяКартотеки — имя картотеки

Ссылка — ссылка на запись в картотеке (UKEY)

Назначение:

Возвращает значение “ИСТИНА”, если запись по ссылке существует в картотеке.

Пример:

```
Проц Проверка;
Перем Зап : Логич;
 Зап = ЗаписьСуществует ("ТМЦ", Текущая);
  Если Зап = Ложь;
     Сообщение ("Запись с таким UKEY в картотеке нет");
  Илсе
     Сообщение ("Текущая запись нельзя удалить");
  Если
      Возврат Группа;
  Конец;
Конец
```

Функция ЕстьКартотека

Б

Синонимы:

ExistsCardFile

Формат вызова:

ЕстьКартотека (ИмяКартотеки : С) : Л;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Возвращает значение “ИСТИНА”, если картотека есть в базе данных.

Пример:

```
Проц Проверка;
Перем Зап : Логич;
 Зап = ЗаписьСуществует ("ТМЦ", Текущая);
  Если Зап = Ложь;
     Сообщение ("Запись с таким UKEY в картотеке нет");
  Илсе
     Сообщение ("Текущая запись нельзя удалить");
  Если
      Возврат Группа;
  Конец;
Конец
```

Функция Иерархическая

Б

Синонимы:

IsHierarchical

Формат вызова:

Иерархическая (ИмяКартотеки : С) : Л ;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Возвращает значение "ИСТИНА" если картотека с заданным именем является иерархической или значение "ЛОЖЬ", если картотека неиерархическая.

Пример:

```
...
Если Иерархическая ("ТМЦ"):
 Сообщение ("Текущую запись удалить нельзя");
Илсе
...
```

Функция КартГруппа

Б

Синонимы:

IsCardGroup

Формат вызова:

КартГруппа (ИмяКартотеки : С; Ссылка : З) : Л;

Входные параметры:

ИмяКартотеки — имя картотеки

Ссылка — ссылка на запись в картотеке (UKEY)

Назначение:

Возвращает значение "ИСТИНА", если запись по ссылке в картотеке является группой.

Пример:

```
Проц ПроверкаНаГруппу;
 Перем Группа : Логич;
 Группа = КартГруппа ("ТМЦ", Текущая);
 Если Группа = Истина;
```

```
    Сообщение ("Текущую запись нельзя удалить");
  Илсе
    Сообщение ("Текущую запись нельзя удалить");
  Если
  Возврат Группа;
 Конец;
```

Функция КартотекаЖурнала

Б

Синоним:

JournalCardfile

Формат вызова:

КартотекаЖурнала (ИмяЖурнала : С) : С;

Входные параметры:

ИмяЖурнала — имя jdf-файла

Назначение:

Возвращает имя картотеки, с которой связан передаваемый jdf-файл, только в том случае, если до этого была выполнена обработка журналов. Для несуществующих или некартотечных журналов возвращает пустую строку.

Примеры:

```
Стр1 = КартотекаЖурнала("Платежные поручения.jdf"); -- Поруч
Стр2 = КартотекаЖурнала("Платежные поручения"); -- Пустая строка
Стр3 = КартотекаЖурнала("C:\TBW65\WRK\Пример\Приход товара.jdf");
-- ДвижениеТМЦ
```

Функция КартотекаПризнака

Б

Синоним:

SignCardFile

Формат вызова:

КартотекаПризнака (АналитПризнак : С [; ОбработатьАналит : Л) : С;

Входные параметры:

АналитПризнак — идентификатор аналитического признака

ОбработатьАналит — логический параметр, если равен "ИСТИНА" или не указан, то вызывает внутри себя функцию **ОбработатьАналит (RefreshAna)** для актуализации дерева аналитики; если

равен “ЛОЖЬ”, то функция для обновления аналитики не вызывается

Примечание:

Чтобы ускорить работу функции **КартотекаПризнака** следует второй необязательный параметр для обновления дерева аналитики установить значением “ЛОЖЬ”, а в коде бланка, до начала работы циклов, в которых используется данная функция, вызвать функцию **ОбработатьАналит**.

Назначение:

По идентификатору аналитического признака возвращает имя его картотеки или пустую строку, если признак не найден или отсутствует в картотеке.

Пример:

```
if not Автообработка then
  RefreshAna;
for Sign vAna = “*”, Sort do
  vCardFile = SignCardFile(vAna, False);
--Можно указать False, т.к. вызвали раньше RefreshAna
  if vCardFile <> «»:
     ...
   end;
     ...
end;
```

Функция КартотекаПризнакЗаписи

Б

Синоним:

CardFileRecSign

Формат вызова:

КартотекаПризнакЗаписи (ИмяКартотеки : С; Поле : С; Справочник : С; Запись : З [; ГрупДоп : С]) : С;

Входные параметры:

ИмяКартотеки — имя картотеки

Поле — имя поля в картотеке из которого берется аналитический признак

Справочник — имя картотечного справочника аналитических признаков

Запись — ссылка на запись картотеки из которой берется аналитический признак

ГруппДоп — строковое значение, которое будет добавлено в конец строки возвращаемого значения функции, в случае, если запись, на которую указывает параметр *Запись*, является группой в иерархической картотеке. Может использоваться, что бы отличать картотечные признаки записи от групп записей. Если параметр не указан, то функция работает как и раньше.

Назначение:

Функция возвращает строку с формированным аналитическим признаком из справочника и значения поля картотеки. Если картотека — иерархическая, то признак формируется с учётом иерархии.

Примеры:

```
Trace(CARDFILERECSIGN("КОНТРАГЕНТ", "ПРИЗНАК", "К", StringToRecord("{21}")));
--К.Ю.00021
Trace(CARDFILERECSIGN("ТМЦ", "ПРИЗНАК", "ТМЦ", {50}, ".*"));
-- ТМЦ.00010.00050.* в случае, если запись с UKey = {50} явл. группой.
Trace(CARDFILERECSIGN("ТМЦ", "ПРИЗНАК", "ТМЦ", {50}, ".*"));
-- ТМЦ.00010.00050  в случае, если запись с UKey = {50} НЕ явл. группой.
```

Функция МожноВставить

Б

Синонимы:

CanInsert

Формат вызова:

МожноВставить (ИмяКартотеки : С) : Л ;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Возвращает значение "ИСТИНА" если в картотеку с заданным именем можно добавлять записи или значение "ЛОЖЬ", если нельзя.

Пример:

```
proc test;
   var МожноВидеть:logical;
   var МожноИзменять:logical;
   var МожноДобавлять:logical;
   var МожноУдалить:logical;
   var ТолькоЧитать:logical;
   МожноВидеть=IsVisible("ДвижениеРесурсовБух");
   МожноИзменять=CanUpdate("ДвижениеРесурсовБух");
   МожноДобавлять=CanInsert("ДвижениеРесурсовБух");
   МожноУдалить=CanDelete("ДвижениеРесурсовБух");
   ТолькоЧитать=IsReadOnly("ДвижениеРесурсовБух");
```

```
        МожноВидеть=true;
        МожноИзменять=true;
        МожноДобавлять=true;
        МожноУдалить=true;
        ТолькоЧитать=false;
    end;
```

Функция МожноИзменить

Б

Синонимы:

CanUpdate

Формат вызова:

МожноИзменить (ИмяКартотеки : С) : Л ;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Возвращает значение “ИСТИНА” если в картотеке с заданным именем можно изменять записи или значение “ЛОЖЬ”, если нельзя.

Пример:

см. пример фунции **МожноВставить**

Функция МожноУдалять

Б

Синонимы:

CanDelete

Формат вызова:

МожноУдалять (ИмяКартотеки : С) : Л ;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Возвращает значение “ИСТИНА” если в картотеке с заданным именем можно удалять записи или значение “ЛОЖЬ”, если нельзя.

Пример:

см. пример фунции **МожноВставить**

Процедура ОткрытьКартотеку

Б

Синоним:

OpenCardFile

Формат вызова:

ОткрытьКартотеку (Имя : С [; ДопФ : С] [; Ссылка : З]);

Входные параметры:

Имя — имя файла настроек или имя картотеки

ДопФ — дополнительный фильтр

Ссылка — ссылка на запись картотеки, которая должна быть текущей при открытии окна

Назначение:

Открывает окно картотеки со стандартными настройками, если в первом параметре задано имя картотеки. Если задано имя файла настроек, то будут использоваться настройки, установленные пользователем.

Примечания:

1. При наличии второго параметра записи в картотеке будут отображаться с учетом основного и дополнительного фильтров.
2. Если в третьем параметре задана ссылка на недоступную запись, то сначала проверяется, доступна ли запись при снятом фильтре. При положительном результате после предупреждения фильтр выключается, и запись становится текущей, иначе — фильтр не выключается, а выдается предупреждение и текущей становится первая запись.
3. Настройки могут сохраняться как по умолчанию, так и по явному запросу пользователя в диалоге “Настройки параметров картотеки”.

Пример:

```
ОткрытьКартотеку ("Crd1", "", R); -- открываем немодальное окно и
-- подсвечиваем в нем запись, только что выбранную пользователем
```

Функция ОткрытьКартотеку

Б

Синоним:

OpenCardFile

Формат вызова:

ОткрытьКартотеку (Имя : С [; ДопФ : С] [; Ссылка : З] [; Выбор : Ц]) : З;

Входные параметры:

Имя — имя файла настроек или имя картотеки

ДопФ — дополнительный фильтр

Ссылка — ссылка на запись картотеки, которая должна быть текущей при открытии окна

Выбор — данный параметр может принимать следующие значения:

- **crdRecordOnly (картТолькоЗапись)** — можно выбрать только запись — значение по умолчанию;
- **crdGroupOnly (картТолькоГруппу)** — можно выбрать только группу;
- **crdAnyRecord(картЛюбаяЗапись)** — можно выбрать любую запись.

Назначение:

Открывает модальное окно картотеки и возвращает ссылку на выбранную запись. Если в первом параметре задано имени картотеки, то окно картотеки будет открываться со стандартными настройками. Если задано имя файла настроек, то будут использоваться настройки, установленные пользователем.

Примечания:

1. При наличии второго параметра записи в картотеке будут отображаться с учетом основного и дополнительного фильтров. Если в данном параметре строка фильтра начинается с двух минусов (“--”), то данный фильтр будет являться не дополнительным а основным, т.е. перекрывать фильтр в картотеке, если он там был установлен ранее.
2. Если в третьем параметре задана ссылка на недоступную запись, то сначала проверяется, доступна ли запись при снятом фильтре. При положительном результате после предупреждения фильтр выключается, и запись становится текущей, иначе — фильтр не выключается, а выдается предупреждение и текущей становится первая запись.
3. Если задан четвертый параметр *Выбор*, то картотека открывается с заданным режимом выбора записей: в ней можно выбрать любую запись (обычную и группу) или обычную запись, или группу.
4. Если в модальном окне картотеки пользователь закрыл окно без выбора записи, то функция возвращает константу **Пусто**.
5. Настройки могут сохраняться в файле с расширением *.bro как по умолчанию, так и по явному запросу пользователя из диалога “Настройки параметров картотеки”.

Пример:

```
vRec = OpenCardFile("ФИЗЛИЦО", "", nil, crdAnyRecord);
```

Функция ПризнакВЗапись

Б

Синоним:

SignToRecord

Формат вызова:

ПризнакВЗапись (ИмяПризнака : С) : З;

Входные параметры:

ИмяПризнака — имя (идентификатор) аналитического признака

Назначение:

По идентификатору аналитического признака возвращает указатель на запись, в которой он описан, или “ПУСТО”, если признак не найден или отсутствует в картотеке.

Пример:

```
Зап = ПризнакВЗапись ("Ф.002");
```

Функция ТолькоНаЧтение

Б

Синонимы:

IsReadOnly

Формат вызова:

ТолькоНаЧтение (ИмяКартотеки : С) : Л ;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Возвращает значение “ИСТИНА” если в картотека с заданным именем доступна только на чтение или значение “ЛОЖЬ”, если в ней можно производить какие-либо изменения.

Пример:

см. пример фунции **МожноВставить**

Функция ПризнакЗаписи

Б

Синоним:

RecSign

Формат вызова:

ПризнакЗаписи (ИмяКартотеки : С; Запись : З; НомерПризнака : Ц) : С;

Входные параметры:

ИмяКартотеки — имя картотеки

Запись — запись картотеки

НомерПризнака — номер аналитического признака

Назначение:

Возвращает идентификатор аналитического признака, созданного заданной записью указанной картотеки, по его порядковому номеру.

Примечание:

С помощью одной записи картотеки-аналитики можно добавить в список признаков один или несколько аналитических признаков (зависит от количества установленных пользователем флагов в заполняемом бланке-редакторе). Каждому такому признаку присваивается порядковый номер, начиная от 1. Общее количество созданных одной записью признаков определяется функцией **ЧислоПризнаковЗаписи**.

Пример:

```
Трассировка ("Идентификатор признака "+ПризнакЗаписи("ТМЦ", Текущая, 1));
```

Функция ЧислоПризнаковЗаписи

Б

Синоним:

RecSignsCount

Формат вызова:

ЧислоПризнаковЗаписи (ИмяКартотеки : С; Запись : З) : Ц;

Входные параметры:

ИмяКартотеки — имя картотеки

Запись — запись картотеки

Назначение:

Возвращает количество аналитических признаков, созданных записью с заданной ссылкой и хранящейся в указанной картотеке.

Примечание:

С помощью одной записи картотеки-аналитики можно добавить в список признаков один или несколько аналитических признаков (зависит от количества установленных пользователем флагов в заполняемом бланке-редакторе). Каждому такому признаку присваивается порядковый номер, начиная от 1. Общее количество созданных одной записью признаков определяется данной функцией.

Пример:

```
Проц Р1 (ИмяКарт : Строка);
-- определяет, используется ли данная запись картотеки для формирования признаков
 Перем I, НомРабОбл, НомЗап : Целое;
 Перем Зап : Запись;
 Перем Признак : Строка;
 НомРабОбл = ОткрытьРабочуюОбласть(ИмяКарт, "");
 Попытка
    Для НомЗап = 1..Записей(НомРабОбл) Цикл
        Зап = ВзятьКлючЗаписи (НомРабОбл, НомЗап);
        Для I = 1..ЧислоПризнаковЗаписи (ИмяКарт, Зап) Цикл
            Признак = ПризнакЗаписи (ИмяКарт, Зап, I);
            Трассировка (Признак+"-"+Если(ПризнакИспользован(Признак),
                        " использовался в ПБ", " не использовался в ПБ"));
        Конец;
        Если ЧислоПризнаковЗаписи(ИмяКарт, Зап)<1
            : Трассировка("Запись - "+ЗаписьВСтроку(Зап)+" не участвует в
                формировании списка признаков");
        Конец;
    Конец;
 Окончание
    ЗакрытьРабочуюОбласть(НомРабОбл);
 Конец;
Конец;
```

Процедуры и функции для работы с картотеками через рабочие области

Функция ВзятьГруппа

Б

Синонимы:

GetGroup

Формат описания:

ВзятьГруппа (НомерРО : Ц; НомЗаписи : Ц) : Л;

Входные параметры:

НомерРО — номер рабочей области, из которой считывается запись

НомерЗап — номер записи, из которой считывается информация

Назначение:

Возвращает значение "ИСТИНА", если запись с заданным номером является группой.

Пример:

```
Proc P1;
 Перем РабОбласть : Ц;
 Перем А : Л;
 Перем НомерЗап : Ц;
 РабОбласть = ОткрытьРабочуюОбласть("ТМЦ", "Признак");
 А = ВзятьГруппа (РабОбласть, 1);
  -- А = True;
end
```

Процедура ВзятьКакМассив

Б

Синоним:

GetAsArray

Формат описания:

ВзятьКакМассив (НомерРО : Ц; НомерЗап: Ц; ИмяБл : С;
ИмяСтруктПоля : С; СписокПер : С);

Входные параметры:

НомерРО — номер рабочей области, из которой считывается запись

НомерЗап — номер записи, из которой считывается информация

ИмяБл — строковое выражение — имя бланка, который заполняется, или пустая строка (для текущего бланка)

ИмяСтруктПоля — строковое выражение — имя структурного поля, из которого считывается информация

СписокПер — строковое выражение, в котором через вертикальную черту перечислены переменные-массивы.

Назначение:

Процедура **ВзятьКакМассив** является обратной к процедуре **ЗаписатьКакМассив** и читает структурное поле в переменные-массивы бланка. Параметры данной процедуры аналогичны параметрам процедуры **ЗаписатьКакМассив**.

Примечание:

Данная процедура является обратной к процедуре **ЗаписатьКакМассив**, поэтому имена переменных-массивов должны совпадать с соответствующими именами полей в подтаблице.

Пример:

```
ПРОЦ КнопкаТестПоНажатию (ИмяОбъекта : Строка);
 Перем НомерРО : Целое;
 НомерРО = ОткрытьРабочуюОбласть ("Требов", "");
 ВзятьКакМассив (НомерРО, 1, "", "Допол", "Дата¦Сумма¦Сумма0");
 -- Допол – имя подтаблицы в картотеке "Требов".
Конец;
```

Функция ВзятьКакМассив

Б

Синоним:

GetAsArray

Формат описания:

ВзятьКакМассив (НомерРО : Ц; НомерЗап: Ц; ИмяБл : С;
ИмяСтруктПоля : С; СписокПер : С) : Ц;

Входные параметры:

НомерРО — номер рабочей области, из которой считывается запись

НомерЗап — номер записи, из которой считывается информация

ИмяБл — строковое выражение — имя бланка, который заполняется, или пустая строка (для текущего бланка)

ИмяСтруктПоля — строковое выражение — имя структурного поля, из которого считывается информация

СписокПер — строковое выражение, в котором через вертикальную черту перечислены переменные-массивы.

Назначение:

Функция **ВзятьКакМассив** возвращает количество элементов в переменные-массивы бланка, считываемые из структурного поля.

Пример:

```
ПРОЦ КнопкаТестПоНажатию (ИмяОбъекта : Строка);
 Перем НомерРО : Целое;
 Перем Колич : Целое;
 НомерРО = ОткрытьРабочуюОбласть ("Требов", "");
 Колич = ВзятьКакМассив (НомерРО, 1, "", "Допол", "Дата¦Сумма¦СуммаО");
 -- Допол - имя подтаблицы в картотеке "Требов".
Конец;
```

Функция ВзятьКлючЗаписи

Б

Синоним:

GetRecordKey

Формат описания:

ВзятьКлючЗаписи (НомерРО : Ц; НомерЗ : Ц) : З;

Входные параметры:

НомерРО — номер рабочей области

НомерЗ — номер записи

Назначение:

Возвращает ссылку на запись с заданным номером в картотеке, над которой открыта заданная рабочая область.

Пример:

```
-- Берем указатель на 1-ую запись
Зап = ВзятьКлючЗаписи (РабОбласть, 1);
```

Функция ВзятьНомерЗаписи

Б

Синоним:

GetRecordNumber

Формат описания:

ВзятьНомерЗаписи (НомерРО : Ц; Ключ : З) : Ц;

Входные параметры:

НомерРО — номер рабочей области

Ключ — ключ записи

Назначение:

Возвращает номер записи в рабочей области по ее ключу. Если записи с таким ключом в рабочей области нет, возвращает ноль.

Пример:

```
НомерЗап = ВзятьНомерЗаписи (РабОбл, Ключ);
```

Функции ВзятьПоле (ВзятьПолеД, ВзятьПолеЗ, ВзятьПолеЛ, ВзятьПолеС, ВзятьПолеЦ, ВзятьПолеЧ, ВзятьПолеВ)

Б

Синонимы:

GetFieldD, GetFieldR, GetFieldL, GetFieldS, GetFieldI, GetFieldN, GetFieldT

Формат описания:

ВзятьПоле* (НомерРО : Ц; НомерЗ : Ц; ИмяПоля : С [; Дата : Д]) : *;

где * — один из типов: Д, З, Л, С, Ц, Ч

Функция **ВзятьПолеВ** имеет тип Ц.

Входные параметры:

НомерРО — номер рабочей области

НомерЗ — номер записи

ИмяПоля — имя поля картотеки, над которой открыта рабочая область

Дата — дата (для периодических полей)

Назначение:

Возвращает содержимое заданного поля записи с заданным номером в заданной рабочей области. Для периодических полей может использоваться дополнительный параметр *Дата*.

Функция **ВзятьПолеВ (GetFieldT)** возвращает число секунд от начала суток из поля картотеки, имеющего тип “Дата/Время”.

Примечания:

1. Если тип поля не совпадает с типом возвращаемого значения, то возникает ошибка.

2. Если указан дополнительный параметр *Дата* при чтении/записи непериодического поля, то возникает ошибка. Если не указан дополнительный параметр *Дата* при чтении/записи периодического поля, то считается, что он равен системной дате.

3. Функция **ВзятьПолеЗ (GetFieldR)** может использоваться в рабочей области открытой функцией **ОткрытьРабочуюОбластьSQL (OpenWorkAreaSQL)** на целые поля.

Пример:

```
-- читаем текущий курс USD
КурсUSD = ВзятьПолеЦ (РабОбласть, 1, "Курс", Сегодня);
```

Процедура ВзятьПоляВПеременные

Б

Синоним:

GetFieldsVars

Формат описания:

GetFieldsVars(НомерРО : Ц; НомерЗ : Ц; ИмяБланка: С; Поля: С | ДлС);

Входные параметры:

НомерРО — номер рабочей области

НомерЗ — номер записи в рабочей области

ИмяБланка — строка с именем бланка с переменными. Если пусто, то текущий бланк

Поля — строка со списком переменных и полей разделённых вертикальной чертой. Формат:
"<ИмяПер1>=<ИмяПоля1>[|<ИмяПерN>=<ИмяПоляN>]",

Где:

<ИмяПер> — имя переменной в которую будут записываться значения из поля <FieldName>.

Назначение:

Процедура для загрузки в переменные бланка полей картотеки из рабочей области. Записывает значение заданного поля записи с заданным номером в заданной рабочей области в переменную бланка.

Пример:

```
var Description: String;
var Sign: Integer;
Проц Кнопка27_ПриНажатии(Отправитель: Строка);
 var vWA: Integer;
```

```
    vWA = OpenWorkArea("ТМЦ", "");
   try
   if Records(vWA) > 0 :
    GetFieldsVars(vWA, 1, "", "Description=КОММЕНТ:Sign=ПРИЗНАК");
    Trace("КОММЕНТ=»' + Description + "«, ПРИЗНАК=»' + Str(Sign) + "«");
   end;
   finally
    CloseWorkArea(vWA);
  end;
end;
```

Функция ВзятьФильтр

Б

Синоним:

GetFilter

Формат описания:

ВзятьФильтр (НомерРО : Ц) : С;

Входные параметры:

НомерРО — номер рабочей области

Назначение:

Возвращает текущий фильтр, установленный для заданной рабочей области.

Пример:

```
...
Попытка
   УстФильтр (РабОбласть, "Дата = 24.11.97");
   Фильтр = ВзятьФильтр (РабОбласть);
   -- Фильтр = "Дата = 24.11.97"
Окончание
...
```

Функция ВставитьЗапись

Б

Синоним:

InsertRecord

Формат описания:

ВставитьЗапись (НомерРО : Ц [, НомерЗаписи : Ц]) : Ц;

Входные параметры:

НомерРО — номер рабочей области

НомерЗаписи — номер записи в подтаблице, под которым она будет вставлена

Назначение:

Вставляет новую пустую запись в рабочую область с заданным номером и возвращает ее номер. Если запись не удовлетворяет текущему фильтру, то функция возвращает ноль.

Примечание:

В запись с номером "0" можно записывать информацию.

Пример:

```
-- вставляем пустую запись в рабочую область
Номер3 = ВставитьЗапись (РабОбласть);
```

Функция ВставитьЗаписьПоКлючу

Б

Синоним:

InsertRecordByKey

Формат описания:

ВставитьЗаписьПоКлючу (НомерРО : Ц; Ключ : З) : Ц;

Входные параметры:

НомерРО — номер рабочей области

Ключ — уникальный ключ записи UKEY

Назначение:

Вставляет новую пустую запись с заданным уникальным ключом в заданную рабочую область и возвращает номер записи.

Примечание:

Если запись с заданным ключом уже существует, то возникает ошибка.

Пример:

```
-- вставляем пустую запись в рабочую область
Номер3 = ВставитьЗаписьПоКлючу (РабОбл, UkeyНом);
```

Функция ЗаблокироватьЗапись

Б

Синоним:

LockRecord

Формат описания:

ЗаблокироватьЗапись (РО : Ц; НомерЗаписи : Ц; Время : Ц): Л;

Входные параметры:

РО — рабочая область;

НомерЗаписи — номер записи в рабочей области;

Время — время ожидания блокировки записи в секундах (0 - не ждать);

Назначение:

Возвращает значение "ИСТИНА", если запись удалось заблокировать и "ЛОЖЬ", если запись заблокировать не удалось. Блокировка с записей снимается после подтверждения или отката изменений в рабочей области (см. функции **EndChanging()**, **CancelChanging()**). Запись автоматически блокируется, если в поле записи уже были произведены изменения в рамках одной транзакции.

Примечание:

Если запись заблокирована с помощью функции LockRecord, то повторный вызов данной функции вернет значение "ЛОЖЬ".

Пример 1:

```
...
if LockRecord(РабОбл, 2, 10): – True
 if LockRecord(РабОбл, 2, 10): – False
end;
 end;
```

Пример 2:

```
proc OnClick(S: String);
 var vWA: Integer; – Дескриптор рабочей области
 var vRN: Integer; – Номер редактируемой записи
 vWA = OpenWorkArea("БАНК_СЧЕТА", "", "");
 try
  BeginChanging(vWA);
   try
    if LockRecord(vWA, 2, 10):  -- Ожидаем блокировку записи в течении 10
                                -- секунд
     vRN = PutFieldS(vWA, 2, "БАНКИМЯ", "МОСПРОМСТРОЙБАНКГАРАНТ");
       fi;
  except
```

```
  CancelChanging(vWA);
  Raise;
    end;
  EndChanging(vWA);
 finally
CloseWorkArea(vWA);
  end;
end; - OnClick
```

Процедура ЗакончитьИзменения

Б

Синоним:

EndChanging

Формат описания:

ЗакончитьИзменения (НомерРО : Ц);

Входные параметры:

НомерРО — номер рабочей области

Назначение:

Вносит все изменения, сделанные в результате редактирования записей в рабочей области, в картотеку.

Примечание:

Перед внесением изменений в картотеке необходимо вызвать функцию **НачатьИзменения;** проделав необходимые правки и изменения, вызовите функцию **ЗакончитьИзменения.** Изменения в картотеку вносятся только после выполнения этой функции. При желании можно отменить все проделанные исправления в картотеке, вызвав процедуру **ОтменитьИзменения** (*см.* пример).

Пример:

```
...
Попытка
   НачатьИзменения (РабОбласть);
   Попытка
      Номер3 = ВставитьЗапись (РабОбласть);
      Номер3 = ЗаписатьПолеЦ (РабОбласть, Номер3, “Название”, “ТОО
Лютики”);
      ЗакончитьИзменения (РабОбласть);
   Исключение
      ОтменитьИзменения (РабОбласть);
      Возбудить;
   Конец;
Окончание
...
```

Процедура ЗакрытьРабочуюОбласть

Б

Синоним:

CloseWorkArea

Формат описания:

ЗакрытьРабочуюОбласть (НомерРО : Ц);

Входные параметры:

НомерРО — номер рабочей области

Назначение:

Закрывает рабочую область с заданным номером.

Пример:

```
-- открываем рабочую область над картотекой 'NNN'
РабОбласть = ОткрытьРабочуюОбласть ('NNN', '');
-- операторы
-- Закрываем рабочую область
ЗакрытьРабочуюОбласть (РабОбласть);
```

Функция ЗаписатьВПоляПеременные

Б

Синоним:

PutFieldsVars

Формат описания:

PutFieldsVars (НомерРО : Ц; НомерЗ : Ц; ИмяБланка: С; Поля: С | ДлС) : Ц;

Входные параметры:

НомерРО — номер рабочей области

НомерЗ — номер записи в рабочей области

ИмяБланка — строка с именем бланка с переменными. Если пусто, то текущий бланк

Поля — строка со списоком переменных и полей разделённых вертикальной чертой. Формат:
"<ИмяПер1>=<ИмяПоля1>[|<ИмяПерN>=<ИмяПоляN>]",

где:

<ИмяПер> — имя переменной из которой будут записываться значения в поля <FieldName>.

Назначение:

Записывает значение переменной в заданное поле записи с заданным номером в заданной рабочей области. Возвращает номер записи.

Пример:

```
var Description: String;
var Sign: Integer;
Проц Кнопка27_ПриНажатии(Отправитель: Строка);
 var vWA, vRec: Integer;
  vWA = OpenWorkArea("ТМЦ", "");
   try
    if Records(vWA) > 0 :
    BeginChanging(vWA);
      try
      vRec = PutFieldsVars(vWA, 1, "", "Description=КОММЕНТ:Sign=ПРИЗНАК");
      except
       CancelChanging(vWA);
        Raise;
       end;
       EndChanging(vWA);
    end;
   finally
     CloseWorkArea(vWA);
  end;
end;
```

Функция ЗаписатьГруппу

Б

Синонимы:

PutIsGroup

Формат описания:

ЗаписатьГруппу (НомерРО : Ц; НомЗаписи : Ц; Группа : Л);

Входные параметры:

НомерРО — номер рабочей области, из которой считывается запись

НомерЗап — номер записи, в которую записывается признак группы

Назначение:

Записывает признак группы заданному номеру записи в заданной рабочей области, если значение третьего параметра равно "ИСТИНА", если "ЛОЖЬ" — признак группы снимается.

Пример:

```
Proc P1;
var РабОбласть : Integer;
```

```
var НомерЗап : Integer;
 РабОбласть = ОткрытьРабочуюОбласть("ТМЦ", "Признак");
  try
  НачатьИзменения (РабОбласть);
    try
    ЗаписатьГруппу (РабОбласть, 1, ИСТИНА);
    Except
    ОтменитьИзменения (РабОбласть);
    end;
    Raise;
  ЗакончитьИзменения(РабОбласть);
  Finally
   ЗакрытьРабочуюОбласть (РабОбласть);
   end;
end;
```

Процедура ЗаписатьКакМассив

Б

Синоним:

PutAsArray

Формат описания:

ЗаписатьКакМассив (НомерРО : Ц; НомерЗап : Ц; ИмяБл : С; ИмяСтруктПоля : С; СписокПер : С);

Входные параметры:

НомерРО — номер рабочей области, в которую производится запись

НомерЗап — номер записи, в которую заносится информация

ИмяБл — строковое выражение — имя бланка, из которого берутся переменные, или пустая строка (для текущего бланка)

ИмяСтруктПоля — строковое выражение — имя структурного поля, в которое производится запись

СписокПер — строковое выражение, в котором через вертикальную черту перечислены переменные-массивы, вставляемые в структуру

Назначение:

Применяется для записи циклов бланков в картотеку и может использоваться в процедурах и функциях бланков. Процедура записывает несколько переменных-массивов в картотеку в виде единого структурного поля.

Примечание:

Новое поле возникает во всех записях картотеки, но информация из переменных бланка будет перенесена только в одну запись, чей

номер указан в этом параметре. Если записи еще нет, то ее необходимо создать функцией **ВставитьЗапись**.

Пример:

```
ПРОЦ КнопкаТестПоНажатию (ИмяОбъекта : Строка);
 Перем НомерРО : Целое;
 НомерРО = ОткрытьРабочуюОбласть ("Требов", "");
 ЗаписатьКакМассив (НомерРО, 1, "", "Допол", "Дата¦Сумма¦СуммаО");
 -- Допол - имя подтаблицы в картотеке "Требов".
Конец;
```

Функция ЗаписатьКакМассив

Б

Синоним:

PutAsArray

Формат описания:

ЗаписатьКакМассив (НомерРО : Ц; НомерЗап : Ц; ИмяБл : С; ИмяСтруктПоля : С; СписокПер : С) : Ц;

Входные параметры:

НомерРО — номер рабочей области, в которую производится запись

НомерЗап — номер записи, в которую заносится информация

ИмяБл — строковое выражение — имя бланка, из которого берутся переменные, или пустая строка (для текущего бланка)

ИмяСтруктПоля — строковое выражение — имя структурного поля, в которое производится запись

СписокПер — строковое выражение, в котором через вертикальную черту перечислены переменные-массивы, вставляемые в структуру

Назначение:

Возвращает количество элементов в переменных-массивах, записываемых в картотеку в виде единого структурного поля.

Пример:

```
ПРОЦ КнопкаТестПоНажатию (ИмяОбъекта : Строка);
 Перем НомерРО : Целое;
 Перем Колич : Целое;
 НомерРО = ОткрытьРабочуюОбласть ("Требов", "");
 Колич = ЗаписатьКакМассив (НомерРО, 1, "", "Допол", "Дата¦Сумма¦СуммаО");
 -- Допол - имя подтаблицы в картотеке "Требов".
Конец;
```

Функции ЗаписатьПоле (ЗаписатьПолеД, ЗаписатьПолеЗ, ЗаписатьПолеЛ, ЗаписатьПолеС, ЗаписатьПолеЦ, ЗаписатьПолеЧ)

Б

Синонимы:

PutFieldD, PutFieldR, PutFieldL, PutFieldS, PutFieldI, PutFieldN

Формат описания:

ЗаписатьПоле* (НомерРО : Ц; НомерЗ : Ц; ИмяПоля : С; Значение : * [, Дата : Д]) : Ц;

где * — один из типов: Д, З, Л, С, Ц, Ч

Входные параметры:

НомерРО — номер рабочей области

НомерЗ — номер записи

ИмяПоля — имя поля картотеки, над которой открыта рабочая область

Значение — значение поля

Дата — дата (для периодических полей)

Назначение:

Записывает значение в заданное поле записи с заданным номером в заданной рабочей области. Для периодических полей может использоваться дополнительный параметр *Дата*. Возвращает номер записи.

Примечания:

1. Если тип поля не совпадает с типом заданного значения, то возникает ошибка.

2. Если указан дополнительный параметр *Дата* при чтении/записи непериодического поля, то возникает ошибка. Если не указан дополнительный параметр *Дата* при чтении/записи периодического поля, то считается, что он равен системной дате.

Пример:

```
Номер3 = ЗаписатьПолеЦ (РабОбласть, 1, "Курс", 5940, Сегодня);
```

Функция Записей

Б

Синоним:

Records

Формат описания:

Записей (НомерРО : Ц) : Ц;

Входные параметры:

НомерРО — номер рабочей области, необязательный параметр

Назначение:

Если параметр *НомерРО* не указан, то функция возвращает число записей на картотеку, для которой выполняется вызвавший её бланк-редактор.

Если параметр *НомерРО* указан, то функция возвращает число записей в рабочей области с заданным номером.

Пример:

```
Proc Проц1;
 var vRO1: Integer;
  Trace("Записей в картотеки БАНК-СЧЕТА для бланка: " + Str(Records));
  vRO1 = OpenWorkArea("БАНК-СЧЕТА", "", "НАШ");
    try
    trace("Записей в рабочей области на картотеку БАНК-СЧЕТА: " +
         Str(Records(vRO1)));
   finally
  CloseWorkArea(vRO1);
   end;
 end;
```

Функция КартотекаРабочейОбласти

Б

Синоним:

WorkAreaCardFile

Формат описания:

КартотекаРабочейОбласти (НомерРО : Ц) : С;

Входные параметры:

НомерРО — номер рабочей области

Назначение:

По заданному номеру рабочей области возвращает имя картотеки, которая была в открыта в этой рабочей области.

Пример:

```
ПРОЦ КнопкаПоНажатию (Sender :Строка);
 Перем РабОбл :Целое;
 -- Открытие рабочей области для работы с картотекой "ЮрЛицо"
 РабОбл = ОткрытьРабочуюОбласть ("ЮрЛицо","КорИмя");
```

```
Попытка
 Трассировка (КартотекаРабочейОбласти (РабОбл));
   -- В окне сообщений появляется строка "ЮрЛицо"
Окончание
 -- Закрытие рабочей области
 ЗакрытьРабочуюОбласть (РабОбл);
Конец;
КОНЕЦ;
```

Процедура КопироватьЗапись

Б

Синоним:

CopyRecord

Формат описания:

КопироватьЗапись (НомерРООткуда : Ц; НомерРОКуда : Ц; НомерЗОткуда) : Ц; НомерЗКуда : Ц);

Входные параметры:

НомерРООткуда — номер рабочей области откуда копируется запись

НомерРОКуда — номер рабочей области куда копируется запись

НомерЗОткуда — номер записи в рабочей области откуда копируется запись

НомерЗКуда — номер записи в рабочей области куда копируется запись

Назначение:

1. Копирование записи из одной рабочей области в другую для картотек с одинаковой структурой.
2. Дублирование записей в картотеке.

Примеры:

Дублирование первой записи в картотеке:

```
proc OnDublRecord(S: String);
var vRO1: Integer;
var vRecNum: Integer;
var i: Integer;
 vRO1 = OpenWorkArea("ТРЕБОВ", "", "");
  try
   BeginChanging(vRO1);
     try
     vRecNum = InsertRecord(vRO1);
     CopyRecord(vRO1, vRO1, 1, vRecNum);
```

```
      except
     CancelChanging(vRO1);
       Raise;
      end;
    EndChanging(vRO1);
  finally
   CloseWorkArea(vRO1);
  end;
end; – OnDublRecord
```

Копирование первой записи из картотеки “ТРЕБОВ” в картотеку “ТМЦ”:

```
proc OnCopyRecord(S: String);
var vRO1: Integer;
var vRO2: Integer;
var vRecNum: Integer;
var i: Integer;
  vRO1 = OpenWorkArea(“ТРЕБОВ”, “”, “”);
    try
    vRO2 = OpenWorkArea(“ТМЦ”, “”, “”);
      try
      BeginChanging(vRO2);
        try
       vRecNum = InsertRecord(vRO2);
       CopyRecord(vRO1, vRO2, 1, vRecNum);
        except
        CancelChanging(vRO2);
          Raise;
        end;
      EndChanging(vRO2);
     finally
      CloseWorkArea(vRO2);
     end;
   finally
   CloseWorkArea(vRO1);
   end;
end; – OnCopyRecord
```

Процедура НачатьИзменения

Б

Синоним:

BeginChanging

Формат описания:

НачатьИзменения (НомерРО : Ц [; Фикс : Л]);

Входные параметры:

НомерРО — номер рабочей области

Фикс — необязательный параметр, определяющий режим работы рабочей области: при значении "ИСТИНА" изменения фиксируются, иначе —нет

Назначение:

Начинает редактирование записей рабочей области.

Примечания:

1. Если параметр *Фикс* не задан, то его значение считается равным “ИСТИНА”.

2. Записи в рабочей области пронумерованы, начиная с единицы. Работа с ними (вставка, удаление, изменение значений) зависит от установленного режима: с фиксацией рабочей области и без нее. Если рабочая область не фиксирована, то значение параметра *Фикс* равно “ЛОЖЬ”, иначе — “ИСТИНА”.

   В режиме *фиксации* все записи в рабочей области от начала изменений до их окончания изменений (вызова процедуры **ЗакончитьИзменения** представлены в том порядке, в котором они были при вызове процедуры **НачатьИзменения**. Добавленные записи вставляются в конец списка записей. Удаленные записи образовывают “дырки” в списке записей и последующее обращение к этой записи приведет к ошибке, например:

```
УдалитьЗапись (Раб0бл, 2);
i = ЗаписатьПолеС (Раб0бл, 2, F1, 2);   -- это приведет к ошибке, т.к. запись
                                        -- с номером 2 удалена
```

   Изменение значения одного из полей упорядочивания не влияет на положение записи в списке. Изменение полей записи таким образом, что запись перестает удовлетворять наложенному фильтру, не приводит к ее удалению из списка.

   Данный режим позволяет наиболее безопасно проводить изменения в рабочей области, но при большом количестве записей, удовлетворяющих установленному фильтру, работает медленно.

   В *расфиксированной* рабочей области, в которой не установлен ни фильтр, ни поля упорядочивания, вставка и редактирование записей производится так же, как и в фиксированной. Удаление записи приводит к сдвигу вверх (уменьшению номера) последующих записей.

   В этом режиме фиксация списка записей не производится. Это означает, что при изменении значения одного из полей упорядочивания запись может изменить свое положение в списке (только в том случае, если указаны поля упорядочивания). Если изменение записи приводит к тому, что она перестает удовлетворять фильтру, то она удаляется из списка (только в том случае, если указан фильтр).

   Данные изменения (изменение положения, удаление из массива) происходят не сразу после модификации поля или вставки

записи, а при первом обращении к другой записи данной рабочей области.

Пример:

```
Попытка
   НачатьИзменения (РабОбласть);
   Попытка
      Номер3 = ВставитьЗапись (РабОбласть);
      Номер3 = ЗаписатьПолеС (РабОбласть, Номер3, “Название”, “ТОО
Лютики”);
      ЗакончитьИзменения (РабОбласть);
   Исключение
      ОтменитьИзменения (РабОбласть);
      Возбудить;
   Конец;
Окончание
```

Процедура ОткрытьБланкРедактор

Б

Синоним:

OpenBlankEditor

Формат описания:

ОткрытьБланкРедактор (ИмяБл : С; НомерРО : Ц; Номер3 : Ц);

Входные параметры:

ИмяБл — имя бланка

НомерРО — номер рабочей области

Номер3 — номер записи картотеки

Назначение:

Открывает бланк-редактор в модальном окне для редактирования заданной записи картотеки.

Пример:

```
ОткрытьБланкРедактор ("Поруч", РабОбл, НомерЗап);
```

Функция ОткрытьМассив

Б

Синоним:

OpenArray

Формат описания:

ОткрытьМассив (НомерРО : Ц; НомерЗ : Ц; ИмяСтрПоля : С; ИмяПоляУп : С [; Фильтр : С]) : Ц;

Входные параметры:

НомерРО — номер исходной рабочей области

НомерЗ — номер записи

ИмяСтрПоля — имя структурного поля

ИмяПоляУп — имя поля упорядочивания

Фильтр — фильтр записей (необязательный параметр)

Назначение:

Открывает новую рабочую область для работы с заданным структурным полем. Возвращает номер новой рабочей области.

Примечание:

Рабочая область, открытая с помощью данной функции, должна быть закрыта процедурой **ЗакрытьРабочуюОбласть.**

Пример:

```
vWA = OpenWorkArea("ABAHCOT",'');
 try
 vSA = OpenArray(vWA, 1, "ДОКУМЕНТЫ", "ДС01", "ДС01 = «20»');
  try
  Trace(Str(Records(vSA)));
 finally
 CloseWorkArea(vSA);
 end;
finally
CloseWorkArea(vWA);
end;
```

Функция ОткрытьРабочуюОбласть

Б

Синоним:

OpenWorkArea

Формат описания:

ОткрытьРабочуюОбласть (ИмяКарт : С; ИмяПоляУп : С [; Фильтр : С]) : Ц;

Входные параметры:

ИмяКарт — имя картотеки

ИмяПоляУп — имя поля упорядочивания

Фильтр — фильтр записей

Назначение:

Открывает рабочую область над картотекой с заданным именемы, вносит туда все записи, удовлетворяющие фильтру, и возвращает номер этой рабочей области. Если фильтр не задан, то считается, что ему удовлетворяют все записи. Если в качестве поля упорядочивания задана пустая строка, то записи располагаются в естественном порядке.

Примечание:

В качестве поля упорядочивания может использоваться составной ключ.

Пример:

```
-- открываем рабочую область над картотекой 'NNN'
РабОбласть = ОткрытьРабочуюОбласть ('NNN', '');
```

Процедура ОтменитьИзменения

Б

Синоним:

CancelChanging

Формат описания:

ОтменитьИзменения (НомерРО : Ц);

Входные параметры:

НомерРО — номер рабочей области

Назначение:

Отменяет все изменения, сделанные в рабочей области после выполнения функции **НачатьИзменения**.

Пример:

```
Попытка
   НачатьИзменения (РабОбласть);
   Попытка
      Номер3 = ВставитьЗапись (РабОбласть);
      Номер3 = ЗаписатьПолеЦ (РабОбласть, Номер3, "Название", "ТОО Лютики");
      ЗакончитьИзменения (РабОбласть);
   Исключение
      ОтменитьИзменения (РабОбласть);
      Возбудить;
   Конец;
Окончание
```

Функция Поиск

Б

Синоним:

Search

Формат описания:

Поиск (НомерРО : Ц; НачЗап : Ц; Эталон) : Ц;

Входные параметры:

НомерРО — номер рабочей области

НачЗап — номер записи, от которой начинается поиск

Эталон — искомое выражение заданного типа

Назначение:

Осуществляет поиск записи по заданному выражению в рабочей области с заданным номером, начиная с записи, номер которой указан в параметре *НачЗап*. Поиск производится по значениям поля упорядочивания (в приведенном ниже примере — по полю *Название*). Возвращает номер найденной записи или ноль в случае ее отсутствия.

Примечание:

Если рабочая область упорядочена по одному полю, то и в функции **Поиск** в качестве параметра *Эталон* должно использоваться одно из значений этого поля. Если же рабочая область упорядочена по многореквизитному ключу, то и в качестве эталона должны быть указаны через запятую значения всех полей, соответствующих заданному ключу. Например, если рабочая область открыта следующим образом:

```
РО = ОткрытьРабочуюОбласть ('NNN', 'Название; Дата');
```

то при вызове функции

```
Номер3 = Поиск (РО, 1, "ТОО Лютики");
```

возникнет ошибка. В данном случае вызов функции **Поиск** должен осуществляться следующим образом:

```
Номер3 = Поиск (РО, 1, "ТОО Лютики", 17.04.98);
```

Пример:

```
Номер3 = Поиск (РабОбласть, 1, "ТОО Лютики");
-- поиск записи, у которой поле Название = "ТОО Лютики"
```

Функции Поиск (ПоискЧ, ПоискЦ, ПоискС, ПоискЛ, ПоискД)

Б

Синонимы:

SearchN, SearchI, SearchS, SearchL, SearchD

Формат описания:

Поиск* (НомерРО : Ц; Эталон : *; НачПоз : Ц; НомерЗап : Ц) : Л;

Входные параметры:

НомерРО — номер рабочей области

Эталон — искомое выражение, тип которого должен совпадать с типом суффикса функции (*)

НачПоз — номер записи, от которой начинается поиск

НомерЗап — номер записи с найденным эталоном

Назначение:

Осуществляет поиск записи по заданному выражению в рабочей области с заданным номером, начиная с записи, номер которой указан в параметре *НачПоз*. Функция возвращает переменную логического типа. При удачном поиске (запись найдена) функция принимает значение "ИСТИНА", а переменная *НомерЗап* —значение, равное номеру найденной записи. При неудачном поиске функция принимает значение "ЛОЖЬ", а *НомерЗап*=0.

Примечания:

1. Поиск производится по значениям поля упорядочивания (в приведенном ниже примере — по полю *Название*). Если в рабочей области значение поля упорядочивания не задано (записи расположены в хронологическом порядке), то генерируется сообщение об ошибке.
2. Для номера записи может использоваться только локальная переменная процедуры или функции.

Пример:

```
НайденаЗапись = ПоискЦ (РабОбласть, "ТОО Лютики", 1, НомОбнаружЗап);
-- поиск записи, у которой поле Название = "ТОО Лютики" Конец;
```

Процедура УдалитьВсеЗаписи

Б

Синоним:

DeleteAllRecords

Формат описания:

УдалитьВсеЗаписи (НомерРО : Ц);

Входные параметры:

НомерРО — номер рабочей области

Назначение:

Удаляет все записи с учетом фильтра в открытой рабочей области.

Пример:

```
УдалитьВсеЗаписи (РабОбласть, 1); -- удаляем все записи
```

Процедура УдалитьЗапись

Б

Синоним:

DeleteRecord

Формат описания:

УдалитьЗапись (НомерРО : Ц; НомерЗ : Ц);

NET **УдалитьЗапись** (НомерРО : Ц; НомерЗ : Ц [; Проверка : Л]);

Входные параметры:

НомерРО — номер рабочей области

НомерЗ — номер записи

NET *Проверка* — логический параметр для проверки ссылочной целостности. Если его значение равно “ИСТИНА”, то запись удаляется с проверкой, иначе — без проверки. По умолчанию, параметр принимает значение “ЛОЖЬ"

Назначение:

Удаляет запись из рабочей области с заданным номером.

Пример:

```
УдалитьЗапись (РабОбласть, 1); -- удаляем первую запись
```

Процедура Упорядочить

Б

Синоним:

Order

Формат описания:

Упорядочить (НомерРо : Ц; Ключ : С);

Входные параметры:

НомерРО — номер рабочей области

Ключ — строка, содержащая перечень имен полей через точку с запятой, по которым происходит упорядочивание записей в рабочей области

Назначение:

Упорядочивает записи в рабочей области с заданным номером по полю упорядочивания.

Пример:

```
Упорядочить (РабОбласть, "Поставщик"); -- упорядочиваем по полю Поставщик
```

Процедура УстФильтр

Б

Синоним:

SetFilter

Формат описания:

УстФильтр (НомерРо : Ц; Фильтр : С);

Входные параметры:

НомерРО — номер рабочей области

Фильтр — фильтр

Назначение:

Устанавливает фильтр для выбора записей в рабочей области.

Пример:

```
УстФильтр (РабОбласть, "Название = "ТОО Лютики"");
```

Процедуры и функции для работы с DBF-файлами

Процедура ДобавитьБланкBDbf

Б

Синоним:

DbfAddBlank

Формат описания:

ДобавитьБланкBDbf (НомерDbf : Ц; ИмяБланка : С);

Входные параметры:

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

ИмяБланка— имя бланка, информация из которого записывается, либо пустая строка, если нужно записать информацию из бланка, вызвавшего процедуру

Назначение:

Записывает содержимое бланка в новую запись DBF-файла в соответствии с содержимым файла структуры БД, созданного функцией **ОткрытьDbf**.

Примеры:

```
ДобавитьБланкBDbf (2,"Инфо.Физлицо");
ДобавитьБланкBDbf (НомФайла,"");
```

Процедура ЗакрытьDbf

Б

Синоним:

DbfClose

Формат описания:

ЗакрытьDbf (НомерDbf : Ц);

Входные параметры:

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

Назначение:

Закрывает DBF-файл, открытый функцией **ОткрытьDbf**.

Пример:

```
ЗакрытьDbf (НомФайла);
```

Функция ЗаписатьПолеBDbf

Б

Синоним:

DbfWriteField

Формат описания:

ЗаписатьПолеBDbf (НомерDbf : Ц; ИмяПоля : С; ИмяБланка : С; ИмяПерем : С) : Л;

Входные параметры

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

ИмяПоля — имя поля *текущей* записи DBF-файла

ИмяБланка — имя бланка, из которого производится запись, или пустая строка, если запись нужно выполнить из бланка, вызвавшего функцию

ИмяПерем — имя переменной бланка *ИмяБланка*

Назначение:

Записывает значение из переменной бланка в поле *текущей* записи DBF-файла и возвращает значение "ИСТИНА", если запись прошла успешно.

Пример:

```
ПРОЦ Запись (Объект :Строка);
    Если ЗаписатьПолеBDbf (НомФайла,"Поле_А","","Перем_1"):
        Трассировка ("Поле записано, ОК");
    Иначе
        Трассировка ("Поле не записывается");
   Конец;
Конец;
```

Функция КонецФайлаDbf

Б

Синоним:

DbfEndOfFile

Формат описания:

КонецФайлаDbf (НомерDbf : Ц) : Л

Входные параметры

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

Назначение:

Возвращает значение “ИСТИНА”, если достигнут конец файла.

Пример:

```
DBF = ОткрытьDbf ("Тест","Struct.ini",ЛОЖЬ,ЛОЖЬ);
...
ПерваяЗаписьDbf (DBF);
REZ = КонецФайлаDbf (DBF);
While NOT КонецФайлаDbf (DBF) do
   NumRec=NumRec+1;
   REZ = DbfReadField (DBF,"Code","","Code");
   HINT ("Проверяется запись номер "+СТР(NumRec));
   ...
   СледующаяЗаписьDbf (DBF);
Конец;
```

Функция ОткрытьDbf

Б

Синоним:

DbfOpen

Формат описания:

ОткрытьDbf (ИмяDbf : С; ИмяСтруктуры : С; Созд : Л; DOS : Л
[; ТолькоЧтение : Л [; КодоваяСтраница: Ц]])) : Ц;

Входные параметры:

ИмяDbf — имя DBF-файла, который нужно открыть или создать.

ИмяСтруктуры — имя текстового файла с описанием структуры базы данных. (Формат строки текстового файла с описанием DBF *см. ниже*)

Созд — логическое выражение (при значении “ИСТИНА”: если DBF-файл с именем *ИмяDbf* уже существует, то будет создан новый файл, а старый файл будет переименован с расширением BAK; при значении “ЛОЖЬ”: если DBF-файл с именем *ИмяDbf* уже существует, то откроется существующий файл, иначе — будет создан новый файл)

DOS — логическое выражение (“ИСТИНА” — информация в DBF-файле переводится в DOS-кодировку)

ТолькоЧтение — параметр, характеризующий режим работы файла. По умолчанию его значение равно “ЛОЖЬ” и файл открывается в монопольном режиме, т.е. он доступен на чтение и запись. При значении “ИСТИНА” файл открывается только на чтение

КодоваяСтраница — число, которое будет записано в 29 байт заголовка DBF-файла, при условии, что создается новый файл *(см. параметр Созд)*. Может принимать значения следующих констант:

DbfCpDos866 = 101 (0x0065);
DbfCpWin1251 = 201 (0x00C9).

Примечание:

Параметр *КодоваяСтраница* никак не связан с параметром *DOS*.

Формат строки текстового файла с описанием DBF

ПолеБД ":" Тип ["(" Длина "." Точность ")"] "=" ИмяПерем

где:

ПолеБД — имя поля в dbf-файле;

Тип — слово "String", "Real", "Integer", "Logical", "Date", "MEMO";

Если DBF файл содержит memo-поля, то к нему прилагается другой файл с тем же именем и расширением DBT (не путать с форматом DBT-файлов программы ТБ 6), в котором находится само содержимое memo-полей. По стандарту размер файла DBT не может превышать 32767 Кб, при нарушении этого условия запись в memo-поле произведена не будет, а пользователю будет выдано сообщение об ошибке.

В бланках допускается обработка строковых memo-полей, заканчивающихся на #0. Бланки ТБ поддерживают работу с базами данных в формате FoxBASE+/dBASEIII+. Чтение/запись из этих полей осуществляется в переменные бланка:

- строкового типа (в соответствии с ограничением бланков размер поля урезается до 255 символов);
- массивы строкового типа.

Элементы массива должны иметь тип “СТРОКА”. При попытке считать или записать элемент memo-поля в массив, тип которого не является строковым, программа выдает сообщение об ошибке.

В операциях чтения при работе с простыми переменными длина этих полей сокращается до 255 символов, т.к. формат строковых полей бланков ТБ 6 ограничен 255 символами. При использовании строковых массивов содержимое memo-поля последовательно помещается в элементы массива, начиная с первого. Каждый элемент массива имеет подстроку длиной до 255 символов.

Функции записи последовательно считывают элементы массива, начиная с первого и заканчивая первой пустой строкой, и помещают в memo-поле результат их последовательной конкатенации (объединения строк).

Длина и Точность — целые числа (точность — только для вещественных полей);

ИмяПерем — переменная бланка, связанная с данным полем.

Назначение:

Открывает DBF-файл на чтение и запись и возвращает номер открытого файла в цепочке открытых DBF-файлов (этот номер используется в процедурах **ДобавитьБланкBDbf, ПерваяЗаписьDbf, СледующаяЗаписьDbf** и функциях **ЗакрытьDbf, ЗаписатьПолеBDbf, КонецФайлаDbf, УдалитьЗаписьDbf, ЧислоЗаписейBDbf, ЧитатьБланкИзDbf, ЧитатьПолеИзDbf**).

Пример:

См. пример для функции **КонецФайлаDbf**

Процедура ПерваяЗаписьDbf

Б

Синоним:

DbfFirstRecord

Формат описания:

ПерваяЗаписьDbf (НомерDbf : Ц);

Входные параметры:

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

Назначение:

Переходит в начало DBF-файла.

Пример:

См. пример для функции **КонецФайлаDbf**

Процедура СледующаяЗаписьDbf

Б

Синоним:

DbfNextRecord

Формат описания:

СледующаяЗаписьDbf (НомерDbf : Ц);

Входные параметры:

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

Назначение:

Переходит на следующую запись в DBF-файле, если не достигнут конец файла.

Пример:

См. пример для функции **КонецФайлаDbf**

Процедура УдалитьЗаписьDbf

Б

Синоним:

DbfDeleteRecord

Формат описания:

УдалитьЗаписьDbf (НомерDbf : Ц);

Входные параметры:

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

Назначение:

Удаляет *текущую* запись в DBF-файле.

Пример:

```
УдалитьЗаписьDbf (DBF);
```

Функция ЧислоЗаписейDbf

Б

Синоним:

DbfRecordCount

Формат описания:

ЧислоЗаписейDbf (НомерDbf : Ц) : Ц;

Входные параметры:

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

Назначение:

Возвращает число записей в DBF-файле.

Пример:

```
ЧислоЗаписейDbf (DBF);
```

Функция ЧитатьБланкИзDbf

Б

Синоним:

DbfReadBlank

Формат описания:

ЧитатьБланкИзDbf (НомерDbf : Ц; ИмяБланка : С) : Л;

Входные параметры:

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

ИмяБланка — имя бланка, в который производится считывание, или пустая строка, если считывание выполняется в бланк, вызвавшего функцию

Назначение:

Считывает *текущую* запись из DBF-файла в бланк и возвращает значение "ИСТИНА", если считывание прошло успешно.

Пример:

```
ПРОЦ Чтение (Объект :Строка);
   Если ЧитатьБланкИзDbf (НомФайла,""):
      Трассировка ("Бланк прочитан, ОК");
   Иначе
      Трассировка ("Бланк не читается");
  Конец;
Конец;
```

Функция ЧитатьПолеИзDbf

Б

Синоним:

DbfReadField

Формат описания:

ЧитатьПолеИзDbf (НомерDbf : Ц; ИмяПоля : С; ИмяБланка : С; ИмяПерем : С) : Л;

Входные параметры:

НомерDbf — номер открытого DBF-файла, возвращенный функцией **ОткрытьDbf**

ИмяПоля — имя поля в DBF-файле

ИмяБланка — имя бланка, в который производится считывание, или пустая строка, если считывание нужно произвести в бланк, вызвавшего функцию

ИмяПерем — имя переменной в бланке с именем *ИмяБланка*

Назначение:

Читает значение из поля *текущей* записи DBF-файла в переменную бланка и возвращает значение "ИСТИНА", если чтение прошло успешно.

Пример:

```
ПРОЦ Чтение (Объект :Строка);
    Если ЧитатьПолеИзDbf (НомФайла, "Поле_А, "", "Перем_1"):
        Трассировка ("Поле прочитано, ОК");
    Иначе
        Трассировка ("Поле не читается");
    Конец;
Конец;
```

Функции проверки типов свойств объектов

Функция КакДата

Б

Синоним:

AsDate

Формат описания:

КакДата (Свойство: *) : Д;

где * — тип свойства объекта

Входные параметры:

Свойство — свойство объекта

Назначение:

Проверяет, соответствует ли тип заданного свойства объекта типу ДАТА.

Примечание:

Если реальный тип соответствующего свойства отличается от типа ДАТА, то возникает ошибка времени исполнения.

Пример:

```
ДатаОб = КакДата ( НекОб );
```

Функция КакЗапись

Б

Синоним:

AsRecord

Формат описания:

КакЗапись (Свойство: *) : З;

где * — тип свойства объекта

Входные параметры:

Свойство — свойство объекта

Назначение:

Проверяет, соответствует тип заданного свойства объекта типу ЗАПИСЬ.

Примечание:

Если реальный тип соответствующего свойства отличается от типа запись, то возникает ошибка времени исполнения.

Пример:

```
Заг = КакСтрока ( Кнопка.Заголовок );
-- Заг = Заголовку кнопки (то, что на ней написано)
```

Функция КакЛогическое

Б

Синоним:

AsLogical

Формат описания:

КакЛогическое (Свойство: *) : Л;

где * — тип свойства объекта

Входные параметры:

Свойство — свойство объекта

Назначение:

Проверяет, соответствует тип заданного свойства объекта логическому типу.

Примечание:

Если реальный тип соответствующего свойства отличается от логического, то возникает ошибка времени исполнения.

Пример:

```
Видима = КакЛогическое ( Кнопка.Видима );
-- Видима = Истина, если кнопка видна на экране
```

Функция КакСтрока

Б

Синоним:

AsString

Формат описания:

КакСтрока (Свойство: *) : С;

где * — тип свойства объекта

Входные параметры:

Свойство — свойство объекта

Назначение:

Проверяет, соответствует тип заданного свойства объекта типу СТРОКА.

Примечание:

Если реальный тип соответствующего свойства отличается от строкового, то возникает ошибка времени исполнения.

Пример:

```
Заг = КакСтрока ( Кнопка.Заголовок );
-- Заг = Заголовку кнопки (то, что на ней написано)
```

Функция КакЦелое

Б

Синоним:

AsInterger

Формат описания:

КакЦелое (Свойство: *) : Ц;

где * — тип свойства объекта

Входные параметры:

Свойство — свойство объекта

Назначение:

Проверяет, соответствует тип заданного свойства объекта целому типу.

Примечание:

Если реальный тип соответствующего свойства отличается от целого, то возникает ошибка времени исполнения.

Пример:

```
Высота = КакЦелое ( Кнопка.Высота ); -- Высота = Высоте кнопки
```

Функция КакЧисло

Б

Синоним:

AsReal

Формат описания:

КакЧисло (Свойство: *) : Ч;

где * — тип свойства объекта

Входные параметры:

Свойство — свойство объекта

Назначение:

Проверяет, соответствует тип заданного свойства объекта числовому типу.

Примечание:

Если реальный тип соответствующего свойства отличается от числового, то возникает ошибка времени исполнения.

Пример:

```
Число = КакЧисло ( НекотОбъект.ЧислСвойство );
-- Число = значению ЧислСвойства НекотОбъекта
```

Функции поддержки автоматических объектов (OLE Avtomation)

Функция СоздатьАвтоОбъект

Б

Синоним:

CreateAutoObject

Формат описания:

СоздатьАвтоОбъект (ИмяOLE-Объекта : С) : АвтоОбъект;

Входные параметры:

ИмяOLE-Объекта — строка с именем OLE-объекта, зарегистрированным в системе Windows

Назначение:

Функция служит для создания автоматического объекта. По имени класса OLE-объекта, зарегистрированного в системе Windows, создает его экземпляр и возвращает указатель на вновь созданный объект. Если объект невозможно создать, генерируется исключительная ситуация (ошибка).

Примечание:

Созданный объект обязательно надо освобождать процедурой ОчиститьПеременную (ClearVariable).

Пример:

```
Proc OpenWordButtonClick(Sender: String);
var vApp: AutoObject;
  vApp = CreateAutoObject("Word.Application"); – Создаем объект.
   try
   vApp.Visible = True;
   finally
    ClearVariable(vApp);
       end;
    end;
```

Функция ОткрытьАвтоОбъект

Б

Синоним:

OpenAutoObject

Формат описания:

ОткрытьАвтоОбъект (ИмяOLE-Объекта : С; Создать : Л) : АвтоОбъект;

Входные параметры:

ИмяOLE-Объекта – строка с именем OLE-объекта, зарегистрированным в системе Windows

Создать — логический параметр, если его знаачение равно "ИСТИНА", то будет создан новый экземпляр объекта (если такого еще нет), если "ЛОЖЬ" — производится проверка, существует ли такой OLE-бъект, и, если существует, то возвращает ссылку на него.

Назначение:

Функция служит для открытия автоматического объекта. По имени класса OLE-объекта, зарегистрированного в системе Windows, проверяет, существует ли экземляр указанного класса. Если объект уже существует, функция возвращает ссылку на него, и далее с объектом можно работать точно также, как и с объектом, созданным с помощью функции СоздатьАвтоОбъект (CreateAutoObject).

В противном случае, если объектов указанного класса пока нет, то функция может создать новый объект при условии, что второй параметр равен "ИСТИНА".

Примечание:

Открытый обеъкт обязательно надо освобождать процедурой ОчиститьПеременную (ClearVariable).

Пример:

```
Proc OpenWordButtonClick(Sender: String);
var vApp: AutoObject;
 vApp = OpenAutoObject("Word.Application", True);
 --Открываем объект. Если его нет, то создаем.
  try
   vApp.Visible = True;
  finally
   ClearVariable(vApp);
   end;
end;
```

Функция КакАвтоОбъект

Б

Синоним:

AsAutoObject

Формат описания:

КакАвтоОбъект (Свойство) : АвтоОбъект;

Входные параметры:

Свойство — свойство автообъекта, которое возвращает ссылку на другой объект

Назначение:

Функция служит для преобразования ссылки OLE-объекта к типу автоматический объект (АвтоОбъект).

Примечание:

Открытый таким образом автообъект обязательно надо освобождать процедурой ОчиститьПеременную (ClearVariable).

Пример 1:

```
Proc OpenWordButtonClick(Sender: String);
var vApp, vDoc: AutoObject;
 vApp = OpenAutoObject("Word.Application", True);
 -- Открываем объект. Если его нет, то создаем.
  try
  vDoc = AsAutoObject(vApp.Documents);
  -- Получаем ссылку на OLE-объект Word.Application.Documents.
    try
   vDoc.Open(D:\Temp\Функции.rtf, ,True);
   -- Word.Application.Documents.Open
   finally
   ClearVariable(vDoc);
    end;
   vApp.Visible = True;
   finally
    ClearVariable(vApp);
    end;
```

Примечание к примеру 1:

Обратите внимание, что для пропуска второго необязательного параметра в методе Open указаны две запятые подряд.

Пример 2:

Для получения значения из свойств OLE-объектов можно воспользоваться функциями проверки типов свойств объектов:

```
Proc OpenWordButtonClick(Sender: String);
var vApp: AutoObject;
 vApp = OpenAutoObject("Word.Application", True);
 -- Открываем объект. Если его нет, то создаем.
 try
  if AsLogical(vApp.CheckSpelling("Привет!")):
     Message("Ошибок нет");
   end;
 finally
  ClearVariable(vApp);
  end;
end;
```

Пример 3

Для того, что бы воспользоваться OLE-объектом, ссылка на который хранится в переменной другого бланка, нужно воспользоваться переприсваиванием в локальную переменную, для обращения по краткой нотификации.

```
БланкСШаблоном "test";
Proc OpenWordButtonClick(Sender: String);
var vApp, OtherBlankApp: AutoObject;
 OtherBlankApp = OpenAutoObject("Word.Application", true);
 -- Открываем объект. Если его нет, то создаем.
  try
 vApp = OtherBlankApp;
  if AsLogical(vApp.CheckSpelling("Привет!")):
  -- Т.к. длиная нотификация (Other.Blank.App.CheckSpelling) недопустима.
   Message("Ошибок нет");
  end;
 vApp.Quit;
 -- Закрываем MS Word.
 finally
  ClearVariable(vApp);
 end;
end;
```

Пример 4

Чтение значения из индексных свойств автоматических объектов.

```
proc SetExcelCellValue;
 var xlsApp, xlsBooks, xlsBook, xlsSheet, xlsRange: AutoObject;
 xlsApp=OpenAutoObject("Excel.Application", True);
  try
   xlsBooks  = AsAutoObject(xlsApp.Workbooks);
     try
      xlsBook = AsAutoObject(xlsBooks.Add);
       try
      xlsSheet = AsAutoObject(xlsBook.Worksheets[1]);
        try
       xlsRange = AsAutoObject(xlsSheet.Cells[1,1]);
       -- Обращение к ячейке с координатами (1, 1)
          try
        xlsRange.Value = 10;
       -- нужно записать в ячейку конкретное значение
        finally
        ClearVariable(xlsRange);
          end;
       xlsApp.Visible = True;
       finally
       ClearVariable(xlsSheet);
        end;
      finally
       ClearVariable(xlsBook);
       end;
    finally
```

```
      ClearVariable(xlsBooks);
     end;
   finally
    ClearVariable(xlsApp);
   end;
 end;
```

Функции для обеспечения совместимости с предыдущими версиями Турбо Бухгалтера

Функция Маска

Б

Синоним:

нет

Формат описания:

Маска (УслНаСчета : С; УслНаПризнаки : С) : С;

Входные параметры:

УслНаСчета — условие на счета

УслНаПризнаки — условие на признаки

Назначение:

Формирует условие отбора проводок, т.е. формирует строку по следующему правилу: *УслНаСчета* + "{" + *УслНаПризнаки* + "}".

Пример:

```
мМаск : Строка = Маска ("1111","22222");
```

Функция Масч

Б

Синоним:

нет

Формат описания:

Масч (Счет : С) : С;

Входные параметры:

Счет — имя счета

Назначение:

Возвращает корректную маску имени счета. Если имя счета имеет длину один символ, то добавляет спереди "0", в противном случае — оставляет маску без изменения (т.е. "1" —> "01", "20" —> "20").

Пример:

```
мСч : Строка = Масч (Сч);
```

Функция Месяц

Б

Синоним:

нет

Формат описания:

Месяц (Номер : Ч) : С;

Входные параметры:

Номер — номер месяца

Назначение:

Выдает русское название месяца по его номеру (“1” —> “январь”, “2” —> “февраль” и т.д.). Если номер имеет дробную часть, то производится округление; если номер превышает 12, то месяц определяется по остатку от деления на 12. Если номер равен нулю, то выдается пустая строка.

Пример:

```
мНом : Строка = Месяц (Ном);
```

Функция Месяца

Б

Синоним:

нет

Формат описания:

Месяца (Номер : Ч) : С;

Входные параметры:

Номер — номер месяца

Назначение:

Выдает русское название месяца по его номеру в родительном падеже (“1” —> “января”, “2” —> “февраля” и т.д.). Если номер имеет дробную часть, то производится округление; если номер превышает 12, то месяц определяется по остатку от деления на 12. Если номер равен нулю, то выдается пустая строка.

Пример:

```
мНом1 : Строка = Месяца (Ном);
```

Функция НДС_Втч

Б

Синоним:

нет

Формат описания:

НДС_Втч (Сумма : Ч; СтавкаНДС : Ч) : Ч;

Входные параметры:

Сумма — некоторая сумма

СтавкаНДС — ставка НДС (в десятичных долях)

Назначение:

Возвращает сумму НДС, выделенную из переданной суммы по указанной ставке.

Пример:

```
нСум : Число = НДС_Втч (Сум,НДС);
```

Функция Прих

Б

Синоним:

нет

Формат описания:

Прих (УслОтбора : С; НачалоМес : Д) : Ч;

Входные параметры:

УслОтбора — условие отбора

НачалоМес — дата начала месяца

Назначение:

Вычисляет сумму остатка на начало месяца и дебетового оборота с начала месяца по заданному условию отбора. Проводки за текущую дату не учитываются.

Пример:

```
дПрих : Число = Прих ("01",Дат);
```

Функция Lang

Б

Синоним:

нет

Формат описания:

Lang (Строка : C) : C;

Входные параметры:

Строка — любая строка

Назначение:

Преобразует строку *Строка* в строку, которая может быть именем файла MS DOS, по следующим правилам:

— вся строка переводится в верхний регистр;

— русская буква преобразуется в ее латинский аналог;

— символы “ “, “-”, “.”, “!”, “,”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “+”, “=”, “/”, “?”, “\”, “~”, “`”, “'”, “:”, “"”, “<”, “>”, “|” заменяются на подчерк;

— остальные символы остаются без изменений.

Пример:

```
LCтр1 : Строка = Lang (Стр1);
```

Функции приведения типов

Функция ДатуВСтроку

Б

Синоним:

DateToString

Формат описания:

ДатуВСтроку (Дат : Д) : С;

Входные параметры:

Дат — исходная дата в формате ДД.ММ.ГГГГ или ДД.ММ.ГГ

Назначение:

По заданной дате возвращает ее строковое представление в формате ДД.ММ.ГГГГ.

Примечания:

1. Если в исходной дате год задан двумя цифрами, то функция возвращает дату в соответствии с установками диалога "Формат даты".
2. Если нужен иной формат преобразования даты, следует пользоваться функциями из исходного описания бланка Раб.Формат.

Пример:

```
Проц Р1;
    Перем РабОбласть : Целое;
    Перем ДатаПлПоруч : Дата;
        -- открываем рабочую область для работы с картотекой "Поруч"
        РабОбласть = ОткрытьРабочуюОбласть ("Поруч", "Сумма");
        Попытка
            -- считываем дату заполнения платежного поручения
            ДатаПлПоруч = ВзятьПолеД (РабОбласть, 2, "Дата");
            Сообщение("Дата платежного поручения =
"+ДатаВСтроку(ДатаПлПоруч));
        Окончание
            -- Закрываем рабочую область
            ЗакрытьРабочуюОбласть(РабОбласть);
        Конец;
Конец;
```

Функция ЗаписьВСтроку

Б

Синоним:

RecordToString

Формат описания:

ЗаписьВСтроку (Зап : З) : С;

Входные параметры:

Зап — ссылка на запись в картотеке

Назначение:

Возвращает строковое представление заданной ссылки на картотечную запись.

Примечание:

Данная функция может быть использована при формировании фильтра для рабочей области.

Пример:

```
Проц Р1;
   Перем ЗапСтр : Строка;
   Перем РабОбласть : Целое;
   Перем Зап : Запись;
   -- Открытие рабочей области для работы с картотекой "ЮрЛицо"
   РабОбласть = ОткрытьРабочуюОбласть ("ЮрЛицо", " ");
   Попытка
      -- Ссылка на текущую запись
      Зап = ВзятьКлючЗаписи (РабОбласть, 1));
      -- получаем ссылку на первую запись
      ЗапСтр = ЗаписьВСтроку (Зап);
   Окончание
      -- Закрытие рабочей области
      ЗакрытьРабочуюОбласть(РабОбласть);
   Конец;
Конец;
```

Функция ЗаписьКЧислу

Б

Синоним:

RecordToInteger

Формат описания:

ЗаписьВЧисло (Зап : З) : Ц;

Входные параметры:

Зап — ссылка на запись в картотеке

Назначение:

Возвращает числовое представление заданной ссылки на картотечную запись.

Пример:

```
Тов = GetFieldR(war, j, Деталь);
    UkeyДетали = RecordToInteger(Тов);
```

Функция СтрокуВДату

Б

Синоним:

StringToDate

Формат описания:

СтрокуВДату (ИсхСтр : С) : Д;

Входные параметры:

ИсхСтр — исходная строка, записанная в формате даты

Назначение:

По текстовому представлению даты возвращает значение типа Дата.

Примечания:

1. Если в исходной строке год задан двумя цифрами, то функция возвращает значение в соответствии с установками диалога “Формат даты”.
2. Если строка не является правильным текстовым представлением даты, возбуждается ошибка. Исходная строка может быть записана в следующих форматах: ДД.ММ.ГГ, ДД.ММ.ГГГГ, ДД/ММ/ГГ или ДД/ММ/ГГГГ.

Примеры:

```
Дата = СтрокуВДату (“25/12/99”);     -- 25.12.1999
Дата = СтрокуВДату (“05/10/39”);     -- 05.10.2039
Дата = СтрокуВДату (“14.02.1999”);   -- 14.02.1999
```

Функция СтрокуВЧисло

Б

Синоним:

StringToNumeric

Формат описания:

СтрокуВЧисло (ИсхСтр : С) : Ч;

Входные параметры:

ИсхСтр — исходная строка, записанная в формате числа

Назначение:

По текстовому представлению числа возвращает значение типа Число.

Примеры:

```
Сумма = СтрокуВЧисло ("00250000");  -- 250 000
Сумма = СтрокуВЧисло ("01.400000"); -- 1.4
```

Функция СтрокуВЗапись

Б

Синоним:

StringToRecord

Формат описания:

СтрокуВЗапись (Стр : Строка) : З;

Входные параметры:

Стр — исходная строка, которая преобразуется в ссылку на запись в картотеке

Назначение:

Возвращает ссылку на картотечную запись, сформированную на основании исходной строки.

Пример:

```
-- преобразование целого числа в запись
Зап = СтрокуВЗапись ( "{" + 6(i) +  " }" );
```

Функция ЧислоВЗапись

Б

Синоним:

IntegerToRecord

Формат описания:

ЧислоВЗапись (Ном : Ц) : З;

Входные параметры:

Ном — исходное число, которое преобразуется в ссылку на запись в картотеке

Назначение:

Возвращает ссылку на картотечную запись, сформированную на основании исходного числа.

Пример:

```
proc Test (кн:строка);
var wa:integer;
var rec:record;
  wa=OpenWorkArea(УчетКарточкаОС,);
  try
     if records(wa)>0:
    rec=IntegerToRecord(records(wa));
    --rec={2}
     fi;
  finally
    CloseWorkArea(wa);
  end;
 end;
end
```

Сервисные процедуры и функции

Функция Версия

Ж Б ф К

Синоним:

Version

Формат описания:

Версия : С;

Назначение:

Возвращает строку с номером версии программы.

Примечание:

Для версий Турбо Бухгалтер 6.9 ***Проф***, ***Клиентское место***, ***Разработчик*** данная функция возвращает ещё и платформу локальная или сетевая.

Примеры:

```
Вер = Версия;
-- Вер = "6.5"

Trace(Version);
  – «6.9 Проф (Локальная)»
```

Функция ВзятьВерсиюБД

Б

Синоним:

GetCardDBVersion

Формат описания:

ВзятьВерсиюБД (ИмяКартотеки : С) : С;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Возвращает строку с версией БД, в которую входит текущая картотека, т.е. Paradox возвращает строку "5", для InterBase — "4".

Пример:

```
ВерсияБД = ВзятьВерсиюБД ("ПрихОрд");
```

Функция ВзятьТипБД

Б

Синоним:

GetCardDBType

Формат описания:

ВзятьТипБД (ИмяКартотеки : С) : С;

Входные параметры:

ИмяКартотеки — имя картотеки

Назначение:

Возвращает строку с типом СУБД сервера, на котором хранится указанная картотека, т.е. для Paradox возвращает строку “Paradox”, для InterBase — “InterBase”.

Пример:

```
ТипБД = ВзятьТипБД ("ПрихОрд");
```

Процедура ВыполнитьПрограмму

Б

Синоним:

ExecuteProgram

Формат описания:

ВыполнитьПрограмму (КомандняяСтрока : С [, РежимОжидания : Л]);

Входные параметры:

КомандняяСтрока — строка, содержащая имя вызываемого приложения/документа, его ключи и параметры, указывающие способ или режим его выполнения

РежимОжидания — логический параметр, определяющий, ожидать ли (задано значение “ИСТИНА”) завершения выполнения запущенной программы или продолжить выполнение вызывающей ее процедуры (задано значение “ЛОЖЬ”). Если параметр отсутствует, он считается равным значению “ЛОЖЬ"

Назначение:

Запускает приложение с командной строки. Если в качестве строки указан документ, не являющийся программой, то он открывается с помощью приложения, связанного с ним в Windows.

Примечание:

Командная строка должна быть заключена в апострофы и затем в кавычки, иначе при компиляции бланка будет возникать ошибка. Также ошибка будет появляться, если в названии программы или каталога есть пробелы.

Примеры:

```
ВыполнитьПрограмму (""Notepad Readme.txt""); -- чтение файла Resdme.txt
                                              -- с помощью программы Notepad
ВыполнитьПрограмму (""c:\TBW6OP\WRK\Readme.txt"" ); -- чтение файла
   -- Readme.txt с помощью приложения Windows, обрабатывающего файлы *.txt
ВыполнитьПрограмму ( ""Calc"", ИСТИНА ); -- вызов калькулятора
```

Функция ВыполнитьПрограмму

Б

Синоним:

ExecuteProgram

Формат описания:

ВыполнитьПрограмму (КомандняяСтрока : С) : Ц;

Входные параметры:

КомандняяСтрока — строка, содержащая имя вызываемой программы/документа, ее ключи и параметры, указывающие способ или режим выполнения программы

Назначение:

Запускает приложение с командной строки и возвращает ноль при нормальном завершении выполнения процедуры, иначе — константу ошибки. Если в командной строке указан документ, не являющийся программой, то он открывается с помощью приложения, связанного с ним в Windows.

Примечания:

1. Константы ошибки можно найти в помощи по Windows API в разделе “Error Codes”.
2. Командная строка должна быть заключена в апострофы и затем в кавычки, иначе при компиляции бланка будет возникать ошибка. Также ошибка будет появляться, если в названии программы или каталога есть пробелы.

Пример:

```
Рез = ВыполнитьПрограмму (""Calc""); -- вызван калькулятор, Рез=0
Рез = ВыполнитьПрограмму (""Calc""); -- Рез=2 (файл не найден)
```

Процедура ЗакрытьПрог

Б

Синоним:

CloseApp

Формат описания:

ЗакрытьПрог;

Назначение:

Осуществляет выход из программы.

Пример:

```
ПРОЦ ПоОткрытию (Отправитель : Строка);
   ПЕРЕМ Код, Рез : Целое;
   Рез = Ввод (Код, "Введите пароль") ;
   Если ( Рез <> кмдВерно) :
      ЗакрытьПрог;
    Истина :
      Если (Код <> КодБланка) :
         ЗакрытьПрог;
      Конец;
   Конец;
Конец;
```

Функция ЗаписатьПеремВДБТ

Б

Синоним:

WriteVarInDBT

Формат описания:

ЗаписатьПеремВДБТ (ИмяФайла : С; ВхПерем : Л [; ИмяБланка : С]) : Л;

Входные параметры:

ИмяФайла — полное имя файла (с указанием пути), в который производится запись

ВхПерем — логический параметр: при значении "ИСТИНА" в файл будут записаны значения только входных полей, в противном случае — все содержимое бланка

ИмяБланка — имя бланка, значения переменных которого считываются в файл

Назначение:

Записывает переменные бланка в файл формата DBT и возвращает "ЛОЖЬ" при успешной записи, "ИСТИНА" — при ошибке (например, файл "захвачен" другим приложением).

Примечания:

1. Если имя бланка не задано, то выгружается текущий бланк.
2. Если файл для записи уже существует, то выполняется его перезапись.
3. Значения переменных записываются в DBT-файл в следующем формате: *Переменная = Значение*.

Пример:

```
Проц Р1;
Если (ЗаписатьПеремВДБТ("С:\Прочти меня.dbt", Истина, "Документы.Касса.ПрихОрд")):
    Трассировка ("Запись прошла успешно");
 Конец;
Конец;
```

Функции ЗаписатьПеремВКартотеку СчитатьПеремИзКартотеки

Б

Синоним:

WriteVarToCardFile

ReadVarFromCardFile

Формат описания:

ЗаписатьПеремВКартотеку (ИмяКартотеки : С ; ЗаписатьВходПоля : Л; ИмяБланкаВх : С; ИмяБланкаИсх : С [; ДатаНачала : Д; ДатаКонца : Д [; ВерсияБланка : Ц; ВерсияПерем : Ц [; ИмяПользователя : С; ИмяСхемы : С]]]);

СчитатьПеремИзКартотеки (ИмяКартотеки : С ; ЗаписатьВходПоля : Л; ИмяБланкаВх : С; ИмяБланкаИсх : С [; ДатаНачала : Д; ДатаКонца : Д [; ВерсияБланка : Ц; ВерсияПерем : Ц [; ИмяПользователя : С; ИмяСхемы : С]]]);

Входные параметры:

ИмяКартотеки — строка с именем картотеки, в/из котор(ую/ой) производится запись/чтение;

ЗаписатьВходПоля — логический параметр, при значении "ИСТИНА" будут записаны/считаны значения только входных полей, в противном случае — все содержимое бланка;

ИмяБланкаВх — строка с именем бланка, значение переменных которого записываются/считываются в/из картотек(у/и). В случае задания пустой строки в качестве имени бланка выгружается/считывается текущий бланк.

При записи заполняется, а при считывании учитывается значение соответсвующего поля в картотеке Blank.

ИмяБланкаИсх — строка с именем бланка под именем которого записываются/считываются переменные из бланка, указанные в параметре *ИмяБланкаВх*. В случае задания пустой строки в качестве имени бланка переменные выгружается/считывается под именем бланка указанного в параметре *ИмяБланкаВх*.

ДатаНачала, *ДатаКонца* — необязательные параметры типа даты для задания периода. При записи заполняют, а при считывании учитывают значения соответсвующих полей в картотеке BeginDate, EndDate (если таковые есть). Если параметры незаданы, то они принимают значение 1.1.100 (нулевая дата) по умолчанию. Если параметры заданы, а соответствующих полей в картотеке нет, то возбуждается ошибка.

ВерсияБланка — необязательный параметр целого типа для задания версии бланка. При записи заполняется, а при считывании учитывается значение соответсвующего поля в картотеке BlankVersion (если таковое есть). Если параметр не задан, то он принимают значение 0 по умолчанию. Если параметр задан, а соответствующего поля в картотеке нет, то возбуждается ошибка.

ВерсияПерем — необязательный параметр целого типа для задания версии переменной. При записи заполняется, а при считывании учитывается значение соответсвующего поля в картотеке VarVersion (если таковое есть). Если параметр незадан, то он принимают значение 0 по умолчанию. Если параметр задан, а соответствующего поля в картотеке нет, то возбуждается ошибка.

ИмяПользователя — необязательный параметр строкового типа для задания имени пользователя, под кот. записываются/считываются переменные бланка. В локальной версии игнорируются. В сетевой, при записи заполняется, а при считывании учитывается значение соответсвующего поля в картотеке UserName (если таковое есть).

Если параметр незадан, то он принимают значение пустой строки по умолчанию. Если параметр задан, а соответствующего поля в картотеке нет, то возбуждается ошибка.

Для определения имени пользователя можно воспользоваться функцией **UserName**.

ИмяСхемы — необязательный параметр строкового типа для задания схемы доступа, под которой записываются/считываются переменные бланка. В локальной версии игнорируются. В сетевой, при

записи заполняется, а при считывании учитывается значение соответсвующего поля в картотеке SchemaName (если таковое есть).

Если параметр не задан, то он принимают значение пустой строки по умолчанию. Если параметр задан, а соответствующего поля в картотеке нет, то возбуждается ошибка.

Для определения схемы доступа можно воспользоваться функцией **SchemaName**.

Описание:

Картотека в/из кот. происходит запись/считывание переменных бланков должна иметь следующие поля, кот. можно продемонстрировать на примере:

```
TABLE DBT = «Для хранения переменных бланка»;
  BLANK       :STRING; -- Имя бланка
  VARIABLE    :STRING; -- Имя переменной
  VALUE       :STRING; -- Значение переменной
  BEGINDATE   :DATE;  -- Дата начала периода (необязательное)
  ENDDATE     :DATE;  -- Дата конца периода (необязательное)
  BLANKVERSION :INTEGER; -- Версия бланка (необязательное)
  VARVERSION  :INTEGER; -- Версия переменной (необязательное)
  UserName  :String; -- Имя пользователя (необязат., только для сетевой)
  SchemaName :String; -- Схема доступа (необязат., только для сетевой)
  GROUPDOC :DBT(BLANK); -- Признак иерархической картотеки по полю бланка
                           (необязат.)
ENDTAB;
```

Если картотека является иерархической и информационным полем является поле с именем бланка, то запись в картотеку будет производится с учетом иерархии. Т.е. иерархия будет строиться по имени бланка, как в дереве бланков в диалоге выбора бланка.

Назначение:

Записывает/считывает переменные бланка в/из картотеку(и).

Пример 1:

Сохранение переменных собственного бланка:

```
WriteVarToCardFile(«DBT», False, «», «»);
```

Пример 2:

Сохранение переменных бланка текущего пользователя:

```
var vAlias: String;
 vAlias := AliasName(«DBT»);
WriteVarToCardFile(«DBT», False, «», «», 1.1.100, 1.1.100, 0, 0,
UserName(vAlias), SchemaName(vAlias));
```

Пример 3:

Сохранение переменных текущего бланка как переменные другого бланка:

```
WriteVarToCardFile(«DBT», False, «», «НАЛОГУЧЕТ.ДЕКЛАРАЦИЯ»);
```

Пример 4:

Считывание переменных, сохранёных из другого бланка, в текущий бланк:

```
ReadVarFromCardFile(«DBT», False, «», «НАЛОГУЧЕТ.ДЕКЛАРАЦИЯ»);
```

Процедура Звук

Б

Синоним:

Beep

Формат описания:

Звук [(Тип : Ц)];

Входные параметры:

Тип — целочисленное число, задающее тип звука

Назначение:

Выдает звуковой сигнал средствами Windows. Тип звука задается в настройках Windows и может использоваться только при наличии в компьютере звуковой карты. Если параметр *Тип* опущен, то он принимается равным нулю.

Примечание:

Допустимые значения типов звука Windows 95:

0 — звук на встроенном динамике компьютера;

16 — hand;

32 — запрос;

48 — exclamation;

64 — asterisk.

Пример:

```
Звук (16);
```

Функция ИмяБазы

Б

Синоним:

AliasName

Формат описания:

ИмяБазы (имя_картотеки: С): С;

Входные параметры:

ИмяКартотеки — имя картотеки, под которым она распознается программой

Назначение:

Возвращает псевдоним базы (алиас в плане бухгалтерии), в которой находится картотека с заданным именем.

Пример:

```
Func UserName(Ob:string):string;
var Robl:integer;
var colrec: integer;
var i :integer;
var rez:logical;
  rez=false;
  If Pos(«SQL»,UP(Version))>0:
     return UserName(AliasName(Ob));
      else
      return User=»»;
   Fi;
End;
```

Функция ИмяПользователя

Б

Синоним:

UserName

Формат описания:

ИмяПользователя (ПсевдонимБД : С) : С;

Входные параметры:

ПсевдонимБД — псевдоним базы данных (идентификатор), под которым она распознается программой

Назначение:

По заданному псевдониму базы данных возвращает имя пользователя, подключившегося к данной базе данных в текущем сеансе работы.

Пример:

```
Пользователь = ИмяПользователя (“Сервер_БД1”);
-- Пользователь = “NNN”, если в плане бухгалтерии задана следующая директива:
-- БАЗА_ДАННЫХ Сервер_БД1 =:123.456.789.115(Дир) Пользователь “NNN”
```

Функция ИмяСхемы

Б

Синоним:

SchemaName

Формат описания:

ИмяСхемы (ПсевдонимБД : С) : С;

Входные параметры:

ПсевдонимБД — псевдоним базы данных (идентификатор), под которым она распознается программой

Назначение:

По заданному псевдониму базы данных возвращает имя текущей схемы доступа указанной базы данных.

Пример:

```
СхемаДоступа = ИмяСхемы ("Сервер_БД1");
-- ИмяСхемы = "Дир", если в плане бухгалтерии задана следующая директива:
-- БАЗА_ДАННЫХ Сервер_БД1 =:123.456.789.115(Дир) Пользователь "NNN"
```

Функция Календарь

Б

Синоним:

Calendar

Формат описания:

Календарь ([НачДата : Д]) : Д;

Входные параметры:

НачДата — начальная дата, которая будет "текущей" при открытии окна с календарем

Назначение:

Открывает модальное окно с календарем и возвращает выбранную дату. Если функция вызвана с параметром, то при открытии календаря текущей будет заданная дата.

Пример:

```
ДатаПоездки = Календарь;
```

Функция Калькулятор

Б

Синоним:

Calculator

Формат описания:

Калькулятор [(Значение : Ч)] : Ч;

Входные параметры:

Значение — значение, передаваемое в диалог калькулятора (необязательный параметр)

Назначение:

Открывает модальное окно с калькулятором и возвращает значение числового типа — результат вычисления.

Процедура КонтекстнаяСправка

Б

Синоним:

HelpContext

Формат описания:

КонтекстнаяСправка (ИмяФайлаСпр : С; ТипСпр : Ц; НомерРазд : Ц);

Входные параметры:

ИмяФайлаСпр — имя файла справки

ТипСпр — тип вызываемой справки:

1 — вызов справки по номеру темы

3 — вызов содержания Справочной системы

4 — вызов справки по использованию Справочной системы Windows

НомерРазд — номер темы в Справочной системе

Назначение:

Вызывает Справочную систему к прикладным системам, разрабатываемым с помощью программы Турбо Бухгалтер.

Примечание:

Для использования вызова Справочной системы рекомендуется применять промежуточную процедуру, которая обращается к конкретному файлу помощи. Для работы этой процедуры необходимо добавить директиву КАТАЛОГИ в текущий план бухгалтерии, в

которой описывается путь для доступа к файлу Справочной системы (help-файлу):

```
КАТАЛОГИ HELP = "d:\...\myhelp.hlp"
```

Пример:

```
ПРОЦ ShowHelpContext (ContextID : Integer);
 ПЕРЕМ HelpFileName, SubString :String;
 Попытка
  HelpFileName = PlanVar ("HELP"); -- получаем имя help-файла
 Исключение
  -- Отсутствует директива КАТАЛОГИ с путем доступа к help-файлу
  Сообщение ("Нет справочной информации по данному контексту");
  Выход;
 Конец;
 SubString = Подстр (HelpFileName, 1, 1); -- Удаляем кавычки, если они есть
 IF (SubString="") or (SubString=""") :
  HelpFileName = Подстр (HelpFileName, 2, Length(HelpFileName)-2);
 Конец;
 КонтекстнаяСправка (HelpFileName, 1, ContextID); -- Вызов справки по
 -- номеру темы
КОНЕЦ;
```

Процедура Отчет

Б

Синоним:

Report

Формат описания:

Отчет (ИмяОтч : С, ДатаНач : Д, ДатаКон : Д, УслНаСч : С, УслНаПризн : С);

Входные параметры:

ИмяОтч — имя отчета из архива внутренних отчетов

ДатаНач, ДатаКон — дата начала и конца периода

УслНаСч, УслНаПризн — условия отбора аналитических счетов и признаков, соответственно

Назначение:

Создает внутренний отчет с заданным именем за определенный период по счетам и аналитическим признакам, удовлетворяющим введенным условиям, и открывает его на экране.

Примечания:

1. В первом параметре можно указать имя отчета как с заданием пути, так и без него. Допускается указывать как полный путь к отчету, так и путь относительно папки с текущей бухгалтерией. В качестве разделителя между путем и именем отчета использу-

ется “.” (точка) или вертикальная черта “|” (*см. примеры*). Если в качестве разделителя используется точка, то выражение для первого параметра заключается в одинарные кавычки, а путь и имя отчета записываются в двойных кавычках.

2. При задании в первом параметре только названия отчета (без папки) поиск его настроек выполняется последовательно по всем папкам отчетов в том порядке, в котором они перечислены в списке диалога “Внутренние отчеты”.
3. Если вместо параметров *УслНаСч* и/или *УслНаПризн* указаны пустые строки, то используются соответствующие значения из архива внутренних отчетов. Для задания пустых условий отбора следует использовать пробелы.

Примеры:

```
Отчет (“Оборотная ведомость”,01.01.98,01.01.99,“5*”,“”); -- задано только
-- имя отчета, сформируется внутренний отчет “Оборотная ведомость” за
-- период с 01.01.98 до 01.01.99 по счетам, начинающимся с цифры 5, и по
-- всем аналитическим признакам
Отчет ("По оборотам.lst".”Оборотная ведомость",01.01.98,01.01.99,“5*”,“”);
-- указывается путь от текущей бухгалтерии, в качестве разделителя задана
-- точка
Отчет (По оборотам.lst¦Оборотная ведомость,01.01.98,01.01.99,“5*”,“”);
-- указывается путь от текущей бухгалтерии, в качестве разделителя задана
-- вертикальная черта
Отчет ("C:\tbw65\Wrk\Пример\По оборотам.lst".”Оборотная
ведомость",01.01.98,01.01.99,“5*”,“”); -- задан полный путь
```

Функция ПеремПБ

Б

Синоним:

PlanVar

Формат описания:

ПеремПБ (Макрос : C);

Входные параметры:

Макрос — идентификатор макропараметра, заданного в директиве КАТАЛОГИ плана бухгалтерии

Назначение:

Возвращает полный путь к каталогу по заданному идентификатору.

Пример:

```
Путь = ПеремПБ (STD) -- Путь = “C:\TBW60\Std”
```

Функция ПроверкаЛицензии

Б

Синоним:

LicenseCheck

Формат описания:

ПроверкаЛицензии (Код : Ц) : Ц;

Входные параметры:

Код — код лицензии

Назначение:

Возвращает число лицензий (рабочих мест). Проверяет допустимость кода лицензии: если код лицензии неверный, то выдает на экран соответствующее сообщение и возвращает “0”.

Пример:

```
ПРОЦ Р1;
 ПЕРЕМ КодЛиц : Ц;
 Если ПроверкаЛицензии ( КодЛиц ) = 0:
  ЗакрытьПрог;
 Конец;
Конец;
```

Функция СчитатьПеремИзДБТ

Б

Синоним:

ReadVarFromDBT

Формат описания:

СчитатьПеремИзДБТ (ИмяФайла : С; [ИмяБланка : С]) : Л;

Входные параметры:

ИмяФайла — полное имя файла (с указанием пути), из которого производится чтение

ИмяБланка — имя бланка, в который считываются значения переменных

Назначение:

Записывает переменные бланка в файл формата DBT и возвращает “ЛОЖЬ” при успешной записи, “ИСТИНА” — при ошибке (например, файл “захвачен” другим приложением).

Примечания:

1. Если имя бланка не задано, то чтение производится в текущий бланк.
2. Читает как стандартный формат DBT (*ИмяБланка.ИмяПеременной=Значение*), при этом *ИмяБланка* игнорируется (оно задается при вызове функции), так и формат функции **Записать ПеремВДБТ** (*ИмяПеременной=Значение*).

Пример:

```
Проц Р1;
 Если (СчитатьПеремИзДБТ("С:\Прочти меня.dbt", "Документы.Касса.ПрихОрд")):
    Трассировка ("Произведено считывание данных в бланк");
 Конец;
Конец;
```

Процедуры и функции импорта/экспорта

Процедура ИмпортБазы

Б

Синоним:

ImportBase

Формат вызова:

ИмпортБазы (Формат : Ц; Файл : С; Режим : Л; DOS : Л; Подтв : Л; Действ : Ц; Удал : Л [; ПоказДиалог : Л]);

Входные параметры:

Формат — допустимая константа формата может принимать следующие значения:

- ieCRD — импорт из файла *.crd в заданную базу данных)
- ieDBF — импорт из файла *.dbf в заданную базу данных)

Файл — полное имя (путь к файлу и его имя) или короткое имя (только одно имя файла *.cdf без задания пути) файла из которого происходит импорт

Режим — логический параметр определяет режим загрузки полей в сетевой версии программы (если "ИСТИНА", то загружаются поля CreateTime, UpdateTime, CreatedUser, ModifiedUser, если "ЛОЖЬ" — данные поля не загружаются. В локальной версии параметр не используется ("ЛОЖЬ")

DOS — логический параметр определяет в какой кодировке загружать текст (если "ИСТИНА", то загружает текст в кодировке DOS, "ЛОЖЬ" — в кодировке Windows)

Подтв — логический параметр определяет запрашивать подтверждения при загрузке ("ИСТИНА") или нет ("ЛОЖЬ"). Если данный параметр равен "ИСТИНА", то следующий параметр *Действ* игнорируется

Действ — целочисленная константа для разрешения конфликтных ситуаций при импорте может принимать следующие значения:

- ovrReplдублЗаменить — заменить запись с одинаковыми кодами
- ovrAddlдублОбъединить — объединить запись с одинаковыми кодами
- ovrNewlдублДублировать — дублировать запись с одинаковыми кодами

- ovrNone|дублПропустить — игнорировать запись с одинаковыми кодами

Удал — логический параметр определяет удалять ли содержимое картотек при загрузке (“ИСТИНА”) или нет (“ЛОЖЬ”)

ПоказДиалог — показывать ли диалог соответствия полей.

Назначение:

Импорт данных из файла *.crd (константа ieCRD) или *.dbf (константа ieDBF), содержащего структуру базы данных, в базу данных.

Примечания:

1. В момент импорта файл, из которого импортируются данные, должен быть закрыт.

2. Если задано короткое имя файла, из которого происходит импорт, то поиск файла производится в папке с текущей бухгалтерией, иначе — в папке, путь к которой указан во втором параметре.

Пример:

```
Проц Р1(Объект:Строка);
 ИмпортБазы(ieDBF, "С:\МоиДокументы\База.dbf", Истина, Ложь, Ложь,
 дублЗаменить, Ложь); -- в папке МоиДокументы
 ImportCard(ieCRD, "База.crd", Истина, Ложь, Ложь, ovrAdd, Ложь);
 -- в текущей бухгалтерии
Конец;
```

Процедура ИмпортБланка

Б

Синоним:

ImportBlank

Формат вызова:

ИмпортБланка (Формат : Ц; Бланк : С; Файл : С; Замена : Л [; СохрЦвет : Л]);

Входные параметры:

Формат — допустимая константа формата может принимать следующие значения

- ieWord|изВорд (импорт из документа MS WORD *.doc в заданный бланк
- ieText|изТекст (импорт из текстового файла *.txt в заданный бланк)

Бланк — имя бланка (квалифицированный идентификатор), в который импортируется содержимое заданного файла, расширение которого зависит от константы формата

Файл — полное или короткое имя файла, из которого импортируются данные в заданный бланк

Замена — логический параметр, если он равен значению "ИСТИНА", то содержимое бланка заменяется при импорте, иначе — не заменяется

СохрЦвет — логический параметр, если он равен значению "ИСТИНА", то сохраняется цвет фона и шрифтов при импорте, иначе — не сохраняется. Параметр не учитывается при импорте в текстовый формат (константа ieText|изТекст)

Назначение:

Данная процедура осуществляет импорт содержимого документа MS WORD (константа формата ieWord|изВорд) или текстового документа (константа формата ieText|изТекст) в заданный бланк.

Примечания:

1. В момент импорта файл должен быть закрыт.

2. Если задано короткое имя файла, из которого происходит импорт, то файл ищется в папке с текущей бухгалтерией, иначе — в папке, путь к которой указан во втором параметре.

Пример:

```
Проц Р1(Объект:Строка);
 ИмпортБланка(ieWord, "Инфо.ТМЦ", "С:\МоиДокументы\КарточкаТМЦ.doc",
 Истина, Истина); -- в папке МоиДокументы
 ImportBlank(ieWord, "Инфо.ТМЦ", "КарточкаТМЦ.doc", Ложь, Истина);
 -- в текущей бухгалтерии
Конец;
```

Процедура ИмпортКартотеки

Б

Синоним:

ImportCard

Формат вызова:

ИмпортКартотеки (Формат : Ц; Картотека : С; Файл : С; ВидимПоля : Л; Режим : Л; DOS : Л; Подтв : Л; Действ : Ц; Удал : Л [; ПоказДиалог : Л] [; НастрБраузера : Л]);

Входные параметры:

Формат — допустимая константа формата может принимать следующие значения:

- ieCRD — импорт из файла *.crd в заданную картотеку
- ieDBF — импорт из файла *.dbf в заданную картотеку;

Картотека — обязательное имя картотеки, в которую импортируется заданный файл *.crd или *.dbf

Файл — полное или короткое имя файла со структурой заданной картотеки, из которого импортируются данные

ВидимПоля — логический параметр, если его значение равно “ИСТИНА”, то выгружаются только видимые поля, “ЛОЖЬ” — все поля

Режим — логический параметр определяет режим загрузки полей в сетевой версии программы (если “ИСТИНА”, то загружаются поля CreateTime, UpdateTime, CreatedUser, ModifiedUser, если “ЛОЖЬ” — данные поля не загружаются. В локальной версии параметр не используется (“ЛОЖЬ”)

DOS — логический параметр определяет в какой кодировке загружать текст (если “ИСТИНА”, то загружает текст в кодировке DOS, “ЛОЖЬ” — в кодировке Windows)

Подтв — логический параметр определяет запрашивать подтверждения при загрузке (“ИСТИНА”) или нет (“ЛОЖЬ”). Если данный параметр равен “ИСТИНА”, то следующий параметр *Действ* игнорируется

Действ — целочисленная константа для разрешения конфликтных ситуаций при импорте может принимать следующие значения:

- ovrReplдублЗаменить — заменить запись с одинаковыми кодами
- ovrAddlдублОбъединить — объединить запись с одинаковыми кодами
- ovrNewlдублДублировать — дублировать запись с одинаковыми кодами
- ovrNonelдублПропустить — игнорировать запись с одинаковыми кодами

Удал — логический параметр определяет удалять ли содержимое картотеки при загрузке (“ИСТИНА”) или нет (“ЛОЖЬ”)

ПоказДиалог — необязательный логический параметр определяет, показывать ли диалог соответствия полей

НастрБраузера — необязательный логический параметр, определяющий использование настроек браузера картотеки при загрузке. Если “ИСТИНА”, то настройки учитываются, если “ЛОЖЬ”, то не учитываются. По умолчанию — “ИСТИНА”.

Назначение:

Импорт из файла *.crd (константа ieCRD) или *.dbf (константа ieDBF) в заданную картотеку.

Примечания:

1. В момент импорта файл, из которого происходит импорт, должен быть закрыт.

2. Если задано короткое имя файла, из которого происходит импорт, то поиск файла производится в папке с текущей бухгалтерией, иначе — в папке, путь к которой указан во втором параметре.

3. Если в параметре *Картотека* указать имя файла браузера, то будут использоваться именно его настройки при импорте.

Пример:

```
Проц Р1(Объект:Строка);
 ИмпортКартотеки(ieDBF, "Сотрудники", "C:\МоиДокументы\Сотрудники.dbf",
 Истина, Ложь, Ложь, Ложь, дублЗаменить, Ложь); -- в папке МоиДокументы
 ImportCard(ieCRD, "Сотрудники", "Сотрудники.crd", Истина, Ложь, Ложь,
 Ложь, ovrAdd, Ложь); -- в текущей бухгалтерии
Конец;:
```

Процедура ЭкспортБазы

Б

Синоним:

ExportBase

Формат вызова:

ЭкспортБазы (Формат : Ц; Файл : С; Заголовок : Л; Режим : Л; DOS : Л);

Входные параметры

Формат — допустимая константа формата может принимать следующие значения:

- ieCRD — экспорт в формат *.crd
- ieDBF — экспорт в формат *.dbf
- ieExcel — экспорт в формат *.xls;

Файл — полное имя (путь к файлу и его имя) или короткое имя (только одно имя файла *.cdf без задания пути) файла в который происходит экспорт

Режим — логический параметр определяет режим выгрузки полей в сетевой версии программы (если “ИСТИНА”, то выгружаются поля CreateTime, UpdateTime, CreatedUser, ModifiedUser, если “ЛОЖЬ” — данные поля не выгружаются. В локальной версии параметр не используется (“ЛОЖЬ”)

DOS — логический параметр определяет в какой кодировке выгружать текст (если “ИСТИНА”, то выгружает текст в кодировке DOS, “ЛОЖЬ” — в кодировке Windows)

Назначение:

В зависимости от выбранной константы формата данная процедура осуществляет экспорт всей базы данных в следующие форматы:

- *.crd (константа формата ieCRD);
- *.dbf (константа формата ieDBF);
- *.xls документа MS EXCEL (константа формата ieExcel).

Примечания:

1. Экспорт всей базы данных в текстовый файл *.crd происходит при следующих условиях:

- выгружаются все записи без учета фильтра;
- выгружаются все поля, в том числе и невидимые;
- не выгружаются разыменованные поля;
- выгружаются поля, содержащие ссылку (Ukey);
- не выгружаются таблицы по ссылкам.

2. Если задано короткое имя файла, в который происходит экспорт бланка, то файл создается в папке с текущей бухгалтерией, иначе — в папке, путь к которой указан во втором параметре.

3. Параметр Заголовок учитывается только при экспорте в формат *.xls.

4. Параметр DOS не учитывается при экспорте в формат *.xls.

Пример:

```
Проц Р1(Объект:Строка);
 ЭкспортБазы(ieCRD, "С:\МоиДокументы\База.crd", Ложь, Истина, Ложь);
 -- в папке МоиДокументы
 ExportBase(ieCRD, "База.crd", Ложь, Истина, Истина);
 -- в текущей бухгалтерии
Конец;
```

Процедура ЭкспортБланка

Б

Синоним:

ExportBlank

Формат вызова:

ЭкспортБланка (Формат : Ц; Бланк : С; Файл : С [; СохрЦвет : Л]);

Входные параметры

Формат — допустимая константа формата может принимать следующие значения:

- ieRTF — экспорт бланка в формат *.rtf
- ieWord — экспорт бланка в формат *.doc документа MS WORD
- ieHTML — экспорт бланка в формат *.html
- ieExcel — экспорт в формат *.xls;

Бланк — имя бланка (квалифицированный идентификатор), который экспортируется в заданный файл, расширение которого зависит от константы формата

Файл — полное (путь к файлу и его имя) или короткое имя (только одно имя файла без задания пути) файла, в который производится экспорт

СохрЦвет — логический параметр, если он равен значению "ИСТИНА", то сохраняется цвет фона и шрифтов при экспорте, иначе — не сохраняется

Назначение:

В зависимости от выбранной константы формата данная процедура осуществляет экспорт бланка в следующие форматы:

- в формат *.rtf (константа формата ieRTF);
- в формат *.doc документа MS WORD (константа формата ieWord);
- в формат *.html (константа формата ieHTML);
- в формат *.xls документа MS EXCEL (константа формата ieExcel).

Примечания:

1. Если задано короткое имя файла, в который происходит экспорт бланка, то файл создается в папке с текущей бухгалтерией, иначе — в папке, путь к которой указан во втором параметре.

2. Если экспортируется текущий бланк, то вместо имени бланка может быть задана пустая строка, т.е. две кавычки ("").

Пример:

```
Проц Р1(Объект:Строка);
ЭкспортБланка(ieWord, "Инфо.ТМЦ", "С:\МоиДокументы\КарточкаТМЦ.doc",
Истина); -- в папке МоиДокументы
ExportBlank(ieWord, "Инфо.ТМЦ", "КарточкаТМЦ.doc", Ложь); -- в текущей
--бухгалтерии
Конец;
```

Процедура ЭкспортКартотеки

Б

Синоним:

ExportCard

Формат вызова:

ЭкспортКартотеки (Формат : Ц; Картотека : С | Картотеки[ ]: С; Файл : С; Фильтр : Л; ВидимПоля : Л; РазымПоля : Л; НомЗап : Л; Заголовок : Л; Режим : Л; DOS : Л; ТаблССыл : Л);

Входные параметры:

Формат — допустимая константа формата может принимать следующие значения:

- ieCRD — экспорт в формат *.crd
- ieDBF — экспорт в формат *.dbf
- ieExcel — экспорт в формат *.xls;

Картотека или *Картотеки[ ]*— обязательное имя картотеки (картотек), которая(ые) экспортируется(ются) в заданный файл *.crd

Файл — полное имя файла (путь к файлу и его имя) или короткое имя (только одно имя файла со структурой картотеки *.crd без задания пути), в который происходит экспорт

Фильтр — логический параметр, если его значение равно “ИСТИНА”, то выгрузка картотеки происходит с учетом фильтра, заданного в настройках картотеки

ВидимПоля — логический параметр, если его значение равно “ИСТИНА”, то выгружаются только видимые поля, “ЛОЖЬ” — все поля

РазымПоля — логический параметр, если его значение равно “ИСТИНА”, то будут выгружаться разыменованные поля

НомЗап — логический параметр, если его значение равно “ИСТИНА”, то будут выгружаться поля, содержащие ссылку (UKEY) на запись

Заголовок — логический параметр, если его значение равно “ИСТИНА”, то к каждому столбцу книги MS Excel будет указан заголовок. Данный параметр учитывается только при экспорте в формат *.xls

Режим — логический параметр определяет режим выгрузки полей в сетевой версии программы (если “ИСТИНА”, то выгружаются поля CreateTime, UpdateTime, CreatedUser, ModifiedUser, если “ЛОЖЬ” — данные поля не выгружаются. В локальной версии данный параметр не используется (“ЛОЖЬ”)

DOS — логический параметр определяет в какой кодировке выгружать текст (если “ИСТИНА”, то текст выгружается в кодировке DOS, “ЛОЖЬ” — в кодировке Windows). Данный параметр не учитывается при экспорте в формат *.xls

ТаблСсыл — логический параметр, если его значение равно "ИСТИНА", то выгружаются таблицы по ссылкам, "ЛОЖЬ" — только значения

Назначение:

В зависимости от выбранной константы формата данная процедура осуществляет экспорт картотеки в следующие форматы:

- *.crd (константа формата ieCRD);
- *.dbf (константа формата ieDBF);
- *.xls документа MS EXCEL (константа формата ieExcel).

Примечания:

1. Если задано короткое имя файла, в который происходит экспорт бланка, то файл создается в папке с текущей бухгалтерией, иначе — в папке, путь к которой указан во втором параметре.

2. Параметр *Заголовок* учитывается только при экспорте в формат *.xls.

3. Если в параметре *Картотека* указать имя файла браузера, то будут использоваться именно его настройки при экспорте.

Пример 1:

```
Проц Р1(Объект:Строка);
ЭкспортКартотеки("Сотрудники", ieCRD, "C:\МоиДокументы\Сотрудники.crd",
Истина, Ложь, Ложь, Ложь, Ложь, Истина, Ложь, Истина);
-- в папке МоиДокументы
ExportCard("Сотрудники", ieCRD, "Сотрудники.crd", Истина, Ложь, Ложь,
Ложь, Ложь, Истина, Ложь, Ложь); -- в текущей бухгалтерии
Конец;
```

Пример 2:

```
ExportCard(ieCrd, "ТМЦ;СДЕЛКА", "СДЕЛКИ", False, False, False, True,
False, False, False, False);
```

Пример 3:

```
var vTables[]: String;
 proc ExportManyCardFile;
  ClearVariable(vTables);
   vTables[1] = ТМЦ;
   vTables[2] = СДЕЛКА;
   ExportCard(ieCrd, vTables,СДЕЛКИ, False, False, False, True, False,
False, False, False);
 end;
```

Процедуры и функции для работы с SQL-запросами

Функция SQLЗапросыРазрешены

Синоним:

SQLQueryEnabled

Формат описания:

SQLЗапросыРазрешены : Л;

Назначение:

Возвращает значение “ИСТИНА”, если у пользователя в плане ТБ.Сервера установлена привилегия “SQL запросы”, то есть разрешены прямые SQL-запросы с помощью **ОткрытьРабочуюОбластьSQL**, **ВыполнитьSQL**. В локальной версии всегда возвращается значение “ЛОЖЬ”.

Пример

См. пример функции **ВзятьДействитИмяКартотеки**.

Функция ВзятьДействитИмяКартотеки

Синоним:

GetRealCardfileName

Формат описания:

ВзятьДействитИмяКартотеки (ИмяMTL : С) : С ;

Входные параметры:

ИмяMTL — логическое имя картотеки, описанное в MTL-файле

Назначение:

Возвращает физическое имя картотеки по ее логическому имени, описанному в MTL-файле.

Примечания:

1. В модификациях Турбо Бухгалтер 6 функция возвращает пустую строку.

2. Может использоваться при построении SQL-запросов к указанной картотеке в функции **ОткрытьРабочуюОбластьSQL** и процедуре **ВыполнитьSQL.**

Пример:

```
Проц Р1;
Перем ИмяКарт : Строка;
Перем РабОблSQL : Целое;
Если не SQLЗапросыРазрешены:
  Exit;
end;
ИмяКарт = ВзятьДействитИмяКартотеки("Сотрудники");
РабОблSQL = ОткрытьРабочуюОбластьSQL("Cool", "Select * From "+ИмяКарт+"
Where [Признак] = Пост.Ю.База1 ");
-- открываем рабочую область для работы с SQL запросами
 Попытка
  ВыполнитьSQL("Cool", "Select * From "+ИмяКарт+"  Where [Признак] =
      Пост.Ю.База1 ");
 Окончание
ЗакрытьРабочуюОбласть(РабОблSQL);
-- Закрываем рабочую область
Конец;
Конец;
```

Процедура ВыполнитьSQL

Синоним:

ExecuteSQL

Формат описания:

ВыполнитьSQL (ИмяБД : С, SQLЗапрос : С | ДлСтр [, ИмяКартотеки : С]);

Входные параметры:

ИмяБД — имя базы данных (псевдоним, записанный в директиве **Базы_Данных** плана бухгалтерии), к которой производится запрос. Данный параметр может быть пустая строка, при условии, что задан третий параметр (картотека). В этом случае, алиас будет вычисляться по имени картотеки

SQLЗапрос — текст запроса на языке SQL

ИмяКартотеки — имя картотеки

Назначение:

Выполняет указанный SQL-запрос, не связанный с получением результирующей таблицы (например, запуск хранимой процедуры).

Примечания:

1. В модификациях Турбо Бухгалтер 6  процедура ничего не выполняет.
2. Данную процедуру могут использовать только те пользователи, которые имеют право выполнять SQL-запросы. Для получения этого права необходимо на сервере установить флаг в столбце **SQL-запросы** страницы "Пользователи" диалога "Настройка плана сервера".

Пример:

```
Проц Р1;
   Перем РабОблSQL: Целое;
   -- открываем рабочую область для работы с SQL запросами
   РабОблSQL = ОткрытьРабочуюОбластьSQL ("Cool", "Select * From
         [Пример]..[СОТРУДНИКИ] Where [Признак] = Пост.Ю.База1 ");
   Попытка
      ВыполнитьSQL ("Cool", "Select * From [Пример]..[СОТРУДНИКИ] Where
         [Признак] = Пост.Ю.База1 ");
   Окончание
      -- Закрываем рабочую область
      ЗакрытьРабочуюОбласть (РабОблSQL);
   Конец;
Конец;
```

Процедура ОбновитьКартотеку

Синоним:

RefreshCard

Формат описания:

ОбновитьКартотеку (ИмяКартотеки : С [; ДатаНач : Д]);

Входные параметры:

ИмяКартотеки — имя картотеки

ДатаНач — дата, начиная с которой программа считает данные картотеки изменившимися. (При отсутствии данного параметра учитываются изменения за все даты.)

Назначение:

Принудительно обновляет данные картотеки на клиентских местах в случаях, когда изменения в картотеках вносятся, минуя Сервер ТБ (например, с помощью прямых SQL-запросов через функцию **ВыполнитьSQL** или другими программами).

Примечания:

1. В модификациях Турбо Бухгалтер 6  ничего не выполняет.
2. Для корректной работы программы на клиентских местах настоятельно рекомендуется при изменении записи картотеки прямыми SQL-запросами или другими программами обновлять поле *UpdateTime*, а при добавлении записи указывать дату создания записи в поле *CreateTime*.

Пример:

```
Проц ОбновлениеДанных (Стр : Строка);
   Перем ИмяКарт : Строка;
   ИмяКарт = ВзятьДействитИмяКартотеки ("ПрихОрд");
   -- Записать новое значение в поле "Сумма" картотеки "ПрихОрд" в запись,
   -- например, с номером Ukey = 1
   ВыполнитьSQL (Cool, Update + ИмяКарт + " SET [UpdateTime] = 20001030
                       00:00:00.0 , [Сумма] = 245.10 Where [Ukey] = 1");
   -- Обновление картотеки на клиентских местах, начиная с даты 30.10.2000
   ОбновитьКартотеку ("ПрихОрд", 30.10.2000);
Конец;
```

Функция ОткрытьРабочуюОбластьSQL

Синоним:

OpenWorkAreaSQL

Формат описания:

ОткрытьРабочуюОбластьSQL (ИмяБД : С; SQLЗапрос : С | ДлСтр [; ИмяКартотеки : С]) : Ц;

Входные параметры:

ИмяБД — имя базы данных (псевдоним, записанный в директиве Базы_Данных плана бухгалтерии), к которой производится запрос. Данный параметр может быть пустая строка, при условии, что задан третий параметр (картотека). В этом случае, алиас будет вычисляться по имени картотеки.

SQLЗапрос — текст запроса на языке SQL

ИмяКартотеки — имя картотеки

Назначение:

Открывает рабочую область по SQL-запросу и возвращает ее номер.

Примечания:

1. В модификациях Турбо Бухгалтер 6  функция возвращает номер неоткрытой рабочей области.

2. Рабочая область, открытая таким образом, доступна только на чтение.
3. Все остальные операции, в т.ч. и закрытие рабочей области, соответствуют обычным операциям с рабочей областью.
4. Данную функцию могут использовать только те пользователи, которые имеют право выполнять SQL-запросы. Для получения этого права необходимо на сервере установить флаг в столбце **SQL-запросы** страницы "Пользователи" диалога "Настройка плана сервера".
5. Если картотека не указана, то для открытия рабочей области на Сервере ТБ используется любое подключение к MS SQL. Если картотека указана, то открытие рабочей области на Сервере ТБ будет производиться через то же подключение, что и при работе с указанной картотекой.
6. При работе с СУБД Oracle необходимо обязательно указывать имя картотеки.

Пример:

```
Проц Р1;
   Перем РабОблSQL: Целое;
   -- открываем рабочую область для работы с SQL запросами
   РабОблSQL = ОткрытьРабочуюОбластьSQL ("Cool", "Select * From
         [Пример]..[ЮРЛИЦО] Where [Признак] = 'Пост.Ю.База1' ");
   Попытка
      ВыполнитьSQL ("Cool", "Select * From [Пример]..[ЮРЛИЦО] Where
                                    [Признак] = 'Пост.Ю.База1' ");
   Окончание
      -- Закрываем рабочую область
      ЗакрытьРабочуюОбласть (РабОблSQL);
   Конец;
Конец;
```

Константы бланков

Константы СчетАктивный, СчетПассивный, СчетАктивноПассиный, СчетНулевой

Б

(для функции СчетАктивный)

Синонимы:

AccIsActive, AccIsPassive, AccIsActPas, AccIsZero

Назначение:

Функция **IsAccActive (СчетАктивный)** с помощью этих констант возвращает следующие значения:

Константа	Возвращаемое значение
СчетАктивный	счет активный
СчетПассивный	счет пассивный
СчетАктивноПассивный	счет активно-пассивный
СчетНулевой	счет нулевой

Пример:

```
if A = AccIsActive : Message ("Счет активный");
if A = AccIsPassive : Message ("Счет пассивный");
```

См. также:

Функцию **СчетАктивный**

Константы кмдВерно, кмдДа, кмдНет, кмдОтказ

Б

Синонимы:

cmOK, cmYes, cmNo, cmCancel

Назначение:

Эти константы возвращаются функциями вывода запросов на экран и обозначают нажатую кнопку:

кмдВерно = 10 — нажата кнопка **ОК**

кмдНет = 13 — нажата кнопка **Нет**

кмдДа = 12 — нажата кнопка **Да**

кмдОтказ = 11 — нажата кнопка **Отказ**

Пример:

```
ПРОЦ Р1;
 Если ( ВопрДаНетОтказ ("Вы будете обедать ?") = кмдДа ) :
  Сообщение ("Приятного аппетита");
 Илсе;
ЦОРП;
```

См. также:

Функции Ввод, ВопрДаОтказ, ВопрДаНетОтказ, процедуру **Сообщение**

Константа Пусто

Б

Синоним:

Nil

Назначение:

Безадресная (пустая) ссылка.

Пример:

```
ПРОЦ Р1;
  ПЕРЕМ R : Запись;
  -- открываем картотеку в модальном окне для выбора записи пользователем
  R = ОткрытьКартотеку ("Crd1", "");
  Если R <> Пусто:
    Сообщение ("Запись выбрана");
   Истина
    Сообщение ("Запись не выбрана");
ЦОРП;
```

Константы ОтчСверн, ОтчДеб, ОтчКре

Б

(для функций типа Отчет)*

Синонимы:

repConj, repDeb, repCre

Назначение:

Эти константы возвращаются функциями типа Отчет:

отчСверн = 0 — свернутое сальдо

отчДеб = 1 — дебетовое сальдо

отчКре = 2 — кредитовое сальдо

Примеры:

```
TD[Секц] = ОтчОборот (Отч, отчДеб, РУБ); -- оборот по строке отчета
TC[Секц] = ОтчОборот (Отч, отчКре, РУБ); -- оборот по строке отчета
```

См. также:

Функции ОтчОстатокН, ОтчОстатокК, ОтчОборот, ОтчИтог, ОтчИтогОстатокН, ОтчИтогОстатокК, ОтчИтогОборот

Константы чтения\записи свойств клетки *(для метода секции Cell)*

Б

Константы

КлеткаСтильШрифта, КлеткаРазмерШрифта, КлеткаЦветШрифта, КлеткаБордюр, КлеткаРамка, КлеткаВыравнивание, КлеткаСвертка, КлеткаФормат, КлеткаТолкоВывод, КлеткаЦвет, КлеткаСКнопкой, КлеткаСодержимое, КлеткаТипПоля, КлеткаКоличествоСтрок

Синонимы

CellFontStyle, CellFontSize, CellFontColor, CellBevel, CellBorder, CellAlignment, CellWrap, CellFormat, CellReadOnly, CellBgColor, CellLookup, CellContents, CellFieldType, CellLinesCount

Назначение:

С помощью этих констант устанавливается определение параметров формата клеток секций в шаблонах. Описание метода *см. Приложение 2 Руководства программиста.*

Property — форматируемый параметр клетки, может принимать следующие значения:

CellFontStyle | КлеткаСтильШрифта — стиль задаваемого шрифта

CellFontSize | КлеткаРазмерШрифта — размер задаваемого шрифта

CellFontColor | КлеткаЦветШрифта — цвет задаваемого шрифта

CellBevel | КлеткаБордюр — бордюр вокруг клетки

CellBorder | КлеткаРамка — рамка вокруг клетки

CellAlignment | КлеткаВыравнивание — выравнивание выводимого текста в клетке

CellWrap | КлеткаСвертка — переносить не поместившийся по ширине клетки конец текста на следующую строку или нет

CellFormat | КлеткаФормат — формат текста в клетке

CellReadOnly | КлеткаТолкоВывод — только вывод

CellBgColor | КлеткаЦвет — цвет клетки

CellValue | КлеткаЗначение — значение в клетке (значение переменной или статический текст). Только для полей ввода\вывода

CellContents | КлеткаСодержимое — содержимое клетки (переменные). Только для полей ввода\вывода

CellText | КлеткаТекст — текст клетки, выводимый на экран

CellLookup|КлеткаСКнопкой — поле имеет кнопку выбора для конкретной клетки

CellContents|КлеткаСодержимое — только для полей ввода\вывода изменяет саму переменную поля по его имени. Если указан 5-й параметр *Запись* и равен значению "True", то производится запись в свойство **CellContents**

ЗАМЕЧАНИЕ. *Если новая переменная имеет другой тип, то воспользуйтесь методом изменения типа поля* ***CellFieldType (КлеткаТипПоля)*** *(см. ниже).*

CellFieldType|КлеткаТипПоля — считывает/изменяет тип поля конкретной клетки

Для анализа этого свойства нужно использовать следующие константы:

cftStaticText (ктпСтатическийТекст) = 1

cftCommonField (ктпОбщееПоле) = 2

cftStringField (ктпСтроковоеПоле) = 3

cftNumberField (ктпЧисловоеПоле) = 4

cftDateField (ктпПолеДаты) = 5

cftFieldEnum (ктпПеречислимоеПоле) = 6

cftReferenceField (ктпСсылочноеПоле) = 7

cftCalcField (ктпВычислимоеПоле) = 8

CellLinesCount|КлеткаКоличествоСтрок — считывает/изменяет количество строк в клетке

CellEventOnEnter | КлеткаСобытиеПриВходе, CellEventOnVerify | КлеткаСобытиеПриПроверке, CellEventOnExit | КлеткаСобытиеПриВыходе, CellEventOnType |КлеткаСобытиеПриНаборе, CellEventOnLookup|КлеткаСобытиеПриОбзоре, CellEventOnClick

| **КлеткаСобытиеПриНажатии** — назначает соответствующее событие во время исполнения кода бланка

Примеры:

```
СекРеквМК.Cell(5, 11, CellBgColor, coTransparent); --устанавливаем бесцветный цвет клетки
СекРеквМК.Cell(5, 11, CellFontColor, coWhite); --устанавливаем черный цвет шрифта в клетке
vLinesCount = AsInteger(Section1.Cell(1, 1, CellLinesCount, "Oops Oops Oops Oops Oops Oops Oops Oops"));
-- Переменная vLinesCount будет содержать кол-во строк на кот. был бы развернут текст, в случае, если тот назначить клетке.
```

См. также:

Константы цвета **цв***, константы шрифта **сф*** объекта Section

Константы цвета цв*

Б

Константы

цвПрозрачный, цвЧерный, цвБелый, цвТемноКрасный, цвКрасный, цвТемноАква, цвАква, цвТемноСиний, цвСиний, цвСалатовый, цвЗелёный, цвОливковый, цвЖелтый, цвСерый, цвСветлоСерый, цвФиолетовый, цвРозовый

Синонимы:

coTransparent, coBlack, coWhite, coMaroon, coRed, coTeal, coAqua, coNavy, coBlue, coLime, coGreen, coOlive, coYellow, coGray, coLtGray, coPurple, coFuchsia, coDkGray, coSilver

Системные константы

coBackground — цвет заднего плана

coActiveCaption — цвет заголовка активного окна

coMenu — цвет фона меню

coWindow — задний план (фон) окна

coWindowFrame — цвет фрейма окна

coMenuText — цвет текста меню

coWindowText — цвет текста в окне

coCaptionText — цвет текста в заголовке активного окна

coActiveBorder — цвет рамки активного окна

coInactiveBorder — цвет рамки неактивного окна

coAppWorkSpace — цвет рабочего пространства приложения

coHighlight - цвет выделения текста

coBtnFace - текст кнопки

coBtnShadow - цвет тени кнопки

coGrayText - цвет неактивного текста

coBtnText - текст на кнопке

coInactiveCaptionText - цвет текста заголовка неактивного окна

coBtnHighlight - цвет подсвеченной (в фокусе) кнопки

Назначение:

Эти константы используются для задания цвета константами чтения/записи свойств клетки. Т.е. вышеперечисленные константы являются ничем иным как параметром *Value* для параметра *Prop* для метода секции Cell *(см. Приложение 2 к Руководству программиста)*.

Примеры:

```
СекРеквМК.Cell(5, 11, CellBgColor, coTransparent); --устанавливаем
прозрачный цвет клетки (у клетки будет тот же цвет, что и у шаблона)
СекРеквМК.Cell(5, 11, CellFontColor, coWhite); --устанавливаем черный
                                                цвет шрифта в клетке
```

См. также:

Константы чтения\записи свойств клетки (для метода секции **Cell**)

Константы шрифта сф*

Б

Константы

сфЖирность, сфНаклон, сфПодчерк, сфЗачерк

Синонимы

fsBold, fsItalic, fsUnderline, fsStrikeout

Назначение:

Эти константы используются для задания шрифта константами чтения/записи свойств клетки. Т.е. вышеперечисленные константы являются ничем иным как параметром *Value* для параметра *Prop* для метода секции Cell *(см. Приложение 2 к Руководству программиста)*.

Пример

```
СекРеквМК.Cell(5, 11, CellFontStyle, fsBold);
--устанавливаем жирный шрифт в клетке
```

См. также:

Константы чтения\записи свойств клетки (для метода секции **Cell**)

Константы чтения\записи свойств столбца *(для метода секции Column)*

Б

Константы

СтбВидим, СтбШирина

Синонимы

ColVisible, ColWidth

Назначение:

С помощью этих констант устанавливается определение формата и свойств столбцов секций в шаблонах. Описание метода *см. Приложение 2 Руководства программиста.*

Property — форматируемый параметр столбца, может принимать следующие значения:

ColVisible|СтбВидим — видимость столбца

ColWidth|СтбШирина — ширина столбца

ColPrinted|СтбПечать — выводит i-столбец на печать

Примеры

```
СекРеквМК.Column(5, ColumnWidth, 120); --  устанавливаем ширину столбца
                                           равной 12 мм
```

Константы чтения\записи свойств строки *(для метода секции Row)*

Б

Константы

СтрВидим, СтрВысота, СтрПечать

Синонимы

RowVisible, RowHeght, RowPrinted

Назначение:

С помощью этих констант устанавливается определение формата и свойств строк секций в шаблонах. Описание метода *см. Приложение 2 Руководства программиста.*

Property — форматируемый параметр строки, может принимать следующие значения:

RowVisible|СтрВидим — видимость строки

RowWidth|СтрВысота — высота строки

RowPrinted|СтрПечать — выводит i-строку на печать

Примеры

```
СекРеквМК.Row(5, RowWidth, 120); --  устанавливаем высоту строки
                                     равной 12 мм
```

Алфавитный указатель

Л

М

Н

О

П

Р

С

Т

У

Ф

Ц

Ч

Э

Английская нотация

Содержание