# Оглавление

[Оглавление 1](#_Toc119732847)

[Глава 2. Рабочая область Microsoft Excel 7](#_Toc119732848)

[Рабочая книга 7](#_Toc119732849)

[Быстрое размножение рабочей книги 7](#_Toc119732850)

[Сохранение рабочей книги с именем, представляющим собой текущую дату 7](#_Toc119732851)

[Создание книги с одним листом 7](#_Toc119732852)

[Установка и снятие защиты рабочей книги 7](#_Toc119732853)

[Закрытие рабочей книги только при выполнении условия 8](#_Toc119732854)

[Быстрое удаление из рабочей книги ненужных имен 8](#_Toc119732855)

[Сортировка листов в текущей рабочей книге 9](#_Toc119732856)

[Импорт данных, для которых нужно более 256 столбцов 12](#_Toc119732857)

[Рабочий лист 14](#_Toc119732858)

[Новый лист — через макрос 14](#_Toc119732859)

[Блокировка использования контекстного меню 14](#_Toc119732860)

[Вставка колонтитула с именем книги, листа и текущей датой 15](#_Toc119732861)

[Проверка существования листа 15](#_Toc119732862)

[Проверка, защищен ли рабочий лист 15](#_Toc119732863)

[Сколько страниц на всех листах? 16](#_Toc119732864)

[Автоматический пересчет данных таблицы при изменении ее значений 16](#_Toc119732865)

[Ячейка и диапазон 17](#_Toc119732866)

[Автоматизация ввода данных в ячейки 17](#_Toc119732867)

[Выделение диапазона над текущей ячейкой 17](#_Toc119732868)

[Поиск ближайшей пустой ячейки столбца 18](#_Toc119732869)

[Поиск максимального значения в диапазоне 18](#_Toc119732870)

[Автоматическая замена значений диапазона 18](#_Toc119732871)

[Быстрое заполнение диапазона 19](#_Toc119732872)

[Первый способ 19](#_Toc119732873)

[Второй способ 20](#_Toc119732874)

[Третий способ 20](#_Toc119732875)

[Помещение в ячейку электронных часов 21](#_Toc119732876)

[«Будильник» 21](#_Toc119732877)

[Поиск данных в диапазоне 22](#_Toc119732878)

[Поиск в диапазоне значения по шаблону 22](#_Toc119732879)

[Поиск значения с выводом результата 22](#_Toc119732880)

[Поиск с выделением найденных данных 22](#_Toc119732881)

[Создание цветной границы диапазона 23](#_Toc119732882)

[Автоматическое определение адреса ячейки 23](#_Toc119732883)

[Автоматизация добавления примечаний в указанном диапазоне 24](#_Toc119732884)

[Заливка диапазона 24](#_Toc119732885)

[Ввод строго ограниченных значений в указанный диапазон 24](#_Toc119732886)

[Ввод данных с помощью диалогового окна 24](#_Toc119732887)

[Непосредственный ввод данных 25](#_Toc119732888)

[Последовательный ввод данных 26](#_Toc119732889)

[Быстрое выделение ячеек с отрицательными значениями 27](#_Toc119732890)

[Получение информации о выделенном диапазоне 27](#_Toc119732891)

[Панель инструментов изменения числового формата ячейки 29](#_Toc119732892)

[Тестирование скорости чтения и записи диапазонов 30](#_Toc119732893)

[Разработка и применение полезных пользовательских функций 31](#_Toc119732894)

[Объединение данных диапазона 31](#_Toc119732895)

[Объединение данных с учетом форматов 32](#_Toc119732896)

[Эксперименты с датой 32](#_Toc119732897)

[Выбор из текста всех чисел 32](#_Toc119732898)

[Прописная буква только в начале текста 33](#_Toc119732899)

[Перевод чисел в «деньги» 33](#_Toc119732900)

[Подсчет количества повторов искомого текста 34](#_Toc119732901)

[Суммирование данных только видимых ячеек 35](#_Toc119732902)

[При суммировании — курсор внутри диапазона 35](#_Toc119732903)

[Начисление процентов в зависимости от суммы 35](#_Toc119732904)

[Еще о расчете процентов 36](#_Toc119732905)

[Сводный пример расчета комиссионного вознаграждения 37](#_Toc119732906)

[Подсчет количества ячеек, содержащих указанные значения 38](#_Toc119732907)

[Подсчет количества видимых ячеек в диапазоне 39](#_Toc119732908)

[Поиск ближайшего понедельника 39](#_Toc119732909)

[Подсчет количества полных лет 39](#_Toc119732910)

[Проверка, была ли сохранена рабочая книга 40](#_Toc119732911)

[Расчет средневзвешенного значения 40](#_Toc119732912)

[Преобразование номера месяца в его название 40](#_Toc119732913)

[Расчет суммы первых значений диапазона 41](#_Toc119732914)

[Поиск последней непустой ячейки диапазона 41](#_Toc119732915)

[Поиск последней непустой ячейки столбца 42](#_Toc119732916)

[Поиск последней непустой ячейки строки 42](#_Toc119732917)

[Подсчет количества ячеек в диапазоне, содержащих указанные значения 42](#_Toc119732918)

[Англоязычный текст — заглавными буквами 42](#_Toc119732919)

[Отображение текста «задом наперед» 43](#_Toc119732920)

[Поиск максимального значения на всех листах книги 43](#_Toc119732921)

[Использование относительных ссылок 44](#_Toc119732922)

[Определение типа данных ячейки 44](#_Toc119732923)

[Выделение из текста произвольного элемента 45](#_Toc119732924)

[Генератор случайных чисел 46](#_Toc119732925)

[Случайные числа — на основании диапазона 47](#_Toc119732926)

[Глава 3. Создание трюков с помощью макросов 48](#_Toc119732927)

[Подсчет количества открытий файла 48](#_Toc119732928)

[Получение «закрытой» информации 48](#_Toc119732929)

[Произвольный текст в строке состояния 48](#_Toc119732930)

[Быстрое изменение заголовка окна 49](#_Toc119732931)

[Ввод данных с помощью диалогового окна 49](#_Toc119732932)

[Применение функции без ввода ее в ячейку 50](#_Toc119732933)

[Скрытие строк и столбцов от посторонних 50](#_Toc119732934)

[Быстрое выделение ячеек, расположенных через интервал 51](#_Toc119732935)

[Определение количества ячеек в диапазоне и суммы их значений 52](#_Toc119732936)

[Подсчет именованных объектов 53](#_Toc119732937)

[Быстрый поиск курсора 53](#_Toc119732938)

[Поиск начала и окончания диапазона, содержащего данные 53](#_Toc119732939)

[Трюки с примечаниями 54](#_Toc119732940)

[Подсчет примечаний 54](#_Toc119732941)

[Вывод на экран всех примечаний рабочего листа 54](#_Toc119732942)

[Создание списка примечаний рабочего листа 55](#_Toc119732943)

[Несколько трюков в одном примере 56](#_Toc119732944)

[Дополнение панели инструментов 58](#_Toc119732945)

[Примеры создания панелей инструментов 58](#_Toc119732946)

[Формирование пользовательского меню 60](#_Toc119732947)

[Проверка наличия файла по указанному пути 61](#_Toc119732948)

[Автоматизация удаления файлов 63](#_Toc119732949)

[Перечень имен листов в виде гиперссылок 64](#_Toc119732950)

[Удаление пустых строк на рабочем листе 64](#_Toc119732951)

[Запись текущих данных в текстовый файл 65](#_Toc119732952)

[Экспорт и импорт данных 66](#_Toc119732953)

[Одновременное умножение всех данных диапазона 67](#_Toc119732954)

[Преобразование таблицы Excel в HTML-формат 68](#_Toc119732955)

[Поиск данных нештатными средствами 71](#_Toc119732956)

[Включение автофильтра с помощью макроса 72](#_Toc119732957)

[Трюки с форматированием 72](#_Toc119732958)

[Изменение формата представления чисел нештатными средствами 72](#_Toc119732959)

[Помещение последнего символа ячейки над строкой 73](#_Toc119732960)

[Создание нестандартной рамки 73](#_Toc119732961)

[Быстрая вставка фамилий должностных лиц в документ 74](#_Toc119732962)

[Вызов окна настройки шрифта 74](#_Toc119732963)

[Вывод информации о текущем документе 74](#_Toc119732964)

[Вывод результата расчетов в отдельном окне 75](#_Toc119732965)

[Вывод разрешения монитора 75](#_Toc119732966)

[Что открыто в данный момент? 75](#_Toc119732967)

[Создание бегущей строки 76](#_Toc119732968)

[Создание бегущей картинки 76](#_Toc119732969)

[Мигающая ячейка 78](#_Toc119732970)

[Вращающиеся автофигуры 78](#_Toc119732971)

[Вызов таблицы цветов 80](#_Toc119732972)

[Создание калькулятора 81](#_Toc119732973)

[Еще о создании пользовательских меню 81](#_Toc119732974)

[Меню с пользовательскими командами 81](#_Toc119732975)

[Замена стандартного меню пользовательским 85](#_Toc119732976)

[Склонение фамилии, имени и отчества 85](#_Toc119732977)

[Получение информации об используемом принтере 90](#_Toc119732978)

[Вывод текущей даты и времени 91](#_Toc119732979)

[Автоматическое создание документов Word на основе табличных данных Excel 92](#_Toc119732980)

[Создание списка панелей инструментов и контекстных меню 94](#_Toc119732981)

[Создание списка пунктов главного меню Excel 94](#_Toc119732982)

[Создание списка пунктов контекстных меню 95](#_Toc119732983)

[Отображение панели инструментов при определенном условии 96](#_Toc119732984)

[Скрытие и отображение панелей инструментов 97](#_Toc119732985)

[Создание меню на основе данных рабочего листа 98](#_Toc119732986)

[Создание контекстного меню 101](#_Toc119732987)

[Просмотр содержимого папки 103](#_Toc119732988)

[Получение информации о состоянии дисков 107](#_Toc119732989)

[Расчет среднего арифметического 108](#_Toc119732990)

[Вывод списка доступных шрифтов 109](#_Toc119732991)

[Обработка сразу нескольких внешних файлов 109](#_Toc119732992)

[Запуск таблицы символов из Excel 110](#_Toc119732993)

[Создание раскрывающегося списка 111](#_Toc119732994)

[Добавление команды в меню 112](#_Toc119732995)

[Добавление команды в меню Сервис 112](#_Toc119732996)

[Добавление команды в меню Вид 113](#_Toc119732997)

[Глава 4. Трюки и эффекты с помощником 115](#_Toc119732998)

[Мультфильм с помощником в главной роли 115](#_Toc119732999)

[Дополнение помощника текстом, заголовком, кнопкой и значком 117](#_Toc119733000)

[Новые параметры помощника 117](#_Toc119733001)

[Использование помощника для выбора цвета заливки 118](#_Toc119733002)

[Глава 5. Эксперименты с диаграммами 119](#_Toc119733003)

[Построение диаграммы с помощью макроса 119](#_Toc119733004)

[Сохранение диаграммы в отдельном файле 120](#_Toc119733005)

[Построение и удаление диаграммы нажатием одной кнопки 121](#_Toc119733006)

[Вывод списка диаграмм в отдельном окне 122](#_Toc119733007)

[Применение случайной цветовой палитры 123](#_Toc119733008)

[Эффект прозрачности диаграммы 123](#_Toc119733009)

[Построение диаграммы на основе данных нескольких рабочих листов 125](#_Toc119733010)

[Создание подписей к данным диаграммы 127](#_Toc119733011)

[Глава 6. Создание полезных программ 127](#_Toc119733012)

[Программа для составления кроссвордов 127](#_Toc119733013)

[Игра «Минное поле» 131](#_Toc119733014)

[Игра «Угадай животное» 133](#_Toc119733015)

[Расчет на основании ячеек определенного цвета 136](#_Toc119733016)

# Глава 2. Рабочая область Microsoft Excel

## Рабочая книга

### Быстрое размножение рабочей книги

Листинг 2.1. Размножение рабочей книги

Sub DuplicateBook()

 Dim avarFileNames As Variant

 ' Формирование массива из путей для копий книги

 avarFileNames = Array("C:\" & \_

 ActiveWorkbook.Name, "D:\" & ActiveWorkbook.Name)

 ' Сохранение книги

 ActiveWorkbook.SaveAs avarFileNames

End Sub

### Сохранение рабочей книги с именем, представляющим собой текущую дату

Листинг 2.2. Сохранение книги с именем, представляющим собой текущую дату

Sub SaveAsDate()

 Dim strDate As String

 ' Получение текущей даты и представление ее в формате "ддммгг"

 strDate = Format(Now(), "ddmmyy")

 ' Сохранение книги в текущую папку под новым именем

 ActiveWorkbook.SaveAs ActiveWorkbook.Path & "\" & strDate

End Sub

### Создание книги с одним листом

Листинг 2.3. Книга с одним листом

Sub NewOneSheetBook()

 Workbooks.Add xlWBATWorksheet

End Sub

### Установка и снятие защиты рабочей книги

Листинг 2.4. Защита рабочей книги

Sub Worksheet\_BeforeRightClick(ByVal Target As Range, \_

 Cancel As Boolean)

 If Target.Address = "$D$2" Then

 ' Установка защиты рабочей книги (с паролем "123", \_

 включенной защитой структуры книги и защитой расположения \_

 окон)

 ThisWorkbook.Protect "123", True, True

 ' Указание не обрабатывать нажатие кнопки мыши \_

 в этой ячейке

 Cancel = True

 ElseIf Target.Address = "$E$5" Then

 ' Снятие защиты с книги (необходимо указать ранее установленный \_

 пароль)

 ThisWorkbook.Unprotect "123"

 Cancel = True

 End If

End Sub

Листинг 2.5. Запрет печати книги

Sub Workbook\_BeforePrint(Cancel As Boolean)

 ' Установка флага в True заставляет Exсel игнорировать команду \_

 отправки книги на печать

 Cancel = True

End Sub

### Закрытие рабочей книги только при выполнении условия

Листинг 2.6. Условное закрытие книги

Sub Workbook\_BeforeClose(Cancel As Boolean)

 If Range("A1").Value <> "Можно закрывать" Then

 ' Условие закрытия не выполнено. Укажем Exсel игнорировать \_

 команду

 Cancel = True

 End If

End Sub

### Быстрое удаление из рабочей книги ненужных имен

Листинг 2.7. Удаление ненужных имен

Sub EraseNames()

 Dim nmName As Name

 Dim strMessage As String

 ' Проверка наличия в книге определенных имен

 If ThisWorkbook.Names.Count = 0 Then

 ' В книге нет определенных имен

 MsgBox "Имена не определены"

 Exit Sub

 End If

 ' Просмотр всей коллекции определенных имен и удаление тех, \_

 которые пользователю не нужны

 For Each nmName In ThisWorkbook.Names

 With nmName

 ' Спрашиваем пользователя о необходимости удалить \_

 найденное имя

 strMessage = "Удалить имя " & .Name & " ? " & vbCr & \_

 "относящееся к " & .RefersTo

 If MsgBox(strMessage, vbYesNo + vbQuestion) = vbYes Then

 ' Имя можно удалить

 .Delete

 End If

 End With

 Next

End Sub

### Сортировка листов в текущей рабочей книге

Листинг 2.8. Сортировка листов

Sub SortSheets()

 Dim astrSheetNames() As String ' Массив для хранения имен листов

 Dim intSheetCount As Integer

 Dim i As Integer

 Dim objActiveSheet As Object

 ' Если нет активной рабочей книги - закрыть процедуру

 If ActiveWorkbook Is Nothing Then Exit Sub

 ' Проверка защищенности структуры рабочей книги

 If ActiveWorkbook.ProtectStructure Then

 ' Сортировка листов защищенной рабочей книги невозможна

 MsgBox "Структура книги " & ActiveWorkbook.Name & \_

 " защищена. Сортировка листов невозможна.", \_

 vbCritical

 Exit Sub

 End If

 ' Сохраняем ссылку на активный лист книги

 Set objActiveSheet = ActiveSheet

 ' Отключение сочетания клавиш Ctrl+Pause Break

 Application.EnableCancelKey = xlDisabled

 ' Отключение обновления экрана

 Application.ScreenUpdating = False

 intSheetCount = ActiveWorkbook.Sheets.Count

 ' Заполнение массива astrSheetNames именами листов книги

 ReDim astrSheetNames(1 To intSheetCount)

 For i = 1 To intSheetCount

 astrSheetNames(i) = ActiveWorkbook.Sheets(i).Name

 Next i

 ' Сортировка массива имен в порядке возрастания

 Call Sort(astrSheetNames)

 ' Перемещение листов книги

 For i = 1 To intSheetCount

 ActiveWorkbook.Sheets(astrSheetNames(i)).Move \_

 ActiveWorkbook.Sheets(i)

 Next i

 ' Переход на исходный рабочий лист

 objActiveSheet.Activate

 ' Включение обновления экрана

 Application.ScreenUpdating = True

 ' Включение сочетания клавиш Ctrl+Pause Break

 Application.EnableCancelKey = xlInterrupt

End Sub

Sub Sort(astrNames() As String)

 ' Сортировка массива строк по алфавиту (в порядке возрастания)

 Dim i As Integer, j As Integer

 Dim strBuffer As String

 Dim fBuffer As Boolean

 For i = LBound(astrNames) To UBound(astrNames) - 1

 For j = i + 1 To UBound(astrNames)

 If astrNames(i) > astrNames(j) Then

 ' Меняем i-й и j-й элементы массива местами

 strBuffer = astrNames(i)

 astrNames(i) = astrNames(j)

 astrNames(j) = strBuffer

 End If

 Next j

 Next i

End Sub

Листинг 2.9. Список отсортированных листов

Sub SortSheets2()

 Dim astrSheetNames() As String ' Массив для хранения имен листов

 Dim intSheetCount As Integer

 Dim i As Integer

 Dim objActiveSheet As Object

 ' Если нет активной рабочей книги - закрыть процедуру

 If ActiveWorkbook Is Nothing Then Exit Sub

 ' Проверка защищенности структуры рабочей книги

 If ActiveWorkbook.ProtectStructure Then

 ' Сортировка листов защищенной рабочей книги невозможна

 MsgBox "Структура книги " & ActiveWorkbook.Name & \_

 " защищена. Сортировка листов невозможна.", \_

 vbCritical

 Exit Sub

 End If

 ' Сохраняем ссылку на активный лист книги

 Set objActiveSheet = ActiveSheet

 ' Отключение сочетания клавиш Ctrl+Pause Break

 Application.EnableCancelKey = xlDisabled

 ' Функция обновления экрана отключается

 Application.ScreenUpdating = False

 With ActiveWorkbook

 ' Cоздаем новый лист "Сортировка" (если он еще не создан)

 On Error Resume Next

 If .Sheets("Сортировка") Is Nothing Then

 .Sheets.Add.Name = "Сортировка"

 End If

 On Error GoTo 0

 ' Размещение данных на листе "Сортировка" (в столбец "A")

 intSheetCount = .Sheets.Count

 For i = 1 To intSheetCount

 .Sheets("Сортировка").Cells(i, 1) = .Sheets(i).Name

 Next i

 ' Сортировка данных в ячейках листа "Сортировка" по содержимому \_

 столбца A

 .Sheets("Сортировка").Range("A1").Sort \_

 Key1:=.Sheets("Сортировка").Range("A1"), \_

 Order1:=xlAscending

 ' Заполнение массива имен отсортированными строками

 ReDim astrSheetNames(1 To intSheetCount)

 For i = 1 To intSheetCount

 astrSheetNames(i) = .Sheets("Сортировка").Cells(i, 1)

 Next i

 ' Перемещение листов

 For i = 1 To intSheetCount

 .Sheets(astrSheetNames(i)).Move .Sheets(i)

 Next i

 End With

 ' Переход на исходный рабочий лист

 objActiveSheet.Activate

 ' Включаем обновление экрана

 Application.ScreenUpdating = True

 ' Включение сочетания клавиш Ctrl+Pause Break

 Application.EnableCancelKey = xlInterrupt

End Sub

### Импорт данных, для которых нужно более 256 столбцов

Листинг 2.10. Импорт большого количества данных

Sub ImportWideSheet()

 Dim rgRange As Range ' Хранит заполняемую ячейку

 Dim lngRow As Long ' Хранит номер текущей строки

 Dim intCol As Integer ' Хранит номер текущего столбца

 Dim i As Integer

 Dim strLine As String ' Обрабатываемая строка (из файла)

 Dim strCurChar As String \* 1

 Dim strCellValue As String ' В этой строке формируется значение \_

 заполняемой ячейки таблицы

 Dim wshtCurrentSheet As Worksheet ' Лист, на котором находится \_

 заполняемая ячейка

 ' Отключение обновления изображения

 Application.ScreenUpdating = False

 ' Создание книги с одним листом

 Workbooks.Add xlWorksheet

 Set rgRange = ActiveWorkbook.Sheets(1).Range("A1")

 ' Чтение первой строки из файла (по этой строке определяется \_

 ширина таблицы)

 Open ThisWorkbook.Path & "\Primer.txt" For Input As #1

 Line Input #1, strLine

 ' Обработка первой строки с добавлением новых листов по мере \_

 необходимости

 For i = 1 To Len(strLine)

 strCurChar = Mid(strLine, i, 1)

 ' Проверка - закончились столбцы или нет

 If intCol <> 0 And intCol Mod 256 = 0 Then

 ' Столбцы текущего листа закончились - добавим новый лист \_

 и перейдем к его первому столбцу

 Set wshtCurrentSheet = ActiveWorkbook.Sheets.Add(, \_

 ActiveWorkbook.Sheets(ActiveWorkbook.Sheets.Count))

 Set rgRange = wshtCurrentSheet.Range("A1")

 intCol = 0

 End If

 ' Проверка - закончилось поле или нет

 If strCurChar = "," Then

 ' Запишем данные в таблицу

 rgRange.Offset(lngRow, intCol) = strCellValue

 intCol = intCol + 1

 strCellValue = ""

 Else

 ' Добавляем очередной символ в строку со значением текущей \_

 ячейки

 strCellValue = strCellValue & Mid(strLine, i, 1)

 ' Проверка - конец строки или нет

 If i = Len(strLine) Then

 ' Дошли до конца строки - запишем значение последней ячейки

 rgRange.Offset(lngRow, intCol) = strCellValue

 intCol = 0

 strCellValue = ""

 End If

 End If

 Next i

 ' Чтение остальных строк файла

 Do Until EOF(1)

 Set rgRange = ActiveWorkbook.Sheets(1).Range("A1")

 lngRow = lngRow + 1

 intCol = 0

 Line Input #1, strLine

 ' Обработка считанной строки

 For i = 1 To Len(strLine)

 strCurChar = Mid(strLine, i, 1)

 ' Проверка - закончились столбцы или нет

 If intCol <> 0 And intCol Mod 256 = 0 Then

 ' Столбцы текущего листа закончились - добавим новый лист \_

 и перейдем к его первому столбцу

 Set wshtCurrentSheet = ActiveWorkbook.Sheets.Add(, \_

 ActiveWorkbook.Sheets(ActiveWorkbook.Sheets.Count))

 Set rgRange = wshtCurrentSheet.Range("A1")

 intCol = 0

 End If

 ' Проверка - закончилось поле или нет

 If strCurChar = "," Then

 ' Запишем данные в таблицу

 rgRange.Offset(lngRow, intCol) = strCellValue

 intCol = intCol + 1

 strCellValue = ""

 Else

 ' Добавляем очередной символ в строку со значением текущей \_

 ячейки

 strCellValue = strCellValue & Mid(strLine, i, 1)

 ' Проверка - конец строки или нет

 If i = Len(strLine) Then

 ' Дошли до конца строки - запишем значение последней \_

 ячейки

 rgRange.Offset(lngRow, intCol) = strCellValue

 strCellValue = ""

 End If

 End If

 Next i

 Loop

 ' Не забываем закрыть входной файл

 Close #1

 ' и разрешить обновление изображения

 Application.ScreenUpdating = True

End Sub

## Рабочий лист

### Новый лист — через макрос

Листинг 2.11. Создание нового листа

Sub NewSheet()

 Worksheets.Add

End Sub

### Блокировка использования контекстного меню

Листинг 2.12. Блокировка контекстного меню

Sub Worksheet\_BeforeRightClick(ByVal Target As Range, Cancel As Boolean)

 Static intCount As Integer ' Счетчик нажатий кнопки мыши

 Dim x As Integer, y As Integer

 ' Блокировать обработку щелчка правой кнопкой мыши

 Cancel = True

 ' Отображение текстового поля с количеством щелчков правой \_

 кнопкой мыши

 x = Target.Left

 y = Target.Top

 intCount = intCount + 1

 ActiveSheet.Shapes.AddTextbox(msoTextOrientationHorizontal, \_

 x, y, 35, 20).TextFrame.Characters.Text = intCount

End Sub

### Вставка колонтитула с именем книги, листа и текущей датой

Листинг 2.13. Вставка колонтитула

Sub AddPageHeader()

 Dim i As Integer

 With ThisWorkbook

 ' Вставка колонтитулов на все листы рабочей книги

 For i = 1 To .Worksheets.Count - 1

 .Worksheets(i).PageSetup.LeftHeader = .FullName

 .Worksheets(i).PageSetup.CenterHeader = Worksheets(i).Name

 .Worksheets(i).PageSetup.RightHeader = Now()

 Next

 End With

End Sub

### Проверка существования листа

Листинг 2.14. Проверка существования листа

Function dhSheetExist(strSheetName As String) As Boolean

 Dim objSheet As Object

 On Error GoTo HandleError ' При ошибке перейти на HandleError

 ' Пытаемся получить ссылку на заданный лист

 objSheet = ActiveWorkbook.Sheets(strSheetName)

 ' Ошибки не возникло - лист существует

 dhSheetExist = True

 Exit Function

HandleError:

 ' При попытке получить доступ к листу с заданным именем \_

 возникла ошибка, значит, такого листа не существует

 dhSheetExist = False

End Function

### Проверка, защищен ли рабочий лист

Листинг 2.15. Проверка наличия защиты рабочего листа

Sub IsSheetProtected()

 ' Проверка, установлена ли защита на содержимое листа

 If Worksheets(1).ProtectContents Then

 MsgBox "Защита листа включена"

 Else

 MsgBox "Защита листа не включена"

 End If

End Sub

### Сколько страниц на всех листах?

Листинг 2.16. Подсчет страниц

Sub GetPrintPagesCount()

 Dim wshtSheet As Worksheet

 Dim intPagesCount As Integer

 ' Суммирование количества страниц, необходимых для печати всех \_

 листов книги

 For Each wshtSheet In Worksheets

 intPagesCount = intPagesCount + (wshtSheet.HPageBreaks.Count + 1) \* \_

 (wshtSheet.VPageBreaks.Count + 1)

 Next

 MsgBox "Всего страниц: " & intPagesCount

End Sub

### Автоматический пересчет данных таблицы при изменении ее значений

Листинг 2.17. Переформирование таблицы

Sub Worksheet\_Change(ByVal Target As Range)

 Dim rgData As Range

 Dim cell As Range

 Dim dblMax As Double, dblMin As Double, dblAverage As Double

 ' Получение контролируемого диапазона ячеек

 Set rgData = Range("B2:B11")

 ' Проверка, не входит ли измененная ячейка в контролируемый \_

 диапазон

 If Not (Application.Intersect(Target, rgData) Is Nothing) Then

 If Application.WorksheetFunction.CountA(rgData) > 0 Then

 ' Изменена ячейка из контролируемого диапазона

 ' Заново рассчитываем минимальное, максимальное и среднее \_

 значения в контролируемом диапазоне ячеек

 dblMin = Application.WorksheetFunction.Min(rgData)

 dblMax = Application.WorksheetFunction.Max(rgData)

 dblAverage = Application.WorksheetFunction.Average(rgData)

 ' Проверяем каждую ячейку из контролируемого диапазона \_

 и изменяем цвет шрифта ячеек с минимальным и максимальным \_

 значениями, а также помечаем желтым цветом ячейки \_

 со значениями больше среднего

 For Each cell In rgData

 If cell.Value = dblMax Then

 ' Ячейку с максимальным значением выделим красным цветом

 cell.Font.Bold = True

 cell.Font.Color = RGB(255, 0, 0)

 ElseIf cell.Value = dblMin Then

 ' Ячейку с минимальным значением выделим синим цветом

 cell.Font.Bold = False

 cell.Font.Color = RGB(0, 0, 255)

 Else

 cell.Font.Bold = False

 cell.Font.Color = RGB(0, 0, 0)

 End If

 If cell.Value > dblAverage Then

 ' Значение в ячейке больше среднего - выделим ее \_

 желтым цветом

 cell.Interior.Color = RGB(255, 255, 0)

 Else

 cell.Interior.ColorIndex = xlNone

 End If

 Next

 Else

 rgData.Interior.ColorIndex = xlNone

 End If

 End If

End Sub

## Ячейка и диапазон

### Автоматизация ввода данных в ячейки

Листинг 2.18. Ввод данных в ячейки

Sub SetCellData()

 ' Заполнение значениями ячеек А3 и В4

 Range("A3") = "Данные для ячейки A3"

 Range("B4") = "Данные для ячейки B4"

End Sub

Листинг 2.19. Ввод данных с использованием формул

Sub SetCellFormula()

 ' Запись в ячейку А6 формулы "=A5+B5"

 Range("A6") = "=A5+B5"

End Sub

### Выделение диапазона над текущей ячейкой

Листинг 2.20. Выделение диапазона над текущей ячейкой

Sub SelectCellRange()

 Dim strSelTop As String, strSelBottom As String

 ' Получение адресов нижней и верхней ячеек диапазона для выделения

 strSelBottom = ActiveCell.Address

 strSelTop = Cells(1, ActiveCell.Column).Address

 ' Выделяем все ячейки выше текущей (вместе с текущей ячейкой)

 Range(strSelTop & ":" & strSelBottom).Select

End Sub

### Поиск ближайшей пустой ячейки столбца

Листинг 2.21. Поиск ближайшей пустой ячейки столбца

Sub FindEmptyCell()

 ' Поиск ближайшей пустой ячейки в текущем столбце

 Do While Not IsEmpty(ActiveCell.Value)

 ActiveCell.Offset(1, 0).Select

 Loop

End Sub

### Поиск максимального значения в диапазоне

Листинг 2.22. Поиск максимального значения

Sub FindMaxValue()

 On Error Goto NoCell

 If Selection.Count > 1 Then

 ' Поиск максимального значения в выделенных ячейках

 Selection.Find(Application.Max(Selection)).Select

 Else

 ' Поиск максимального значения во всех ячейках листа

 ActiveSheet.Cells.Find(Application.Max(ActiveSheet.Cells)).Select

 End If

 Exit Sub

NoCell:

 MsgBox "Максимальное значение не найдено"

End Sub

### Автоматическая замена значений диапазона

Листинг 2.23. Автоматическая замена значений

Sub ReplaceValues()

 Dim cell As Range

 ' Проверка каждой ячейки диапазона на возможность замены \_

 значения в ней (отрицательные значения заменяются на -1, \_

 положительные - на 1)

 For Each cell In Range("C1:C3").Cells

 If cell.Value < 0 Then

 cell.Value = -1

 ElseIf cell.Value > 0 Then

 cell.Value = 1

 End If

 Next

End Sub

### Быстрое заполнение диапазона

#### Первый способ

Листинг 2.24. Быстрое заполнение диапазона

Sub FillCells()

 Dim intStartVal As Integer ' Начальное значение

 Dim intStep As Integer ' Шаг при изменении значения

 Dim intEndVal As Integer ' Конечное значение

 Dim intVal As Integer ' Текущее значение

 Dim intCellOffset As Integer ' Смещение от начальной ячейки

 ' Установка параметров заполнения

 intStartVal = 1

 intStep = 1

 intEndVal = 100

 ' Заполнение ячеек текущего столбца значениями от 1 до 100

 For intVal = intStartVal To intEndVal Step intStep

 ActiveCell.Offset(intCellOffset, 0).Value = intVal

 intCellOffset = intCellOffset + 1

 Next intVal

End Sub

Листинг 2.25. Заполнение через интервал

Sub FillCells()

 Dim intStartVal As Integer ' Начальное значение

 Dim intStep As Integer ' Шаг при изменении значения

 Dim intEndVal As Integer ' Конечное значение

 Dim intVal As Integer ' Текущее значение

 Dim intCellOffset As Integer ' Смещение от начальной ячейки

 Dim intCellStep As Integer ' Шаг при перемещении между \_

 заполняемыми ячейками

 ' Установка параметров заполнения

 intStartVal = 3

 intStep = 3

 intEndVal = 30

 intCellStep = 3

 ' Заполнение ячеек текущего столбца значениями от 3 до 30

 For intVal = intStartVal To intEndVal Step intStep

 ActiveCell.Offset(intCellOffset, 0).Value = intVal

 intCellOffset = intCellOffset + intCellStep

 Next intVal

End Sub

#### Второй способ

Листинг 2.26. Заполнение указанного диапазона

Sub FillCellRect()

 Dim lngRows As Long, intCols As Integer ' Количество ячеек по \_

 горизонтали и вертикали

 Dim lngRow As Long, intCol As Integer ' Координаты текущей ячейки

 Dim lngStep As Long, lngVal As Long

 ' Установка начального значения и шага заполнения

 lngVal = 1

 lngStep = 1

 ' Ввод количества ячеек по горизонтали и вертикали, которое \_

 необходимо заполнить

 lngRows = Val(InputBox("Количество ячеек в высоту"))

 intCols = Val(InputBox("Количество ячеек в ширину"))

 ' Отключение обновления экрана

 Application.ScreenUpdating = False

 ' Заполнение ячеек значениями

 For lngRow = 1 To lngRows

 For intCol = 1 To intCols

 ActiveCell.Offset(lngRow, intCol).Value = lngVal

 lngVal = lngVal + lngStep

 Next intCol

 Next lngRow

 ' Включение обновления экрана

 Application.ScreenUpdating = True

End Sub

#### Третий способ

Листинг 2.27. Заполнение диапазона

Sub FillCellRect1()

 Dim lngRows As Long, intCols As Integer

 Dim lngRow As Long, intCol As Integer

 Dim lngStep As Long, lngVal As Long

 Dim alngValues() As Long

 Dim rgRange As Range

 ' Установка начального значения и шага заполнения

 lngVal = 1

 lngStep = 1

 ' Ввод количества ячеек по горизонтали и вертикали, которое \_

 необходимо заполнить

 lngRows = Val(InputBox("Количество ячеек в высоту"))

 intCols = Val(InputBox("Количество ячеек в ширину"))

 ReDim alngValues(1 To lngRows, 1 To intCols)

 Set rgRange = ActiveCell.Range(Cells(1, 1), \_

 Cells(lngRows, intCols))

 ' Заполнение массива alngValues значениями

 For lngRow = 1 To lngRows

 For intCol = 1 To intCols

 alngValues(lngRow, intCol) = lngVal

 lngVal = lngVal + lngStep

 Next intCol

 Next lngRow

 ' Перенос значений из массива в таблицу

 rgRange.Value = alngValues

End Sub

### Помещение в ячейку электронных часов

Листинг 2.28. Размещение в ячейке электронных часов

Sub UpdateTime()

 Dim varNextCall As Variant

 ' Записываем в ячейку текущее время

 Cells(1, 1).Value = Now

 ' Записываем в varNextCall время, когда вызвать этот макрос \_

 в следующий раз (через 1 секунду)

 varNextCall = TimeSerial(Hour(Now), Minute(Now), Second(Now) + 1)

 ' Уведомляем Excel в необходимости вызова макроса

 Application.OnTime varNextCall, "UpdateTime"

End Sub

### «Будильник»

Листинг 2.29. «Будильник»

Sub Clock()

 ' Уведомляем Excel, что процедуру Alarm нужно вызвать в 20:55

 Application.OnTime TimeValue("20:55:00"), "Alarm"

End Sub

Sub Alarm()

 MsgBox "Пора ужинать!!!"

End Sub

### Поиск данных в диапазоне

#### Поиск в диапазоне значения по шаблону

Листинг 2.30. Поиск и замена по шаблону

Sub ReplaceCellsData()

 Dim cell As Range

 ' Просмотр всех ячеек диапазона G1:K20 и замена искомого текста

 For Each cell In [G1:K20]

 If cell.Value Like "\*Доход\*" Then

 cell.Value = "Выручка"

 cell.Interior.Color = RGB(255, 255, 0)

 Else

 cell.Interior.Color = RGB(255, 255, 255)

 End If

 Next

End Sub

#### Поиск значения с выводом результата

Листинг 2.31. Поиск значения с отображением результата в отдельном окне

Sub Search()

 Dim rgResult As Range

 ' Поиск заданного значения в диапазоне B1:B20 и вывод результата

 Set rgResult = Range("B1:B20").Find(9999, , xlValues)

 If rgResult Is Nothing Then

 MsgBox "Поиск не дал результатов"

 Else

 MsgBox rgResult.Address

 End If

End Sub

#### Поиск с выделением найденных данных

Листинг 2.32. Выделение найденных данных

Sub FindAndSelect()

 Dim strStartAddr As String ' Хранит координаты первого найденного \_

 значения

 Dim rgResult As Range

 ' Поиск первого входжения искомого слова

 Set rgResult = Range("B1:B10").Find("Прибыль", , xlValues)

 If Not rgResult Is Nothing Then

 ' Сохраним адрес найденной ячейки (чтобы контролировать \_

 зацикливание поиска)

 strStartAddr = rgResult.Address

 End If

 Do While Not rgResult Is Nothing

 ' Обработка результата поиска

 rgResult.Interior.Color = RGB(255, 255, 0)

 ' Новый поиск

 Set rgResult = Range("B1:B10").FindNext(rgResult)

 If rgResult.Address = strStartAddr Then

 ' Поиск завершен

 Exit Do

 End If

 Loop

End Sub

### Создание цветной границы диапазона

Листинг 2.33. Оформление верхней и нижней границ диапазона

Sub RangeBorder()

 Dim rgRange As Range

 Set rgRange = Range("B2:D5")

 ' Оформление верхней границы диапазона

 With rgRange.Borders(xlEdgeTop)

 .Weight = xlThick

 .LineStyle = xlContinuous

 .Color = RGB(0, 0, 255)

 End With

 ' Оформление нижней границы диапазона

 With rgRange.Borders(xlEdgeBottom)

 .Weight = xlMedium

 .LineStyle = xlDash

 .Color = RGB(255, 0, 255)

 End With

End Sub

### Автоматическое определение адреса ячейки

Листинг 2.34. Информация об адресе активной ячейки

Sub Worksheet\_SelectionChange(ByVal Target As Range)

 ' Вывод адреса ячейки в различных форматах

 MsgBox Target.Address() & vbCr & \_

 Target.Address(RowAbsolute:=False) & vbCr & \_

 Target.Address(ReferenceStyle:=xlR1C1) & vbCr & \_

 Target.Address(ReferenceStyle:=xlR1C1, \_

 RowAbsolute:=False, ColumnAbsolute:=False, \_

 RelativeTo:=Worksheets(1).Cells(2, 2))

End Sub

### Автоматизация добавления примечаний в указанном диапазоне

Листинг 2.35. Добавление примечаний в диапазон

Sub CreateComments()

 Dim cell As Range

 ' Производим поиск по всем ячейкам диапазона и добавляем примечания \_

 ко всем ячейкам, содержащим слово "Выручка"

 For Each cell In Range("B1:B100")

 If cell.Value Like "\*Выручка\*" Then

 cell.ClearComments

 cell.AddComment "Неучтенная наличка"

 End If

 Next

End Sub

### Заливка диапазона

Листинг 2.36. Создание заливки диапазона

Sub FillRange()

 ' Заливка диапазона

 With Range("B1:E10")

 ' Задаем узор - сетчатый

 .Interior.Pattern = xlPatternChecker

 ' Цвет узора - синий

 .Interior.PatternColor = RGB(0, 0, 255)

 ' Цвет ячейки - красный

 .Interior.Color = RGB(255, 0, 0)

 End With

End Sub

### Ввод строго ограниченных значений в указанный диапазон

#### Ввод данных с помощью диалогового окна

Листинг 2.37. Настройка ввода данных в диалоговом окне

Sub DialogInputData()

 Dim intMin As Integer, intMax As Integer ' Диапазон значений

 Dim strInput As String ' Введенная пользователем строка

 Dim strMessage As String

 Dim intValue As Integer

 intMin = 1 ' Минимальное значение

 intMax = 50 ' Максимальное значение

 strMessage = "Введите значение от " & intMin & " до " & intMax

 ' Ввод значения (цикл завершается, когда пользователь вводит \_

 значение из заданного диапазона или отменяет ввод)

 Do

 strInput = InputBox(strMessage)

 If strInput = "" Then Exit Sub ' Отмена ввода

 ' Проверка, содержит ли введенная пользователем строка число

 If IsNumeric(strInput) Then

 intValue = CInt(strInput)

 ' Проверка, удовлетворяет ли значение диапазону

 If intValue >= intMin And intValue <= intMax Then

 ' Все условия выполнены

 Exit Do

 End If

 End If

 ' Формирование сообщения с текстом ошибки

 strMessage = "Вы ввели некорректное значение." & vbNewLine & \_

 "Введите число от " & intMin & " до " & intMax

 Loop

 ' Внесение данных в ячейку

 ActiveSheet.Range("A1").Value = strInput

End Sub

#### Непосредственный ввод данных

Листинг 2.38. Ограничение возможных значений диапазона

Sub Worksheet\_Change(ByVal Target As Excel.Range)

 Dim rgInputRange As Range

 Dim cell As Range

 Dim strMessage As String

 Dim varResult As Variant

 ' Диапазон, в котором контролируется ввод

 Set rgInputRange = Range("A1:E10")

 ' Просмотр всех измененных ячеек и контроль ввода в тех, которые \_

 принадлежат заданному диапазону

 For Each cell In Target

 ' Проверка принадлежности диапазону

 If Union(cell, rgInputRange).Address = rgInputRange.Address Then

 ' Контроль правильности ввода

 varResult = IsCellDataValid(cell)

 If varResult = True Then

 ' Введено корректное значение

 Exit Sub

 Else

 ' Формирование и вывод сообщения об ошибке

 strMessage = "Ячейка " & cell.Address(False, False) & ":" \_

 & vbCrLf & vbCrLf & varResult

 MsgBox strMessage, vbCritical, "Неправильное значение"

 ' Очистка ввода

 Application.EnableEvents = False

 cell.ClearContents

 cell.Activate

 Application.EnableEvents = True

 End If

 End If

 Next cell

End Sub

Function IsCellDataValid(cell As Range) As Variant

 ' Возвращает True, если в ячейку вводится целое число \_

 в диапазоне от 1 до 12. В противном случае выдается \_

 соответствующее сообщение

 ' Проверка, является ли содержимое ячейки числом

 If Not WorksheetFunction.IsNumber(cell.Value) Then

 IsCellDataValid = "Нечисловое значение"

 Exit Function

 End If

 ' Проверка, является ли введенное число целым

 If Int(cell.Value) <> cell.Value Then

 IsCellDataValid = "Введите целое число"

 Exit Function

 End If

 ' Проверка соответствия числа диапазону

 If cell.Value < 1 Or cell.Value > 12 Then

 IsCellDataValid = "Значение должно быть от 1 до 12"

 Exit Function

 End If

 ' В ячейку введено допустимое значение

 IsCellDataValid = True

End Function

### Последовательный ввод данных

Листинг 2.39. Последовательный ввод данных

Sub StreamInput()

 Dim strDate As String

 Dim strSum As String

 Dim lngRow As Long

 ' Ввод данных в цикле (повторяется до тех пор, пока пользователь \_

 не введет пустую строку или не нажмет "Отмена" в окне ввода)

 Do

 lngRow = Range("A65536").End(xlUp).Row + 1

 ' Ввод даты

 strDate = InputBox("Вводим дату")

 If strDate = "" Then Exit Sub

 ' Ввод выручки

 strSum = InputBox("Вводим выручку")

 If strSum = "" Then Exit Sub

 ' Запись данных в ячейки

 Cells(lngRow, 1) = strDate

 Cells(lngRow, 2) = strSum

 Loop

End Sub

### Быстрое выделение ячеек с отрицательными значениями

Листинг 2.40. Выделение отрицательных значений

Sub NegSelect()

 Dim cell As Range

 ' Просмотр всех ячеек выделенного диапазона и пометка тех, \_

 которые содержат отрицательные значения

 For Each cell In Selection

 If cell.Value < 0 Then

 cell.Interior.Color = RGB(255, 0, 0)

 Else

 cell.Interior.ColorIndex = xlNone

 End If

 Next cell

End Sub

### Получение информации о выделенном диапазоне

Листинг 2.41. Получение информации о выделенном диапазоне

Sub TypeOfSelection()

 Dim rgSelUnion As Range ' Объединение выделенных областей

 Dim strTitle As String ' Заголовок сообщения

 Dim strMessage As String ' Текст сообщения

 Dim strSelType As String ' Тип выделения (простой или \_

 множественный)

 Dim intBlockCount As Integer ' Количество блоков в выделении

 Dim intCellCount As Long ' Общее количество выделенных ячеек

 Dim intColCount As Integer ' Количество выделенных столбцов

 Dim intRowCount As Long ' Количество выделенных строк

 Dim intAreasCount As Integer ' Количество выделенных областей

 Dim strCurSelType As String

 Dim rgArea As Range

 ' Подсчет количества выделенных областей и определение типа выделения: \_

 простое (одна область) или сложное(несколько областей)

 intAreasCount = Selection.Areas.Count

 If intAreasCount = 1 Then

 strTitle = "Простое выделение"

 Else

 strTitle = "Множественное выделение"

 End If

 ' Определение типа выделения первой области

 strSelType = dhGetAreaType(Selection.Areas(1))

 ' Создание объединения во избежание повторного учета \_

 пересекающихся участков выделенных диапазонов

 Set rgSelUnion = Selection.Areas(1)

 For Each rgArea In Selection.Areas

 strCurSelType = dhGetAreaType(rgArea)

 ' Изменение надписи о типе всего выделения, если \_

 есть выделения различного типа

 If strCurSelType <> strSelType Then

 strSelType = "Множественный"

 End If

 ' Определение количества блоков перед их добавлением в объединение

 If strCurSelType = "Block" Then

 intBlockCount = intBlockCount + 1

 End If

 ' Добавление в объединение

 Set rgSelUnion = Union(rgSelUnion, rgArea)

 Next rgArea

 ' Просматриваются элементы созданного объединения

 For Each rgArea In rgSelUnion.Areas

 Select Case dhGetAreaType(rgArea)

 Case "Строка"

 intRowCount = intRowCount + rgArea.Rows.Count

 Case "Столбец"

 intColCount = intColCount + rgArea.Columns.Count

 Case "Лист"

 intColCount = intColCount + rgArea.Columns.Count

 intRowCount = intRowCount + rgArea.Rows.Count

 End Select

 Next rgArea

 ' Определение количества неперекрывающихся ячеек

 intCellCount = rgSelUnion.Count

 ' Формирование и вывод итогового сообщения

 strMessage = "Тип выделения:" & vbTab & strSelType & vbCrLf & \_

 "Количество областей: " & vbTab & intAreasCount & vbCrLf & \_

 "Полных столбцов: " & vbTab & intColCount & vbCrLf & \_

 "Полных строк: " & vbTab & intRowCount & vbCrLf & \_

 "Блоков ячеек: " & vbTab & intBlockCount & vbCrLf & \_

 "Всего ячеек: " & vbTab & Format(intCellCount, "#,###")

 MsgBox strMessage, vbInformation, strTitle

End Sub

Function dhGetAreaType(rgRangeArea As Range) As String

 ' Определение типа диапазона

 If rgRangeArea.Count = Cells.Count Then

 ' Все ячейки рабочего листа

 dhGetAreaType = "Лист"

 ElseIf rgRangeArea.Cells.Count = 1 Then

 ' Одна ячейка

 dhGetAreaType = "Ячейка"

 ElseIf rgRangeArea.Rows.Count = Cells.Rows.Count Then

 ' Весь столбец

 dhGetAreaType = "Столбец"

 ElseIf rgRangeArea.Columns.Count = Cells.Columns.Count Then

 ' Вся строка

 dhGetAreaType = "Строка"

 Else

 ' Блок ячеек

 dhGetAreaType = "Блок"

 End If

End Function

### Панель инструментов изменения числового формата ячейки

Листинг 2.42. Код в модуле рабочего листа

Sub Worksheet\_Change(ByVal Target As Excel.Range)

 Call UpdateToolbar

End Sub

Sub Worksheet\_SelectionChange(ByVal Target As Excel.Range)

 Call UpdateToolbar

End Sub

Листинг 2.43. Код в стандартном модуле

Sub FastChangeNumberFormat()

 Dim bar As CommandBar

 Dim button As CommandBarButton

 ' Удаление существующей панели инструментов (если она есть)

 On Error Resume Next

 CommandBars("Числовой формат").Delete

 On Error GoTo 0

 ' Формирование новой панели

 Set bar = CommandBars.Add

 With bar

 .Name = "Числовой формат"

 .Visible = True

 End With

 ' Создание кнопки

 Set button = CommandBars("Числовой формат").Controls.Add \_

 (Type:=msoControlButton)

 With button

 .Caption = ""

 .OnAction = "ChangeNumFormat"

 .TooltipText = "Щелкните для изменения числового формата"

 .Style = msoButtonCaption

 End With

 ' Обновление созданной панели инструментов

 Call UpdateToolbar

End Sub

Sub UpdateToolbar()

 ' Обновление панели инструментов (если она создана)

 On Error Resume Next

 ' Изменение заголовка кнопки (на название формата выделенной ячейки)

 CommandBars("Числовой формат").Controls(1).Caption = \_

 ActiveCell.NumberFormat

End Sub

Sub ChangeNumFormat()

 ' Отображение диалогового окна изменения формата ячейки

 Application.Dialogs(xlDialogFormatNumber).Show

 Call UpdateToolbar

End Sub

### Тестирование скорости чтения и записи диапазонов

Листинг 2.44. Тестирование скорости чтения и записи диапазонов

Sub TableSpeedTest()

 Dim alngData() As Long ' Массив с числами

 Dim lngCount As Long ' Количество элементов в массиве

 Dim dtStart As Date ' Хранит время (и даже дату) начала \_

 тестирования

 Dim strArrayToTable As String ' Время записи в таблицу

 Dim strTableToArray As String ' Время чтения из таблицы

 Dim strMessage As String

 Dim i As Long

 ' Подготовка диапазона ячеек

 Range("A:A").ClearContents

 ' Ввод размера массива, формирование массива заданного размера

 lngCount = InputBox("Введите количество элементов")

 ReDim alngData(1 To lngCount)

 ' Заполнение массива данными

 For i = 1 To lngCount

 alngData(i) = i

 Next i

 ' Перенос массива в таблицу

 Application.ScreenUpdating = False

 dtStart = Timer

 For i = 1 To lngCount

 Cells(i, 1) = i

 Next i

 strArrayToTable = Format(Timer - dtStart, "00:00")

 ' Чтение данных из таблицы обратно в массив

 dtStart = Timer

 For i = 1 To lngCount

 alngData(i) = Cells(i, 1)

 Next i

 strTableToArray = Format(Timer - dtStart, "00:00")

 Application.ScreenUpdating = True

 ' Вывод на экран результатов тестирования

 strMessage = "Запись: " & strArrayToTable & vbCrLf & \_

 "Чтение: " & strTableToArray

 MsgBox strMessage, , lngCount & " элементов"

End Sub

## Разработка и применение полезных пользовательских функций

### Объединение данных диапазона

Листинг 2.45. Функция Couple

Function Couple(Diapazon)

 ' Объединение данных, содержащихся в ячейках диапазона \_

 Diapazon (разделитель между значениями - пробел)

 ' iCell - текущая ячейка

 For Each iCell In Diapazon

 ' Сцепляются данные только заполненных ячеек

 If IsEmpty(iCell) <> True Then

 ' Добавление значения ячейки в выходную строку

 If Couple = "" Then

 Couple = iCell

 Else

 Couple = Couple & " " & iCell

 End If

 End If

 Next

End Function

### Объединение данных с учетом форматов

Листинг 2.46. Функция CoupleFormat

Function CoupleFormat(Diapazon)

 ' Объединение текстовых данных, содержащихся в ячейках \_

 диапазона Diapazon (разделитель между значениями - пробел)

 ' iCell - текущая ячейка

 For Each iCell In Diapazon

 ' Сцепляются данные только заполненных ячеек

 If IsEmpty(iCell) <> True Then

 ' Добавление текста ячейки в выходную строку

 If CoupleFormat = "" Then

 CoupleFormat = iCell.Text

 Else

 CoupleFormat = CoupleFormat & " " & iCell.Text

 End If

 End If

 Next

End Function

### Эксперименты с датой

Листинг 2.47. Функция ДатаПолная

Function ДатаПолная(Ячейка)

 ' Получение данных в заданной ячейке в формате \_

 "dd mmmm yyyy"

 Дата = Format(Ячейка, "dd mmmm yyyy")

 If IsDate(Ячейка) = True Or IsDate(Дата) = True Then

 ' Возврат строки с полной датой

 ДатаПолная = StrConv(Дата, vbProperCase)

 Else

 ' Данные в ячейке не являются датой

 ДатаПолная = "<>"

 End If

End Function

### Выбор из текста всех чисел

Листинг 2.48. Функция ExtractNumeric

Function ExtractNumeric(iCell)

 ' Анализируется каждый символ входной строки iCell

 For iCount = 1 To Len(iCell)

 ' Проверка, является ли анализируемый символ числом

 If IsNumeric(Mid(iCell, iCount, 1)) = True Then

 ' Число добавляется в выходную строку

 ExtractNumeric = ExtractNumeric & Mid(iCell, iCount, 1)

 End If

 Next

End Function

### Прописная буква только в начале текста

Листинг 2.49. Функция ПрописнНач

Function ПрописнНач(Текст)

 ' Пустой текст функция не обрабатывает

 If Текст = "" Then ПрописнНач = "<>": Exit Function

 ' Выделение первого символа и перевод его в верхний регистр

 ПервыйСимвол = UCase(Left(Текст, 1))

 ' Выделение остальной части строки и перевод \_

 ее в нижний регистр

 Обрубок = LCase(Mid(Текст, 2))

 ' Соединение частей строки и возврат значения

 ПрописнНач = ПервыйСимвол & Обрубок

End Function

### Перевод чисел в «деньги»

Листинг 2.50. Функция RubKop

Function RubKop(Число)

 ' Пустые ячейки и ячейки, содержащие текст, функция \_

 не обрабатывает

 If IsNumeric(Число) = False Or Число = "" Then RubKop = \_

 "<>": Exit Function

 ' Из числа целой части - рубли

 ДлинаЧисла = Len(Число)

 ЦелаяЧасть = Fix(Число)

 ДлинаЦелой = Len(ЦелаяЧасть)

 ' Вычисление длины дробной части

 ДлинаДроби = ДлинаЧисла - ДлинаЦелой

 If ДлинаДроби <> 0 Then

 ДлинаДроби = ДлинаЧисла - ДлинаЦелой - 1

 End If

 ' Формирование количества копеек в зависимости от длины \_

 дробной части

 If ДлинаДроби = 0 Then

 ' Ноль копеек

 Копейки = "00"

 ElseIf ДлинаДроби = 1 Then

 ' Дробная часть состоит из одного числа - это \_

 десятки копеек

 Копейки = Right(Число, ДлинаДроби) & "0"

 ElseIf ДлинаДроби = 2 Then

 ' Дробная часть полностью соответствует количеству копеек

 Копейки = Right(Число, ДлинаДроби)

 Else

 ' Длина дробной части больше двух - округлим \_

 дробную часть

 Копейки = Right(Число, ДлинаДроби)

 If Mid(Копейки, 3, 1) > 4 Then

 Копейки = Left(Копейки, 2) + 1

 Else

 Копейки = Left(Копейки, 2)

 End If

 End If

 ' Составление полной надписи из количества рублей и копеек

 Рубли = ЦелаяЧасть

 RubKop = Рубли & " " & "руб." & " " & Копейки & " " & "коп."

End Function

### Подсчет количества повторов искомого текста

Листинг 2.51. Функция CoincideCount

Function CoincideCount(Text, Search)

 ' Проверка правильности входных данных \_

 (аргумента Search)

 If IsArray(Search) = True Then Exit Function

 If IsError(Search) = True Then Exit Function

 If IsEmpty(Search) = True Then Exit Function

 ' Просмотр заданного в параметре Text диапазона

 For Each iCell In Text

 ' Анализируются только ячейки, содержащие \_

 корректные значения

 If Not IsError(iCell) Then

 ' iText - строка для просмотра (в нижнем регистре)

 iText = LCase(iCell)

 ' iSearch - искомое значение (в нижнем регистре)

 iSearch = LCase(Search)

 ' Длина искомой строки

 iLen = Len(Search)

 ' Первый поиск строки iSearch в строке iText \_

 (этот и последующий поиски производятся без \_

 учета регистра символов)

 iNumber = InStr(iText, iSearch)

 While iNumber > 0

 ' Поиск следующего вхождения строки

 iNumber = InStr(iNumber + iLen, iText, iSearch)

 ' Подсчет количества вхождений

 CoincideCount = CoincideCount + vbNull

 Wend

 End If

 Next

End Function

### Суммирование данных только видимых ячеек

Листинг 2.52. Функция СуммаВид

Function СуммаВид(Диапазон) As Double

 ' Просмотр всех ячеек заданного диапазона

 For Each Ячейка In Диапазон

 ' Анализ только видимых ячеек

 If Not Ячейка.EntireRow.Hidden And Not \_

 Ячейка.EntireColumn.Hidden Then

 ' При расчете учитываются только ячейки \_

 с численными значениями

 If IsNumeric(Ячейка) = True Then

 СуммаВид = СуммаВид + Ячейка

 End If

 End If

 Next

End Function

### При суммировании — курсор внутри диапазона

Листинг 2.53. Функция Сумма

Function Сумма(Диапазон, АдресЯчейки) As Double

 ' Просмотр всех ячеек диапазона

 For Each Ячейка In Диапазон

 ' Проверка, чтобы в суммировании не участвовала \_

 ячейка с формулой

 If АдресЯчейки.Address <> Ячейка.Address Then

 ' В суммировании участвуют только ячейки \_

 с численными значениями

 If IsNumeric(Ячейка) = True Then

 Сумма = Сумма + Ячейка

 End If

 End If

 Next

End Function

### Начисление процентов в зависимости от суммы

Листинг 2.54. Функция dhCalculatePercent (вариант 1)

Function dhCalculatePercent(lngSum As Long) As Double

 ' Процентные ставки (декларация констант)

 Const dblRate1 As Double = 0.09

 Const dblRate2 As Double = 0.11

 Const dblRate3 As Double = 0.15

 ' Граничные суммы вкладов (декларация констант)

 Const intSum1 As Long = 5000

 Const intSum2 As Long = 10000

 ' Возвращаем сумму, умноженную на соответствующую ставку

 If lngSum < intSum1 Then

 dhCalculatePercent = lngSum \* dblRate1

 ElseIf lngSum < intSum2 Then

 dhCalculatePercent = lngSum \* dblRate2

 Else

 dhCalculatePercent = lngSum \* dblRate3

 End If

End Function

Листинг 2.55. Функция dhCalculatePercent (вариант 2)

Function dhCalculatePercent(lngSum As Long) As Double

 ' Процентные ставки (декларация констант)

 Const dblRate1 As Double = 0.09

 Const dblRate2 As Double = 0.11

 Const dblRate3 As Double = 0.15

 ' Граничные суммы вкладов (декларация констант)

 Const intSum1 As Long = 5000

 Const intSum2 As Long = 10000

 ' Возвращаем сумму, умноженную на соответствующую ставку

 Select Case lngSum

 Case Is < intSum1

 dhCalculatePercent = lngSum \* dblRate1

 Case Is < intSum2

 dhCalculatePercent = lngSum \* dblRate2

 Case Else

 dhCalculatePercent = lngSum \* dblRate3

 End Select

End Function

### Еще о расчете процентов

Листинг 2.56. Функция dhCalculatePercent (вариант 3)

Function dhCalculatePercent(Sales As Long, IsTemporal As Boolean) As Double

 ' Процентные ставки (декларация констант)

 Const dblRate1 As Double = 0.09

 Const dblRate2 As Double = 0.11

 Const dblRate3 As Double = 0.15

 Const dblAdd As Double = 1.1

 ' Граничные суммы

 Const lngSum1 As Long = 5000

 Const lngSum2 As Long = 10000

 ' Расчет суммы для выплаты (как обычно)

 If Sales < lngSum1 Then

 dhCalculatePercent = Sales \* dblRate1

 ElseIf Sales < lngSum2 Then

 dhCalculatePercent = Sales \* dblRate2

 Else

 dhCalculatePercent = Sales \* dblRate3

 End If

 If IsTemporal Then

 ' Для сторонних вкладчиков - надбавка

 dhCalculatePercent = dblAdd \* dhCalculatePercent

 End If

End Function

### Сводный пример расчета комиссионного вознаграждения

Листинг 2.57. Расчет комиссионного вознаграждения

Function dhCalculateCom(dblSales As Double) As Double

 Const dblRate1 = 0.09

 Const dblRate2 = 0.11

 Const dblRate3 = 0.15

 ' Расчет комиссионных с продаж (без выслуги) в зависимости \_

 от суммы

 Select Case dblSales

 Case 0 To 4999.99: dhCalculateCom = dblSales \* dblRate1

 Case 5000 To 9999.99: dhCalculateCom = dblSales \* dblRate2

 Case Is >= 10000: dhCalculateCom = dblSales \* dblRate3

 End Select

End Function

Function dhCalculateCom2(dblSales As Double, intYears As Double) \_

 As Double

 Const dblRate1 = 0.09

 Const dblRate2 = 0.11

 Const dblRate3 = 0.15

 ' Расчет комиссионных с продаж (без учета выслуги лет) \_

 в зависимости от суммы

 Select Case dblSales

 Case 0 To 4999.99: dhCalculateCom2 = dblSales \* dblRate1

 Case 5000 To 9999.99: dhCalculateCom2 = dblSales \* dblRate2

 Case Is >= 10000: dhCalculateCom2 = dblSales \* dblRate3

 End Select

 ' Надбавка за выслугу лет

 dhCalculateCom2 = dhCalculateCom2 + \_

 (dhCalculateCom2 \* intYears / 100)

End Function

Sub ComCalculator()

 Dim strMessage As String

 Dim dblSales As Double

 Dim ан As Integer

Calc:

 ' Отображение окна для ввода данных

 dblSales = Val(InputBox("Сумма реализации:", \_

 "Расчет комиссионного вознаграждения"))

 ' Формирование сообщения (с одновременным расчетом \_

 вознаграждения)

 strMessage = "Объем продаж:" & vbTab & Format(dblSales, "$#,##0") & \_

 vbCrLf & "Сумма вознаграждения:" & vbTab & \_

 Format(dhCalculateCom(dblSales), "$#,##0") & \_

 vbCrLf & vbCrLf & "Считаем дальше?"

 ' Вывод окна с сообщением (о рассчитанной сумме и вопросом \_

 о продолжении расчетов)

 If MsgBox(strMessage, vbYesNo, \_

 "Расчет комиссионного вознаграждения") = vbYes Then

 ' Продолжение расчетов

 GoTo Calc

 End If

End Sub

### Подсчет количества ячеек, содержащих указанные значения

Листинг 2.58. Количество ячеек с определенным значением

Function dhCount(rgn As Range, LowBound As Double, \_

 UpperBound As Double) As Long

 Dim cell As Range

 Dim lngCount As Long

 ' Проходим по всем ячейкам диапазона rgn и подсчитываем значения, \_

 попадающие в интервал от LowBound до UpperBound

 For Each cell In rgn

 If cell.Value >= LowBound And cell.Value <= UpperBound Then

 ' Значение попадает в заданный интервал

 lngCount = lngCount + 1

 End If

 Next

 dhCount = lngCount

End Function

### Подсчет количества видимых ячеек в диапазоне

Листинг 2.59. Подсчет количества видимых ячеек

Function dhCountVisibleCells(rgRange As Range)

 Dim lngCount As Long

 Dim cell As Range

 ' Проходим по всему диапазону и подсчитываем непустые \_

 видимые ячейки

 For Each cell In rgRange

 ' Проверка, есть ли данные в ячейке

 If Not IsEmpty(cell) Then

 ' Проверка, видима ли ячейка

 If Not cell.EntireRow.Hidden And Not \_

 cell.EntireColumn.Hidden Then

 ' Еще одна видимая ячейка

 lngCount = lngCount + 1

 End If

 End If

 Next cell

 dhCountVisibleCells = lngCount

End Function

### Поиск ближайшего понедельника

Листинг 2.60. Ближайший день недели по отношению к дате

Function dhGetNextMonday(datDate As Date) As Date

 ' Определение даты следующего понедельника (функция Weekday \_

 возвращает номер дня недели, считая от понедельника, если \_

 в качестве второго аргумента задавать vbMonday)

 If Weekday(datDate, vbMonday) = 1 Then

 ' Заданная дата и есть понедельник

 dhGetNextMonday = datDate

 Else

 ' Расчет даты следующего понедельника

 dhGetNextMonday = datDate + 8 - Weekday(datDate, vbMonday)

 End If

End Function

### Подсчет количества полных лет

Листинг 2.61. Функция dhCalculateAge

Function dhCalculateAge(datDate As Date) As Long

 Dim lngAge As Long

 ' Находим разность между текущей датой и указанной (лет)

 lngAge = DateDiff("yyyy", datDate, Date)

 If DateSerial(Year(datDate) + lngAge, Month(datDate), \_

 Day(datDate)) > Date Then

 ' В этом году день рождения еще не наступил

 lngAge = lngAge - 1

 End If

 dhCalculateAge = lngAge

End Function

### Проверка, была ли сохранена рабочая книга

Листинг 2.62. Функция dhBookIsSaved

Function dhBookIsSaved() As Boolean

 ' Если путь файла рабочей книги не задан, то она \_

 не сохранена (ThisWorkbook.path равняется "")

 dhBookIsSaved = ThisWorkbook.path <> ""

End Function

### Расчет средневзвешенного значения

Листинг 2.63. Расчет средневзвешенного значения

Function dhAverageWithWeight(rgWeights As Range, rgValues As Range) \_

 As Double

 If (rgWeights.Count <> rgValues.Count) Then

 ' Количество весов не соответствует количеству аргументов

 dhAverageWithWeight = 0

 Exit Function

 End If

 Dim i As Integer

 Dim dblSum As Double ' Сумма значений

 Dim dblSumWeight As Double ' Взвешенная сумма значений

 ' Вычисление...

 For i = 1 To rgWeights.Count

 ' Взвешенной суммы значений

 dblSumWeight = dblSumWeight + rgWeights(i) \* rgValues(i)

 ' Суммы значений

 dblSum = dblSum + rgWeights(i)

 Next

 ' Возвращение средневзвешенного значения

 dhAverageWithWeight = dblSumWeight / dblSum

End Function

### Преобразование номера месяца в его название

Листинг 2.64. Название месяца

Function dhMonthName(intMonth As Integer) As String

 ' Возвращение имени месяца по его номеру (intMonth \_

 является номером элемента в массиве с названиями месяцев)

 dhMonthName = Choose(intMonth, "Январь", "Февраль", "Март", \_

 "Апрель", "Май", "Июнь", "Июль", "Август", "Сентябрь", \_

 "Октябрь", "Ноябрь", "Декабрь")

End Function

### Расчет суммы первых значений диапазона

Листинг 2.65. Функция dhNSum

Function dhNSum(ByVal intCount As Integer, \_

 rgValues As Range) As Double

 Dim i As Integer

 Dim dblSum As Double

 If intCount > rgValues.Count Then

 ' Задано количество элементов большее, чем есть \_

 в переданном диапазоне

 intCount = rgValues.Count

 End If

 ' Расчет суммы первых intCount элементов

 For i = 1 To intCount

 dblSum = dblSum + rgValues(i)

 Next i

 ' Возврат результата

 dhNSum = dblSum

End Function

### Поиск последней непустой ячейки диапазона

Листинг 2.66. Функция dhLastUsedCell

Function dhLastUsedCell(rgRange As Range) As Long

 Dim lngCell As Long

 ' Пойдем по диапазону с конца (тогда первая попавшаяся \_

 заполненная ячейка и будет искомой)

 For lngCell = rgRange.Count To 1 Step -1

 If Not IsEmpty(rgRange(lngCell)) Then

 ' Нашли непустую ячейку

 dhLastUsedCell = lngCell

 Exit Function

 End If

 Next lngCell

 ' Непустую ячейку не нашли

 dhLastUsedCell = 0

End Function

### Поиск последней непустой ячейки столбца

Листинг 2.67. Функция dhLastColUsedCell

Function dhLastColUsedCell(rgColumn As Range) As Variant

 ' Вывод значения последней непустой ячейки столбца

 dhLastColUsedCell = rgColumn.Parent.Cells(Rows.Count, \_

 rgColumn.Column).End(xlUp).Value

End Function

### Поиск последней непустой ячейки строки

Листинг 2.68. Функция dhLastRowUsedCell

Function dhLastRowUsedCell(rgRow As Range) As Variant

 ' Вывод значения последней непустой ячейки строки

 dhLastRowUsedCell = rgRow.Parent.Cells(rgRow.Row, 256). \_

 End(xlToLeft).Address

End Function

### Подсчет количества ячеек в диапазоне, содержащих указанные значения

Листинг 2.69. Функция dhCountSomeCells

Function dhCountSomeCells(rgRange As Range, dblMin As Double, \_

 dblMax As Double) As Long

 ' Расчет количества ячеек со значениями от dblMin до dblMax \_

 с использованием стандартной функции CountIf

 With Application.WorksheetFunction

 dhCountSomeCells = .CountIf(rgRange, ">=" & dblMin) - \_

 .CountIf(rgRange, ">" & dblMax)

 End With

End Function

### Англоязычный текст — заглавными буквами

Листинг 2.70. Английский текст — в верхнем регистре

Function dhFormatEnglish(strText As String) As String

 Dim i As Integer

 Dim strCurChar As String \* 1

 ' Анализируется каждый символ строки strText. Каждый символ \_

 латинского алфавита преобразуется в верхний регистр

 For i = 1 To Len(strText)

 strCurChar = Mid(strText, i, 1)

 ' Код латинских строчных символов лежит в пределах \_

 от 97 до 122

 If Asc(strCurChar) >= 97 And Asc(strCurChar) <= 122 Then

 ' Переводим символ в верхний регистр

 dhFormatEnglish = dhFormatEnglish & UCase(strCurChar)

 Else

 ' Просто добавляем символ в выходную строку

 dhFormatEnglish = dhFormatEnglish & strCurChar

 End If

 Next i

End Function

### Отображение текста «задом наперед»

Листинг 2.71. Преобразование текста в обратном порядке

Function dhReverseText(strText As String) As String

 Dim i As Integer

 ' Переписываем символы из входной строки в выходную \_

 в обратном порядке

 For i = Len(strText) To 1 Step -1

 dhReverseText = dhReverseText & Mid(strText, i, 1)

 Next i

End Function

Sub ReverseText()

 Dim strText As String

 ' Ввод строки посредством стандартного окна ввода

 strText = InputBox("Введите текст:")

 ' Реверсия строки и вывод результата

 MsgBox dhReverseText(strText), , strText

End Sub

### Поиск максимального значения на всех листах книги

Листинг 2.72. Поиск максимального значения

Function dhMaxInBook(cell As Range) As Double

 Dim sheet As Worksheet

 Dim dblMax As Double

 Dim dblResult As Double

 Dim fFirst As Boolean

 fFirst = True

 ' Расчет максимальных значений на всех листах рабочей книги \_

 и выбор наибольшего из них

 For Each sheet In cell.Parent.Parent.Worksheets

 ' Расчет максимального значения на листе

 dblResult = Application.WorksheetFunction.Max(sheet.UsedRange)

 If fFirst Then

 ' Найдено первое значение - его не с чем сравнивать

 dblMax = dblResult

 fFirst = False

 End If

 ' Выбираем большее из dblMax и dbmResult

 If dblResult > dblMax Then

 dblMax = dblResult

 End If

 Next sheet

 ' Возврат результата

 dhMaxInBook = dblMax

End Function

### Использование относительных ссылок

Листинг 2.73. Функция dhSheetOffset

Function dhSheetOffset(offset As Integer, cell As Range) As Variant

 ' Возврат корректного значения ячейки cell листа, смещение \_

 которого относительно текущего задано переменной offset

 dhSheetOffset = Sheets(Application.Caller.Parent.Index \_

 + offset).Range(cell.Address)

End Function

Листинг 2.74. Функция dhSheetOffset2

Function dhSheetOffset2(offset As Integer, cell As Range) As Variant

 ' Корректировка смещения (чтобы ссылка была на рабочий лист)

 Do While TypeName(Sheets(cell.Parent.Index + offset)) \_

 <> "Worksheet"

 If offset > 0 Then

 ' Пропускаем лист и проходим вперед по книге

 offset = offset + 1

 Else

 ' Пропускаем лист и проходим назад по книге

 offset = offset - 1

 End If

 Loop

 ' Возврат корректного значения ячейки cell листа, смещение \_

 которого относительно текущего задано переменной offset \_

 с пропуском листов с диаграммами

 dhSheetOffset2 = Sheets(cell.Parent.Index \_

 + offset).Range(cell.Address)

End Function

### Определение типа данных ячейки

Листинг 2.75. Тип данных, хранящихся в ячейке

Function dhCellType(rgRange As Range) As String

 ' Переходим к левой верхней ячейке, если rgRange - диапазон, \_

 а не одна ячейка

 Set rgRange = rgRange.Range("A1")

 ' Определение типа значения в ячейке

 Select Case True

 Case IsEmpty(rgRange)

 ' Ячейка пуста

 dhCellType = "Пусто"

 Case Application.IsText(rgRange)

 ' В ячейке текст

 dhCellType = "Текст"

 Case Application.IsLogical(rgRange)

 ' В ячейке логическое значение (True или False)

 dhCellType = "Булево выражение"

 Case Application.IsErr(rgRange)

 ' При вычислении значения в ячейке произошла ошибка

 dhCellType = "Ошибка"

 Case IsDate(rgRange)

 ' В ячейке дата

 dhCellType = "Дата"

 Case InStr(1, rgRange.Text, ":") <> 0

 ' В ячейке время

 dhCellType = "Время"

 Case IsNumeric(rgRange)

 ' В ячейке числовое значение

 dhCellType = "Число"

 End Select

End Function

### Выделение из текста произвольного элемента

Листинг 2.76. Выделение элемента текста

Function dhGetTextItem(ByVal strTextIn As String, intItem As \_

 Integer, strSeparator As String) As String

 Dim intStart As Integer ' Позиция начала текущего элемента

 Dim intEnd As Integer ' Позиция конца текущего элемента

 Dim i As Integer ' Номер текущего элемента

 ' Проверка корректности номера элемента

 If intItem < 1 Then Exit Function

 ' Убираются лишние пробелы, если разделитель - пробел

 If strSeparator = " " Then strTextIn = Application.Trim(strTextIn)

 ' Разделитель добавляется в конец строки

 If Right(strTextIn, Len(strTextIn)) <> strSeparator Then \_

 strTextIn = strTextIn & strSeparator

 ' Поиск всех элементов в строке до нужного

 For i = 1 To intItem

 ' Начало элемента (перемещение вперед по строке)

 intStart = intEnd + 1

 ' Конец элемента

 intEnd = InStr(intStart, strTextIn, strSeparator)

 If (intEnd = 0) Then

 ' Дошли до конца строки, но элемент не нашли

 Exit Function

 End If

 Next i

 ' Выделение текста из входной строки

 dhGetTextItem = Mid(strTextIn, intStart, intEnd - intStart)

End Function

### Генератор случайных чисел

Листинг 2.77. Функция dhGetRandomValues

Function dhGetRandomValues() As Variant

 Dim intRow As Integer ' Номер текущей строки

 Dim intCol As Integer ' Номер текущего столбца

 Dim aintOut() As Integer ' Выходной массив (двумерный)

 Dim aintValues() As Integer ' Массив с возможными значениями

 Dim intMax As Integer ' Последний доступный элемент массива \_

 aintValues

 Dim i As Integer

 ReDim aintOut(1 To Application.Caller.Rows.Count, 1 To \_

 Application.Caller.Columns.Count)

 ' Всего нужно чисел...

 intMax = Application.Caller.Rows.Count \* \_

 Application.Caller.Columns.Count

 ReDim aintValues(1 To intMax)

 ' Заполнение массива aintValues значениями от 1 до intMax

 For i = 1 To intMax

 aintValues(i) = i

 Next i

 ' Занесение значений в выходной массив aintOut, в произвольном \_

 порядке выбирая их из aintValues

 Randomize

 For intRow = 1 To Application.Caller.Rows.Count

 For intCol = 1 To Application.Caller.Columns.Count

 ' Определение номера элемента из aintValues

 i = Rnd \* intMax

 If i = 0 Then i = 1

 ' Занесение этого элемента в выходной массив

 aintOut(intRow, intCol) = aintValues(i)

 ' Уменьшение массива aintValues (то есть еще один его \_

 элемент выбран) - замена выбранного элемента последним \_

 в массиве

 aintValues(i) = aintValues(intMax)

 intMax = intMax - 1

 Next intCol

 Next intRow

 ' Возвращение массива значений

 dhGetRandomValues = aintOut

End Function

### Случайные числа — на основании диапазона

Листинг 2.78. Функция dhGetRandomValues1

Function dhGetRandomValues1(rgSource As Range) As Variant

 Dim intRow As Integer ' Номер текущей строки

 Dim intCol As Integer ' Номер текущего столбца

 Dim avarOut() As Variant ' Выходной массив (двумерный)

 Dim avarValues() As Variant ' Массив с возможными значениями

 Dim intValCount As Integer ' Количество возможных значений

 Dim cell As Range

 Dim i As Integer

 ReDim avarOut(1 To Application.Caller.Rows.Count, 1 To \_

 Application.Caller.Columns.Count)

 ' Всего нужно чисел...

 intValCount = rgSource.Rows.Count \* rgSource.Columns.Count

 ReDim avarValues(1 To intValCount)

 ' Заполнение массива avarValues значениями из указанного \_

 диапазона

 For Each cell In rgSource

 i = i + 1

 avarValues(i) = cell.Value

 Next cell

 ' Занесение значений в выходной массив avarOut, в произвольном \_

 порядке выбирая их из avarValues

 Randomize

 For intRow = 1 To Application.Caller.Rows.Count

 For intCol = 1 To Application.Caller.Columns.Count

 ' Определение номера элемента из avarValues

 i = Rnd \* intValCount

 If i = 0 Then i = 1

 ' Занесение этого элемента в выходной массив

 avarOut(intRow, intCol) = avarValues(i)

 Next intCol

 Next intRow

 ' Возвращение массива значений

 dhGetRandomValues1 = avarOut

End Function

# Глава 3. Создание трюков с помощью макросов

## Подсчет количества открытий файла

Листинг 3.1. Количество открытий файла (вариант 1)

Sub Auto\_Open()

 Worksheets(1).Cells(1) = Worksheets(1).Cells(1) + 1

End Sub

Листинг 3.2. Количество открытий файла (вариант 2)

Sub Auto\_Open()

 Worksheets(1).Cells(1, 1) = Worksheets(1).Cells(1, 1) + 1

End Sub

Листинг 3.3. Количество открытий файла (вариант 3)

Sub Auto\_Open()

 Worksheets(1).Range("A1") = Worksheets(1).Range("A1") + 1

End Sub

## Получение «закрытой» информации

Листинг 3.4. Извлечение данных из закрытого файла

Sub GetDataFromFile()

 Range("A1").Formula = "='C:\[Example.xls]Лист1'!A1"

End Sub

## Произвольный текст в строке состояния

Листинг 3.5. Произвольный текст в строке состояния

Sub ChangeStatusBarText()

 Application.StatusBar = "Как надоело работать!!!"

End Sub

Листинг 3.6. Восстановление строки состояния

Sub ReturnStatusBarText()

 Application.StatusBar = False

End Sub

Листинг 3.7. Бегущая строка в строке состояния

Sub MovingTextInStatusBar()

 Dim intSpaces As Integer

 ' Изменение количества пробелов в начале строки (от 20 до 0) - \_

 строка бежит (скорее, ползет) влево

 For intSpaces = 20 To 0 Step -1

 ' Запись текста в строку состояния

 Application.StatusBar = Space(intSpaces) & "Как надоело работать!!!"

 ' Выдерживаем паузу

 Application.Wait Now + TimeValue("00:00:01")

 ' Дадим Excel обработать пользовательский ввод

 DoEvents

 Next

 Application.StatusBar = False

End Sub

## Быстрое изменение заголовка окна

Листинг 3.8. Изменение заголовка окна

Sub NewTitle()

 Application.Caption = "Какая хорошая погода"

End Sub

Листинг 3.9. Изменение заголовка окна (со скрытием названия файла)

Sub NewTitle()

 Application.Caption = "Какая хорошая погода"

 ActiveWindow.Caption = ""

End Sub

Листинг 3.10. Еще один вариант изменения заголовка

Sub NewTitle()

 Application.Caption = "Какая хорошая погода"

 ActiveWindow.Caption = "А завтра будет дождь"

End Sub

Листинг 3.11. Возврат к первоначальному заголовку

Sub ReturnTitle()

 ' Возвращение заголовка приложения (то есть Excel)

 Application.Caption = Empty

 ' Указание правильного названия открытого файла (книги)

 ActiveWindow.Caption = ThisWorkbook.Name

End Sub

## Ввод данных с помощью диалогового окна

Листинг 3.12. Диалоговое окно ввода данных

Sub InputDialog()

 Dim strInput As String

 ' Вызов стандартного диалогового окна ввода данных

 strInput = InputBox("Введите данные", "Ввод данных")

End Sub

Листинг 3.13. Значения по умолчанию

Sub NewInputDialog()

 Dim strInput As String

 ' Вызов стандартного диалогового окна ввода со значением \_

 по умолчанию

 strInput = InputBox("Введите данные", "Ввод данных", \_

 "Значение по умолчанию", 200, 200)

End Sub

## Применение функции без ввода ее в ячейку

Листинг 3.14. Применение функции без ввода в ячейку

Sub Func()

 [A1] = Application.Sum([B5:B10])

End Sub

## Скрытие строк и столбцов от посторонних

Листинг 3.15. Скрытие строки

Sub HideString()

 Rows(2).Hidden = True

End Sub

Листинг 3.16. Скрытие нескольких строк

Sub HideStrings()

 Rows("3:5").Hidden = True

End Sub

Листинг 3.17. Скрытие столбца

Sub HideCollumn()

 Columns(2).Hidden = True

End Sub

Листинг 3.18. Скрытие нескольких столбцов

Sub HideCollumns()

 Columns("E:F").Hidden = True

End Sub

Листинг 3.19. Скрытие строки по имени ячейки

Sub HideCell()

 Range("Секрет").EntireRow.Hidden = True

End Sub

Листинг 3.20. Скрытие нескольких строк по адресам ячеек

Sub HideCell()

 Range("B3:D4").EntireRow.Hidden = True

End Sub

Листинг 3.21. Скрытие столбца по имени ячейки

Sub HideCell()

 Range("Секрет").EntireColumn.Hidden = True

End Sub

Листинг 3.22. Скрытие нескольких столбцов по адресам ячеек

Sub HideCell()

 Range("C2:D5").EntireColumn.Hidden = True

End Sub

## Быстрое выделение ячеек, расположенных через интервал

Листинг 3.23. Выделение ячеек через интервал (вариант 1)

Sub IntervalCellSelect()

 Dim intFirstRow As Integer ' Первая строка для выделения

 Dim intLastRow As Integer ' Последняя строка для выделения

 Dim rgCells As Range ' Объединение выделяемых ячеек

 Dim intRow As Integer

 intFirstRow = 3

 intLastRow = 300

 ' Формирование объединения ячеек в столбце "B" от строки \_

 intFirstRow до строки intLastRow с шагом 3

 For intRow = intFirstRow To intLastRow Step 3

 If rgCells Is Nothing Then

 ' Первая ячейка в объединении

 Set rgCells = Cells(intRow, 1)

 Else

 ' Добавление очередной ячейки в объединение

 Set rgCells = Union(rgCells, Cells(intRow, 1))

 End If

 Next

 ' Выделение всех ячеек в объединении

 rgCells.Select

End Sub

Листинг 3.24. Выделение ячеек через интервал (вариант 2)

Sub IntervalCellSelect()

 Dim intFirstRow As Integer ' Первая строка для выделения

 Dim intLastRow As Integer ' Последняя строка для выделения

 Dim rgCells As Range ' Объединение выделяемых ячеек

 Dim cell As Range ' Текущая ячейка

 Dim intRow As Integer

 intFirstRow = 3

 intLastRow = 300

 ' Формирование объединения ячеек в столбце "B" от строки \_

 intFirstRow до строки intLastRow с шагом 3

 For intRow = intFirstRow To intLastRow Step 3

 Set cell = Cells(intRow, 1)

 Set rgCells = Union(cell, \_

 IIf(intRow = intFirstRow, cell, rgCells))

 Next

 ' Выделение всех ячеек в объединении

 rgCells.Select

End Sub

Листинг 3.25. Выделение нескольких диапазонов

Sub SelectRange()

 Range("D3:D10, A3:A10 , F3").Select

End Sub

## Определение количества ячеек в диапазоне и суммы их значений

Листинг 3.26. Сумма данных диапазона

Sub CalculateSum()

 Dim i As Integer

 Dim intSum As Integer

 ' Расчет суммы ячеек столбца "A" (с первой по пятую)

 For i = 1 To 5

 intSum = intSum + Cells(i, 1)

 Next

 MsgBox "Сумма ячеек: " & intSum

End Sub

Листинг 3.27. Сумма ячеек с числовыми значениями

Sub CalculateSum()

 Dim i As Integer

 Dim intSum As Integer

 ' Расчет суммы ячеек столбца "A" (с первой по пятую)

 For i = 1 To 5

 If IsNumeric(Cells(i, 1)) Then

 intSum = intSum + Cells(i, 1)

 End If

 Next

 MsgBox "Сумма ячеек: " & intSum

End Sub

Листинг 3.28. Подсчет количества ячеек

Sub CountOfCells()

 MsgBox (Range("A1:A20, D1:D20").Count)

End Sub

## Подсчет именованных объектов

Листинг 3.29. Количество именованных объектов

Sub CountNames()

 Dim intNamesCount As Integer

 ' Получаем и отображаем количество имен в активной \_

 рабочей книге

 intNamesCount = ActiveWorkbook.Names.Count

 If intNamesCount = 0 Then

 MsgBox "Имен нет"

 Else

 MsgBox "Имен: " & intNamesCount & " шт."

 End If

End Sub

Листинг 3.30. Количество имен рабочей книги

Sub CountNames()

 Dim intNamesCount As Integer

 ' Получаем и отображаем количество имен на активном \_

 листе рабочей книги

 intNamesCount = Names.Count

 If intNamesCount = 0 Then

 MsgBox "Имен нет"

 Else

 MsgBox "Имен: " & intNamesCount & " шт."

 End If

End Sub

## Быстрый поиск курсора

Листинг 3.31. Поиск активной ячейки

Sub FindActiveCell()

 ' Выводим адрес активной ячейки

 MsgBox ActiveCell.Address

End Sub

## Поиск начала и окончания диапазона, содержащего данные

Листинг 3.32. Поиск данных

Sub FindSheetData()

 ' Выводим диапазон используемых ячеек листа

 MsgBox ActiveSheet.UsedRange.Address

End Sub

Листинг 3.33. Поиск начала данных

Sub FindStartOfData()

 With ActiveSheet

 ' Заносим текст в ячейку, являющуюся левой верхней \_

 ячейкой используемого диапазона

 .Cells(.UsedRange.Row, .UsedRange.Column).Value = \_

 "Начало данных"

 End With

End Sub

## Трюки с примечаниями

### Подсчет примечаний

Листинг 3.34. Подсчет примечаний

Sub CountOfComment()

 Dim intCommentCount As Integer

 ' Получение и отображение количества примечаний \_

 на текущем листе

 intCommentCount = ActiveSheet.Comments.Count

 If intCommentCount = 0 Then

 MsgBox "Примечаний нет"

 Else

 MsgBox "Примечаний: " & intCommentCount & " шт."

 End If

End Sub

### Вывод на экран всех примечаний рабочего листа

Листинг 3.35. Список примечаний

Sub ShowComments()

 Dim cell As Range

 Dim rgCells As Range

 ' Получение всех ячеек с примечаниями

 Set rgCells = Selection.SpecialCells(xlComments)

 If rgCells Is Nothing Then

 ' Примечаний нет

 Exit Sub

 End If

 ' Проходим по всем ячейкам диапазона

 For Each cell In rgCells

 ' Вывод примечаний в соседнюю ячейку

 cell.Next.Value = cell.Comment.Text

 Next

End Sub

Листинг 3.36. Список примечаний защищенных листов

Sub ShowComments1()

 Dim cell As Range

 Dim strFirstAddress As String

 Dim strComments As String

 ' Получаем все ячейки выделения, в которых есть комментарий

 Set cell = Selection.Find("\*", LookIn:=xlComments)

 If Not cell Is Nothing Then

 ' Сохранение адреса первой найденной ячейки \_

 (для предотвращения зацикливания поиска)

 strFirstAddress = cell.Address

 Do

 ' Добавление текста примечания в выходную строку

 strComments = strComments & "Комментарий: " & \_

 cell.Comment.Text & Chr(13)

 ' Продолжение поиска

 Set cell = Selection.FindNext(cell)

 Loop While Not cell Is Nothing And \_

 cell.Address <> strFirstAddress

 End If

 If strComments <> "" Then

 ' Отображение окна с текстом примечаний

 MsgBox strComments

 Else

 MsgBox "В выделенной ячейке/ячейках комментариев нет"

 End If

End Sub

### Создание списка примечаний рабочего листа

Листинг 3.37. Перечень примечаний в отдельном списке (вариант 1)

Sub ListOfComments()

 Dim cell As Range

 Dim rgCells As Range

 Dim intRow As Integer

 ' Получение всех ячеек с примечаниями

 On Error Resume Next

 Set rgCells = Selection.SpecialCells(xlComments)

 If rgCells Is Nothing Then

 ' Примечаний нет

 Exit Sub

 End If

 ' Проходим по всем ячейкам диапазона

 For Each cell In rgCells

 ' Вывод примечаний в ячейку столбца "C"

 intRow = intRow + 1

 Cells(intRow, 3) = cell.Comment.Text

 Next

End Sub

Листинг 3.38. Перечень примечаний в отдельном списке (вариант 2)

Sub ListOfComments1()

 Dim cell As Range

 Dim strFirstAddress As String

 Dim intRow As Integer

 ' Получение всех ячеек выделения, в которых есть примечания

 Set cell = Cells.Find("\*", LookIn:=xlComments)

 If Not cell Is Nothing Then

 ' Сохранение адреса первой найденной ячейки \_

 (для предотвращения зацикливания поиска)

 strFirstAddress = cell.Address

 Do

 ' Вывод текста в столбец "C"

 intRow = intRow + 1

 Cells(intRow, 3) = cell.Comment.Text

 ' Продолжение поиска

 Set cell = Cells.FindNext(cell)

 Loop While Not cell Is Nothing And \_

 cell.Address <> strFirstAddress

 End If

End Sub

### Несколько трюков в одном примере

Листинг 3.39. Операции с примечаниями

Sub CountOfComments()

 Dim intCommentCount As Integer

 ' Получение и отображение количества примечаний

 intCommentCount = ActiveSheet.Comments.Count

 If intCommentCount = 0 Then

 MsgBox "Текущая рабочая книга не содержит примечаний.", \_

 vbInformation

 Else

 MsgBox "В текущей рабочей книге содержится " & intCommentCount \_

 & " комментариев.", vbInformation

 End If

End Sub

Sub SelectComments()

 ' Выделение всех ячеек с примечаниями

 Cells.SpecialCells(xlCellTypeComments).Select

End Sub

Sub ShowComments()

 ' Отображение всех примечаний

 If Application.DisplayCommentIndicator = xlCommentAndIndicator Then

 Application.DisplayCommentIndicator = xlCommentIndicatorOnly

 Else

 Application.DisplayCommentIndicator = xlCommentAndIndicator

 End If

End Sub

Sub ListOfCommentsToFile()

 Dim rgCells As Range ' Ячейки с примечаниями

 Dim intDefListCount As Integer ' Используется для временного \_

 хранения количества листов в книге по умолчанию

 Dim strSheet As String ' Имя анализируемого листа

 Dim strWorkBook As String ' Имя книги с анализируемым листом

 Dim intRow As Integer

 Dim cell As Range

 ' Получение ячеек с примечаниями

 On Error Resume Next

 Set rgCells = ActiveSheet.Cells.SpecialCells(xlComments)

 On Error GoTo 0

 ' Если примечаний нет, то можно не продолжать

 If rgCells Is Nothing Then

 MsgBox "Текущая рабочая книга не содержит примечаний.", \_

 vbInformation

 Exit Sub

 End If

 ' Сохранение имен анализируемого листа и книги

 strSheet = ActiveSheet.Name

 strWorkBook = ActiveWorkbook.Name

 ' Создание отдельной книги с одним листом \_

 для отображения результатов

 intDefListCount = Application.SheetsInNewWorkbook

 Application.SheetsInNewWorkbook = 1

 Workbooks.Add

 Application.SheetsInNewWorkbook = intDefListCount

 ActiveWorkbook.Windows(1).Caption = "Comments for " & strSheet & \_

 " in " & strWorkBook

 ' Создание списка примечаний

 Cells(1, 1) = "Адрес"

 Cells(1, 2) = "Содержимое"

 Cells(1, 3) = "Комментарий"

 Range(Cells(1, 1), Cells(1, 3)).Font.Bold = True

 intRow = 2 ' Данные начинаются со второй строки

 For Each cell In rgCells

 Cells(intRow, 1) = cell.Address(rowabsolute:=False, \_

 columnabsolute:=False)

 Cells(intRow, 2) = " " & cell.Formula

 Cells(intRow, 3) = cell.comment.Text

 intRow = intRow + 1

 Next

End Sub

Sub ChangeCommentColor()

 ' Автоматическое изменение цвета комментариев

 Dim comment As comment

 For Each comment In ActiveSheet.Comments

 ' Задаем случайные цвета заливки и шрифта комментариев

 comment.Shape.Fill.ForeColor.SchemeColor = Int((80) \* Rnd + 1)

 comment.Shape.TextFrame.Characters.Font.ColorIndex = Int((56 \_

 ) \* Rnd + 1)

 Next

End Sub

## Дополнение панели инструментов

Листинг 3.40. Дополнение панели инструментов

Sub AddCustomCommandBar()

 ' Добавление кнопки на панель инструментов

 With Application.CommandBars(3).Controls.Add(Type:=msoControlButton)

 .FaceId = 42 ' Значок Word

 .Caption = "Кнопка"

 .OnAction = "Макрос"

 End With

End Sub

Листинг 3.41. Добавление кнопки на панель инструментов

Sub AddCustomButton()

 ' Добавление кнопки на панель инструментов

 With Application.Toolbars(1).ToolbarButtons.Add(button:=222)

 .Name = "Кнопка"

 .OnAction = "Макрос"

 End With

End Sub

## Примеры создания панелей инструментов

Листинг 3.42. Панель с одной кнопкой

Sub CreateCustomControlBar()

 ' Создание панели инструментов

 With Application.CommandBars.Add(Name:="Панель", Temporary:=True)

 ' Создание и настройка кнопки

 With .Controls.Add(Type:=msoControlButton)

 .Style = msoButtonIconAndCaption

 .FaceId = 66

 .Caption = "Просто кнопка"

 End With

 ' Покажем панель

 .Visible = True

 End With

End Sub

Листинг 3.43. Панель с двумя кнопками

Sub CreateCustomControlBar()

 ' Создание панели инструментов

 With Application.CommandBars.Add(Name:="Панель", Temporary:=True, \_

 Position:=msoBarLeft)

 ' Создание и настройка первой кнопки

 With .Controls.Add(Type:=msoControlButton)

 .Style = msoButtonWrapCaption

 .Caption = "Просто кнопка"

 End With

 ' Создание и настройка второй кнопки

 With .Controls.Add(Type:=msoControlButton)

 .Style = msoButtonIconAndWrapCaption

 .Caption = "Кнопка"

 .FaceId = 225

 End With

 ' Покажем панель

 .Visible = True

 End With

End Sub

Листинг 3.44. Создание панели справа

Sub CreateCustomControlBar()

 ' Создание панели инструментов

 With Application.CommandBars.Add(Name:="Правая панель", \_

 Temporary:=True)

 ' Создание и настройка кнопки

 With .Controls.Add(Type:=msoControlButton)

 .Style = msoButtonWrapCaption

 .Caption = "Кнопка"

 End With

 ' Задание позиции - справа

 .Position = msoBarRight

 ' Покажем панель

 .Visible = True

 End With

End Sub

## Формирование пользовательского меню

Листинг 3.45. Создание пользовательского меню (вариант 1)

Sub AddCustomMenu()

 ' Добавление меню

 With Application.CommandBars(1).Controls.Add(Type:=msoControlPopup, \_

 Temporary:=True)

 .Caption = "Архив"

 With .Controls

 ' Добавление и настройка первого пункта

 With .Add(Type:=msoControlButton)

 .FaceId = 280

 .Caption = "Просмотр"

 .OnAction = "Макрос1"

 End With

 ' Добавление вложенного меню

 With .Add(Type:=msoControlPopup)

 .Caption = "База данных"

 With .Controls

 ' Добавление и настройка первого пункта \_

 вложенного меню

 With .Add(Type:=msoControlButton)

 .FaceId = 1643

 .Caption = "Поставщики"

 .OnAction = "Макрос2"

 End With

 ' Добавление и настройка второго пункта \_

 вложенного меню

 With .Add(Type:=msoControlButton)

 .FaceId = 1000

 .Caption = "Покупатели"

 .OnAction = "Макрос3"

 End With

 End With

 End With

 End With

 End With

End Sub

Листинг 3.46. Создание пользовательского меню (вариант 2)

Sub AddCustomMenu1()

 ' Добавление меню с названием "Архив" в часть меню, \_

 относящуюся к рабочей книге

 With MenuBars("Worksheet").Menus.Add(Caption:="Архив")

 ' Добавление кнопки

 .MenuItems.Add Caption:="Просмотр", OnAction:="Макрос1"

 ' Добавление подменю

 With .MenuItems.AddMenu(Caption:="База данных")

 ' Добавление пунктов подменю

 .MenuItems.Add Caption:="Поставщики", OnAction:="Макрос2"

 .MenuItems.Add Caption:="Покупатели", OnAction:="Макрос3"

 End With

 End With

End Sub

Листинг 3.47. Создание пользовательского меню (вариант 3)

Sub AddCustomMenu2()

 ' Добавление меню с названием "Архив" в часть меню, \_

 относящуюся к рабочей книге

 With MenuBars("Worksheet").Menus.Add(Caption:="Архив")

 ' Добавление кнопки

 .MenuItems.Add Caption:="Просмотр", OnAction:="Макрос1"

 ' Добавление подменю

 With .MenuItems.AddMenu(Caption:="База данных")

 ' Добавление первого пункта подменю

 With .MenuItems.Add(Caption:="Поставщики")

 ' Настройка кнопки

 .OnAction = "Макрос2"

 End With

 ' Добавление второго пункта подменю

 With .MenuItems.Add(Caption:="Покупатели")

 ' Настройка кнопки

 .OnAction = "Макрос3"

 End With

 End With

 End With

End Sub

## Проверка наличия файла по указанному пути

Листинг 3.48. Проверка наличия файла (вариант 1)

Sub VerifyFileLocation()

 Dim strFileName As String

 Dim strFileTitle As String

 ' Имя и путь искомого файла

 strFileTitle = "primer.xls"

 strFileName = "C:\Документы\primer.xls"

 ' Проверка наличия файла (функция Dir возвращает пустую \_

 строку, если по указанному пути файл обнаружить не удалось)

 If Dir(strFileName) <> "" Then

 MsgBox "Файл " & strFileTitle & " найден"

 Else

 MsgBox "Файл " & strFileTitle & " не найден"

 End If

End Sub

Листинг 3.49. Проверка наличия файла (вариант 2)

Sub VerifyFileLocation1()

 Dim strFileName As String

 ' Имя искомого файла

 strFileName = "C:\Документы\primer.xls"

 ' Проверка наличия файла (функция Dir возвращает пустую \_

 строку, если по указанному пути файл обнаружить не удалось)

 If Dir(strFileName) <> "" Then

 MsgBox "Файл " & strFileName & " найден"

 Else

 MsgBox "Файл " & strFileName & " не найден"

 End If

End Sub

Листинг 3.50. Поиск нужного файла

Sub FileSearch()

 Dim strFileName As String

 Dim strFolder As String

 Dim strFullPath As String

 ' Задание имени папки для поиска

 strFolder = InputBox("Определите папку:")

 If strFolder = "" Then Exit Sub

 ' Задание имени файла для поиска

 strFileName = Application.InputBox("Введите имя файла:")

 If strFileName = "" Then Exit Sub

 ' При необходимости дополняем имя папки "\"

 If Right(strFolder, 1) <> "\" Then strFolder = strFolder & "\"

 ' Полный путь файла

 strFullPath = strFolder & strFileName

 ' Вывод окна с отчетом о поиске средствами VBA

 MsgBox "Использование команды VBA..." & vbCrLf & vbCrLf & \_

 dhSearchVBA(strFullPath), vbInformation, strFullPath

 ' Вывод окна с отчетом о поиске средствами объекта FileSearch

 MsgBox "Использование объекта FileSearch..." & vbCrLf & \_

 vbCrLf & dhSearchFileSearch(strFolder, strFileName), vbInformation, \_

 strFullPath

 ' Вывод окна с отчетом о поиске средствами объекта \_

 FileSystemObject

 MsgBox "Использование объекта FileSystemObject..." & vbCrLf & \_

 vbCrLf & dhSearchFileSystemObject(strFullPath), vbInformation, \_

 strFullPath

End Sub

Function dhSearchVBA(varFullPath As Variant) As Boolean

 ' Использование команды VBA

 dhSearchVBA = Dir(varFullPath) <> ""

End Function

Function dhSearchFileSearch(varFolder As Variant, varFileName \_

 As Variant) As Boolean

 ' Использование объекта FileSearch

 With Application.FileSearch

 ' Создание нового поиска

 .NewSearch

 ' Имя для поиска

 .FileName = varFileName

 ' Папка поиска

 .LookIn = varFolder

 ' Собственно поиск

 .Execute

 dhSearchFileSearch = .FoundFiles.Count <> 0

 End With

End Function

Function dhSearchFileSystemObject(varFullPath As Variant) As Boolean

 Dim objFSObject As Object

 ' Использование объекта FileSystemObject

 Set objFSObject = CreateObject("Scripting.FileSystemObject")

 dhSearchFileSystemObject = objFSObject.FileExists(varFullPath)

End Function

## Автоматизация удаления файлов

Листинг 3.51. Удаление файла

Sub DeleteFile()

 Kill "C:\Документы\primer.xls"

End Sub

Листинг 3.52. Удаление группы файлов

Sub DeleteFiles()

 ' Удаление всех файлов с расширением XLS из заданной папки

 Kill "C:\Документы" & "\*.xls"

End Sub

## Перечень имен листов в виде гиперссылок

Листинг 3.53. Перечень имен рабочих листов

Sub SheetNamesAsHyperLinks()

 Dim sheet As Worksheet

 Dim cell As Range

 With ActiveWorkbook

 ' Просмотр всех листов книги и создание гиперссылок на них \_

 на первом листе

 For Each sheet In ActiveWorkbook.Worksheets

 Set cell = Worksheets(1).Cells(sheet.Index, 1)

 .Worksheets(1).Hyperlinks.Add Anchor:=cell, Address:="", \_

 SubAddress:="'" & sheet.Name & "'" & "!A1"

 cell.Formula = sheet.Name

 Next

 End With

End Sub

## Удаление пустых строк на рабочем листе

Листинг 3.54. Удаление пустых строк (вариант 1)

Sub DeleteEmptyStrings()

 Dim intLastRow As Integer ' Номер последней используемой строки

 Dim intRow As Integer ' Номер проверяемой строки

 ' Получение номера последней используемой строки

 intLastRow = Worksheets(ActiveSheet.Index).UsedRange.Row + \_

 Worksheets(ActiveSheet.Index).UsedRange.Rows.Count - 1

 ' Счетчик устанавливается на используемую первую строку

 intRow = Worksheets(ActiveSheet.Index).UsedRange.Row

 ' Удаление пустых строк

 Do While intRow <= intLastRow

 If ActiveSheet.Rows(intRow).Text = "" Then

 ' Удаление строки

 ActiveSheet.Rows(intRow).Delete

 ' Данные сдвинулись вверх, поэтому номер последней \_

 строки уменьшился, а текущей - не изменился

 intLastRow = intLastRow - 1

 Else

 ' Текущая строка заполнена - переходим к следующей

 intRow = intRow + 1

 End If

 Loop

End Sub

Листинг 3.55. Удаление пустых строк (вариант 2)

Sub DeleteEmptyStrings1()

 Dim intRow As Integer

 Dim intLastRow As Integer

 ' Получение номера последней используемой строки

 intLastRow = ActiveSheet.UsedRange.Row + \_

 ActiveSheet.UsedRange.Rows.Count - 1

 ' Удаление пустых строк

 For intRow = intLastRow To 1 Step -1

 If ActiveSheet.Rows(intRow).Text = "" Then

 ActiveSheet.Rows(intRow).Delete

 End If

 Next intRow

End Sub

## Запись текущих данных в текстовый файл

Листинг 3.56. Запись в текстовый файл

Sub SaveAsText()

 Dim cell As Range

 ' Открытие файла для сохранения (имя файла соответствует имени \_

 рабочей книги, но отличается расширением - TXT)

 Open ThisWorkbook.Path & "\" & ThisWorkbook.Name & ".txt" \_

 For Output As #1

 ' Запись содержимого заполненных ячеек таблицы в файл

 For Each cell In ActiveSheet.UsedRange

 If Not IsEmpty(cell) Then

 Print #1, cell.Address, cell.Formula

 End If

 Next

 ' Не забываем закрывать файл

 Close #1

End Sub

Листинг 3.57. Экспорт в текстовый файл

Sub SaveAsText1()

 Dim cell As Range

 ' Открытие файла для сохранения (имя файла соответствует имени \_

 рабочей книги, но отличается расширением - TXT)

 Open ThisWorkbook.Path & "\" & ThisWorkbook.Name & ".txt" \_

 For Output As #1

 ' Запись содержимого заполненных ячеек таблицы в файл

 For Each cell In ActiveSheet.UsedRange

 If Not IsEmpty(cell) Then

 Print #1, cell.Address, cell.FormulaLocal

 End If

 Next

 ' Не забываем закрывать файл

 Close #1

End Sub

## Экспорт и импорт данных

Листинг 3.58. Экспорт и импорт данных

Sub ExportAsText()

 Dim lngRow As Long

 Dim intCol As Integer

 ' Открытие файла для сохранения

 Open "C:\primer.txt" For Output As #1

 ' Запись выделенной части таблицы в файл (построчно)

 For lngRow = 1 To Selection.Rows.Count

 ' Запись содержимого всех столбцов строки lngRow

 For intCol = 1 To Selection.Columns.Count

 Write #1, Selection.Cells(lngRow, intCol).Value;

 Next intCol

 ' Начнем новую строку в файле

 Print #1, ""

 Next lngRow

 ' Не забываем закрыть файл

 Close #1

End Sub

Sub ImportText()

 Dim strLine As String ' Одна строка файла

 Dim strCurChar As String \* 1 ' Анализируемый символ строки файла

 Dim strValue As String ' Значение для записи в ячейку

 Dim lngRow As Long ' Номер текущей строки

 Dim intCol As Integer ' Номер текущего столбца

 Dim i As Integer

 ' Открытие импортируемого файла

 Open "C:\primer.txt" For Input As #1

 ' Считываем все строки файла и записываем данные, разделенные \_

 запятой, в ячейки таблицы (начиная с текущей ячейки)

 Do Until EOF(1)

 ' Считываем строку из файла

 Line Input #1, strLine

 ' Разбираем считанную строку

 For i = 1 To Len(strLine)

 strCurChar = Mid(strLine, i, 1)

 If strCurChar = "," Then

 ' Найден разделитель столбцов - запятая. Запишем \_

 сформированное значение в ячейку

 ActiveCell.Offset(lngRow, intCol) = strValue

 intCol = intCol + 1

 strValue = ""

 ElseIf i = Len(strLine) Then

 ' Конец строки - запишем в таблицу последнее \_

 значение в строке (перед этим дополним его последним \_

 символом строки, кроме кавычки)

 If strCurChar <> Chr(34) Then

 strValue = strValue & strCurChar

 End If

 ' Запись в таблицу

 ActiveCell.Offset(lngRow, intCol) = strValue

 strValue = ""

 ElseIf strCurChar <> Chr(34) Then

 ' Добавление символа в формируемое значение ячейки \_

 (кавычки игнорируются)

 strValue = strValue & strCurChar

 End If

 Next i

 ' Переход к новой строке таблицы

 intCol = 0

 lngRow = lngRow + 1

 Loop

 ' Закрываем файл

 Close #1

End Sub

## Одновременное умножение всех данных диапазона

Листинг 3.59. Умножение данных

Sub MultAllCells()

 Dim dblMult As Double

 Dim cell As Range

 ' Ввод коэффициента для умножения

 dblMult = InputBox("Введите коэффициент, на который следует умножать")

 ' Умножение содержимого на введенный коэффициент

 For Each cell In Selection

 If IsNumeric(cell.Value) And cell.Value <> "" Then

 ' Умножаются только ячейки, содержащие числовые данные

 cell.Value = cell.Value \* dblMult

 Else

 MsgBox "В ячейке " & cell.Address & " нечисловое значение"

 End If

 Next

End Sub

## Преобразование таблицы Excel в HTML-формат

Листинг 3.60. Преобразование таблицы в HTML-формат

Sub ExportAsHtml()

 Dim strStyle As String ' Параметры стиля отображения ячейки

 Dim strAlign As String ' Параметры выравнивания ячейки

 Dim strOut As String ' Выходная строка с HTML-кодом

 Dim cell As Object ' Обрабатываемая ячейка

 Dim strCellText As String ' Текст обрабатываемой ячейки

 Dim lngRow As Long ' Номер строки обрабатываемой ячейки

 Dim lngLastRow As Long ' Номер строки предыдущей ячейки

 Dim strTemp As String

 Dim objWordApp As Object

 Dim i As Long

 lngLastRow = Selection.Row

 ' Просмотр всех выделенных ячеек

 For Each cell In Selection

 ' Значение строки для рассматриваемой ячейки

 lngRow = cell.Row

 ' Если перешли на другую строку, то вставляем <tr>

 If lngRow <> lngLastRow Then

 strOut = strOut & vbTab & "</tr>" & vbCrLf & vbTab & \_

 "<tr>" & vbCrLf

 ' Переход на следующую строку

 lngLastRow = lngRow

 End If

 ' Задание шрифта ячейки

 If Not IsNull(cell.Font.Size) Then

 strStyle = " style=" & "font-size: " & Int(100 \* \_

 cell.Font.Size / 19) & "%;"

 End If

 ' Для полужирного шрифта вставляем <b>

 If cell.Font.Bold Then

 strCellText = "<b>" & strCellText & "</b>"

 End If

 ' Задание выравнивания

 If cell.HorizontalAlignment = xlRight Then

 ' По правому краю

 strAlign = " align=" & "right"

 ElseIf cell.HorizontalAlignment = xlCenter Then

 ' По центру

 strAlign = " align=" & "center"

 Else

 ' По левому краю (по умолчанию)

 strAlign = ""

 End If

 ' Чтение текста в ячейке

 strCellText = cell.Text

 ' Если нужно, то вертикальный вывод текста (в строку strTemp \_

 с последующим перенесением обратно в strCellText)

 If cell.Orientation <> xlHorizontal Then

 strTemp = ""

 ' Печать после каждого символа специального \_

 разделителя - <br>

 For i = 1 To Len(strCellText)

 strTemp = strTemp & Mid$(strCellText, i, 1) & "<br>"

 Next i

 strCellText = strTemp

 strStyle = ""

 End If

 strOut = strOut & vbTab & vbTab & "<td" & strStyle & strAlign \_

 & ">" & strCellText & "</td>" & vbCrLf

 Next

 ' Вставка <tr> для первой строки и </tr> - для последней

 strOut = vbTab & "<tr>" & vbCrLf & strOut & vbTab & "</tr>" & vbCrLf

 ' Вставка дескриптора <table>

 strOut = "<table border=1 cellpadding=3 cellspacing=1>" & vbCrLf & \_

 strOut & vbCrLf & "</table>"

 ' Запускаем Word и показываем в нем сформированный HTML-код

 Set objWordApp = CreateObject("Word.Application")

 objWordApp.documents.Add

 objWordApp.Selection = strOut

 objWordApp.Selection.Copy

 objWordApp.Visible = True

 Set objWordApp = Nothing

End Sub

Листинг 3.61. Экспорт данных в HTM-файл

Sub ExportAsHtmlFile()

 Dim strStyle As String ' Параметры стиля отображения ячейки

 Dim strAlign As String ' Параметры выравнивания ячейки

 Dim strOut As String ' Выходная строка с HTML-кодом

 Dim cell As Object ' Обрабатываемая ячейка

 Dim strCellText As String ' Текст обрабатываемой ячейки

 Dim lngRow As Long ' Номер строки обрабатываемой ячейки

 Dim lngLastRow As Long ' Номер строки предыдущей ячейки

 Dim strTemp As String

 Dim strFileName As String ' Имя файла для сохранения HTML-кода

 Dim i As Long

 ' Запрос у пользователя имени файла для сохранения

 strFileName = Application.GetSaveAsFilename( \_

 InitialFileName:="Primer.htm", \_

 fileFilter:="HTML Files(\*.htm), \*.htm")

 ' Проверка, задал ли пользователь имя файла (если нет, \_

 то можно выходить)

 If strFileName = "" Then Exit Sub

 lngLastRow = Selection.Row

 ' Просмотр всех выделенных ячеек

 For Each cell In Selection

 ' Значение строки для рассматриваемой ячейки

 lngRow = cell.Row

 ' Если перешли на другую строку, то вставляем <tr>

 If lngRow <> lngLastRow Then

 strOut = strOut & vbTab & "</tr>" & vbCrLf & vbTab & \_

 "<tr>" & vbCrLf

 ' Переход на следующую сроку

 lngLastRow = lngRow

 End If

 ' Задание шрифта ячейки

 If Not IsNull(cell.Font.Size) Then

 strStyle = " style=" & "font-size: " & Int(100 \* \_

 cell.Font.Size / 19) & "%;"

 End If

 ' Для полужирного шрифта вставляем <b>

 If cell.Font.Bold Then

 strCellText = "<b>" & strCellText & "</b>"

 End If

 ' Задание выравнивания

 If cell.HorizontalAlignment = xlRight Then

 ' По правому краю

 strAlign = " align=" & "right"

 ElseIf cell.HorizontalAlignment = xlCenter Then

 ' По центру

 strAlign = " align=" & "center"

 Else

 ' По левому краю (по умолчанию)

 strAlign = ""

 End If

 ' Чтение текста в ячейке

 strCellText = cell.Text

 ' Если нужно, то вертикальный вывод текста (в строку strTemp \_

 с последующим перенесением обратно в strCellText)

 If cell.Orientation <> xlHorizontal Then

 strTemp = ""

 ' Печать после каждого символа специального \_

 разделителя - <br>

 For i = 1 To Len(strCellText)

 strTemp = strTemp & Mid$(strCellText, i, 1) & "<br>"

 Next i

 strCellText = strTemp

 strStyle = ""

 End If

 strOut = strOut & vbTab & vbTab & "<td" & strStyle & \_

 strAlign & ">" & strCellText & "</td>" & vbCrLf

 Next

 ' Вставка <tr> для первой строки и </tr> - для последней

 strOut = vbTab & "<tr>" & vbCrLf & strOut & vbTab & "</tr>" & vbCrLf

 ' Вставка дескриптора <table>

 strOut = "<table border=1 cellpadding=3 cellspacing=1>" \_

 & vbCrLf & strOut & vbCrLf & "</table>"

 ' Сохранение HTML-кода в файл

 Open strFileName For Output As 1

 Print #1, strOut

 Close 1

 ' Вывод окна с информационным сообщением о результатах работы

 MsgBox Selection.Count & " ячеек экспортировано в файл " & \_

 strFileName

End Sub

## Поиск данных нештатными средствами

Листинг 3.62. Поиск данных с помощью макроса

Sub CustomSearch()

 Dim strFindData As String

 Dim rgFound As Range

 Dim i As Integer

 ' Ввод строки для поиска

 strFindData = InputBox("Введите данные для поиска")

 ' Просмотр всех рабочих листов книги

 For i = 1 To Worksheets.Count

 With Worksheets(i).Cells

 ' Поиск на i-м листе

 Set rgFound = .Find(strFindData, LookIn:=xlValues)

 If Not rgFound Is Nothing Then

 ' Ячейка с заданным значением найдена - выделим ее

 Sheets(i).Select

 rgFound.Select

 Exit Sub

 End If

 End With

 Next

 ' Поиск завершен. Ячейка не найдена

 MsgBox ("Поиск не дал результатов")

End Sub

## Включение автофильтра с помощью макроса

Листинг 3.63. Включение автофильтра

Sub EnableAutoFilter()

 On Error Resume Next

 Selection.AutoFilter

End Sub

## Трюки с форматированием

### Изменение формата представления чисел нештатными средствами

Листинг 3.64. Формат «два знака после запятой»

Sub ChangeNumberFormat()

 Selection.NumberFormat = "0.00"

End Sub

Листинг 3.65. Использование разделителя по разрядам

Sub ThreeNullSepatator()

 Selection.NumberFormat = "#,##"

End Sub

Листинг 3.66. Изменение формата

Sub ChangeNumerFormatEx()

 Selection.NumberFormat = "#,##0.00"

End Sub

### Помещение последнего символа ячейки над строкой

Листинг 3.67. Помещение последнего символа над строкой

Sub LastCharUp()

 ' Изменение расположения последнего символа ячейки

 With ActiveCell.Characters(Start:=Len(Selection), Length:=1).Font

 .Superscript = True

 End With

End Sub

### Создание нестандартной рамки

Листинг 3.68. Нестандартная рамка

Sub ChangeSelGrid()

 ' Оформление границ выделения

 ' Левая граница

 With Selection.Borders(xlEdgeLeft)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

 ' Правая граница

 With Selection.Borders(xlEdgeRight)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

 ' Верхняя граница

 With Selection.Borders(xlEdgeTop)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

 ' Нижняя граница

 With Selection.Borders(xlEdgeBottom)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

 ' Изменение сетки внутри выделения

 ' Вертикальные линии сетки

 With Selection.Borders(xlInsideVertical)

 .LineStyle = xlContinuous

 .Weight = xlHairline

 .ColorIndex = xlAutomatic

 End With

 ' Горизонтальные линии сетки

 With Selection.Borders(xlInsideHorizontal)

 .LineStyle = xlContinuous

 .Weight = xlHairline

 .ColorIndex = xlAutomatic

 End With

End Sub

### Быстрая вставка фамилий должностных лиц в документ

Листинг 3.69. Вставка текста в документ

Sub InsertCustomText()

 ' Заполнение текущей ячейки

 ActiveCell = "Генеральный директор"

 Selection.Font.Bold = True

 ' Фамилия на три столбца правее должности

 Cells(ActiveCell.Row, ActiveCell.Column + 3).Select

 ActiveCell.FormulaR1C1 = "А. Б. Рублев"

 Selection.Font.Bold = True

 ' Ячейка с "Главный бухгалтер" на три столбца левее \_

 и на три строки ниже ячейки с фамилией директора

 Cells(ActiveCell.Row + 3, ActiveCell.Column - 3).Select

 ActiveCell = "Главный бухгалтер"

 Selection.Font.Bold = True

 ' Фамилия на три столбца правее должности

 Cells(ActiveCell.Row, ActiveCell.Column + 3).Select

 ActiveCell = "Т. С. Копейкин"

 Selection.Font.Bold = True

End Sub

### Вызов окна настройки шрифта

Листинг 3.70. Окно настройки шрифта

Sub ShowFontDialog()

 ' Вызов стандартного окна настройки шрифта текущей ячейки

 Application.Dialogs(xlDialogActiveCellFont).Show

End Sub

## Вывод информации о текущем документе

Листинг 3.71. Информация о текущем документе

Sub ShowInfo()

 Dim i As Integer

 ' Выводим имя файла рабочей книги

 Range("A1") = ActiveWorkbook.Name

 ' Выводим имя текущего листа

 Range("B1") = ActiveSheet.Name

 ' Выводим номера листов

 For i = 1 To ActiveWorkbook.Sheets.Count

 ActiveSheet.Cells(i, 3) = i

 Next i

End Sub

## Вывод результата расчетов в отдельном окне

Листинг 3.72. Окно с результатом расчетов

Sub ResultToWindow()

 ' Переходим на первый лист

 Worksheets(1).Activate

 ' Заносим в ячейки данные

 Range("A2") = 5

 Range("A3") = "=A2+3"

 ' Выводим результат расчета

 MsgBox Range("A3").Formula + " = " + str(Range("A3").Value)

End Sub

## Вывод разрешения монитора

Листинг 3.73. Разрешение монитора

'Объявление API-функции

Declare Function GetSystemMetrics Lib "user32" \_

 (ByVal nIndex As Long) As Long

' Константы, которые передаются в функцию для определения \_

 горизонтального и вертикального размеров изображения

Const SM\_CXSCREEN = 0

Const SM\_CYSCREEN = 1

Sub GetMonitorResolution()

 Dim lngHorzRes As Long

 Dim lngVertRes As Long

 ' Получение ширины и высоты изображения на мониторе

 lngHorzRes = GetSystemMetrics(SM\_CXSCREEN)

 lngVertRes = GetSystemMetrics(SM\_CYSCREEN)

 ' Отображение сообщения

 MsgBox "Текущее разрешение: " & lngHorzRes & "x" & lngVertRes

End Sub

## Что открыто в данный момент?

Листинг 3.74. Открытые файлы

Sub WorkBooksList()

 Dim book As Object

 ' Вывод имени каждой рабочей книги

 For Each book In Workbooks

 MsgBox (book.Name)

 Next

End Sub

Листинг 3.75. «Перелистывание» книги

Sub SheetsOfBook()

 Dim sheet As Object

 ' Отображение имен всех листов активной рабочей книги

 For Each sheet In ActiveWorkbook.Sheets

 MsgBox (sheet.Name)

 Next

End Sub

## Создание бегущей строки

Листинг 3.76. Создание бегущей строки

Dim intSpacesLeft As Integer ' Количество пробелов в начале строки

Sub Start()

 ' Установка начального количества пробелов

 intSpacesLeft = 10

 ' Первый вызов функции бегущей строки

 MovingString

End Sub

Sub MovingString()

 If intSpacesLeft >= 0 Then

 ' Отображение строки

 Range("A1").Value = Space(intSpacesLeft) & "Привет!"

 intSpacesLeft = intSpacesLeft - 1

 ' Указывем Excel, что данную процедуру нужно вызвать через \_

 1 секунду

 Application.OnTime Now + TimeValue("00:00:01"), "MovingString"

 End If

End Sub

## Создание бегущей картинки

Листинг 3.77. Бегущая картинка

Sub MovingImage()

 Dim i As Integer

 Dim image As Object

 ' Создание изображения (в ячейке "A1")

 With Range("A1")

 ' Формирование значения в ячейке:

 ' текст

 .Value = "Привет!"

 ' полужирный шрифт

 .Font.Bold = True

 ' цвет

 .Font.Color = RGB(233, 133, 229)

 ' размер шрифта

 .Font.Size = 16

 ' угол наклона

 .Orientation = 30

 ' Отображение текста полностью

 .EntireColumn.AutoFit

 ' Копирование в буфер обмена

 .Copy

 ' Создание самостоятельного изображения (на основе \_

 скопированных в буфер обмена данных)

 Set image = ActiveSheet.Pictures.Paste(Link:=False)

 ' Содержимое ячейки больше не нужно

 .Clear

 End With

 ' Задание начального положения изображения (левый верхний \_

 угол листа)

 With image

 .Top = 0

 .Left = 0

 End With

 MsgBox "ПУСК!"

 With image

 ' Перемещение изображения по диагонали

 For i = 0 To 100

 .Top = i

 .Left = i

 Next

 ' Удаление изображения

 .Delete

 End With

 ' Удаление ссылки на изображение

 Set image = Nothing

End Sub

## Мигающая ячейка

Листинг 3.78. Мигание ячейки

Sub BlinkingCell()

 Static intCalls As Integer ' Счетчик количества миганий

 ' Если ячейка мигала менее 10 раз, то изменим \_

 в очередной раз ее цвет

 If intCalls < 10 Then

 intCalls = intCalls + 1

 ' Определение, какой цвет необходимо установить

 If Range("A1").Interior.Color <> RGB(255, 0, 0) Then

 ' Цвет ячейки не красный, так что теперь назначим \_

 именно красный цвет

 Range("A1").Interior.Color = RGB(255, 0, 0)

 Else

 ' Назначим ячейке зеленый цвет

 Range("A1").Interior.Color = RGB(0, 255, 0)

 End If

 ' Эту процедуру необходимо вызвать через 5 секунд

 Application.OnTime Now + TimeValue("00:00:05"), "BlinkingCell"

 Else

 ' Хватит мигать

 Range("A1").Interior.ColorIndex = xlNone

 intCalls = 0

 End If

End Sub

## Вращающиеся автофигуры

Листинг 3.79. Вращение автофигур

Sub RotatingAutoShapes()

 Static fRunning As Boolean

 ' Проверка, выполняется ли уже этот макрос

 If fRunning Then

 ' При повторном запуске останавливаем все запущенные макросы

 fRunning = False

 End

 End If

 ' Укажем, что макрос запущен

 fRunning = True

 Dim cell As Range ' Рабочая ячейка

 Dim intLeftBorder As Long ' Левая граница ячейки

 Dim intRightBorder As Long ' Правая граница ячейки

 Dim intTopBorder As Long ' Верхняя граница ячейки

 Dim intBottomBorder As Long ' Нижняя граница ячейки

 Dim alngVertSpeed(1 To 2) As Long ' Массивы со значениями

 Dim alngHorzSpeed(1 To 2) As Long ' горизонтальной и вертикальной

 ' составляющих скоростей фигур

 Dim ashShapes(1 To 2) As Shape ' Массив перемещаемых автофигур

 Dim i As Integer

 ' Заполнение массива автофигур

 Set ashShapes(1) = ActiveSheet.shapes(1)

 Set ashShapes(2) = ActiveSheet.shapes(2)

 ' Заполнение массива скоростей:

 ' для первой фигуры

 alngVertSpeed(1) = 3

 alngHorzSpeed(1) = 3

 ' для второй фигуры

 alngVertSpeed(2) = 4

 alngHorzSpeed(2) = 4

 ' Получение границ рабочей ячейки

 Set cell = Range("B2")

 intLeftBorder = cell.Left

 intRightBorder = cell.Left + cell.Width

 intTopBorder = cell.Top

 intBottomBorder = cell.Top + cell.Height

 ' Выполнение вращения и перемещения фигур

 Do

 ' Изменение положения каждой автофигуры

 For i = 1 To 2

 With ashShapes(i)

 ' Контроль достижения правой границы ячейки

 If .Left + .Width + alngHorzSpeed(i) > intRightBorder Then

 ' Корректировка положения

 .Left = intRightBorder - .Width

 ' Изменение направления горизонтальной скорости \_

 на противоположное

 alngHorzSpeed(i) = -alngHorzSpeed(i)

 End If

 ' Контроль достижения левой границы ячейки

 If .Left + alngHorzSpeed(i) < intLeftBorder Then

 ' Корректировка положения

 .Left = intLeftBorder

 ' Изменение направления горизонтальной скорости \_

 на противоположное

 alngHorzSpeed(i) = -alngHorzSpeed(i)

 End If

 ' Контроль достижения нижней границы ячейки

 If .Top + .Height + alngVertSpeed(i) > intBottomBorder Then

 ' Корректировка положения

 .Top = intBottomBorder - .Height

 ' Изменение направления вертикальной скорости \_

 на противоположное

 alngVertSpeed(i) = -alngVertSpeed(i)

 End If

 ' Контроль достижения верхней границы ячейки

 If .Top + alngVertSpeed(i) < intTopBorder Then

 ' Корректировка положения

 .Top = intTopBorder

 ' Изменение направления вертикальной скорости \_

 на противоположное

 alngVertSpeed(i) = -alngVertSpeed(i)

 End If

 ' Перемещение автофигуры

 .Left = .Left + alngHorzSpeed(i)

 .Top = .Top + alngVertSpeed(i)

 ' Вращение автофигуры (изменение направления вращения \_

 происходит каждый раз при изменении направления \_

 вертикального перемещения)

 .IncrementRotation alngVertSpeed(i)

 ' Даем Excel команду обработать пользовательский ввод

 DoEvents

 End With

 Next

 Loop

End Sub

## Вызов таблицы цветов

Листинг 3.80. Отображение таблицы цветов

Sub ShowColorTable()

 Dim intColor As Integer

 ' Формирование заголовка таблицы

 Range("A1").Value = "Цвет"

 Range("B1").Value = "Значение свойства ColorIndex"

 ' Вывод таблицы

 Range("A2").Select

 For intColor = 1 To 56

 ' Окрашиваем ячейку столбца "A" в текущий цвет

 With ActiveCell.Interior

 .ColorIndex = intColor

 .Pattern = xlSolid

 .PatternColorIndex = xlAutomatic

 End With

 ' В ячейку столбца "B" вносим индекс текущего цвета

 ActiveCell.Offset(0, 1).Value = intColor

 ' Переходим на следующую строку

 ActiveCell.Offset(1, 0).Activate

 Next

 ' Покажем ячейку "A1" (начало таблицы)

 Range("A1").Select

 ActiveWindow.ScrollRow = 1

End Sub

## Создание калькулятора

Листинг 3.81. Создание калькулятора

Sub SimpleCalculator()

 Dim strExpr As String

 ' Ввод выражения

 strExpr = InputBox("Что будем считать?")

 ' Подсчет и вывод результата

 MsgBox strExpr & " = " & Application.Evaluate(strExpr)

End Sub

## Еще о создании пользовательских меню

### Меню с пользовательскими командами

Листинг 3.82. Код в модуле ЭтаКнига

Sub Workbook\_Open()

 ' Задание имени меню

 strMenuName = "MyCommandBarName"

 ' Создание меню

 CreateCustomMenu

End Sub

Sub Workbook\_BeforeClose(Cancel As Boolean)

 ' Удаление меню перед закрытием книги

 DeleteCustomMenu

End Sub

Листинг 3.83. Код в стандартном модуле

Public strMenuName As String ' Имя строки меню

Private cbrcBar As CommandBarControl

Sub CreateCustomMenu()

 Dim cbrMenu As CommandBar

 Dim cbrcMenu As CommandBarControl ' Выпадающее меню "Меню"

 Dim cbrcSubMenu As CommandBarControl ' Выпадающее меню "Дополнительно"

 ' Если уже есть пользовательское меню, то оно удаляется

 DeleteCustomMenu

 ' Создание меню вместо стандартного

 Set cbrMenu = Application.CommandBars.Add(strMenuName, msoBarTop, \_

 True, True)

 ' Создание выпадающего меню с названием "Меню"

 Set cbrcMenu = cbrMenu.Controls.Add(msoControlPopup, , , , True)

 With cbrcMenu

 .Caption = "&Меню"

 End With

 ' Создание пункта меню

 With cbrcMenu.Controls.Add(Type:=msoControlButton, \_

 Temporary:=True)

 .Caption = "&Меню1"

 .OnAction = "CallMenu1"

 End With

 ' Создание пункта меню

 With cbrcMenu.Controls.Add(Type:=msoControlButton, \_

 Temporary:=True)

 .Caption = "Меню2"

 .OnAction = "CallMenu2"

 End With

 ' Создание подменю первого уровня

 Set cbrcSubMenu = cbrcMenu.Controls.Add(Type:=msoControlPopup, \_

 Temporary:=True)

 With cbrcSubMenu

 .Caption = "Подменю1"

 .BeginGroup = True

 End With

 ' Создание пункта меню

 With cbrcMenu.Controls.Add(Type:=msoControlButton, \_

 Temporary:=True)

 .Caption = "Вкл/Выкл"

 .OnAction = "MenuOnOff"

 .Style = msoButtonIconAndCaption

 .FaceId = 463

 End With

 ' Создание пункта меню в подменю первого уровня

 With cbrcSubMenu.Controls.Add(Type:=msoControlButton, \_

 Temporary:=True)

 .Caption = "Подменю1"

 .OnAction = "CallSubMenu1"

 .Style = msoButtonIconAndCaption

 .FaceId = 2950

 .State = msoButtonDown

 End With

 ' Cоздание пункта меню в подменю первого уровня (его состояние \_

 изменяется посредством пункта "Вкл/Выкл"), для чего сохраним ссылку \_

 на созданный пункт меню

 Set cbrcBar = cbrcSubMenu.Controls.Add(Type:=msoControlButton, \_

 Temporary:=True)

 With cbrcBar

 .Caption = "Подменю2"

 .OnAction = "CallSubMenu2"

 ' Сначала меню деактивировано

 .Enabled = False

 End With

 ' Создание подменю второго уровня

 Set cbrcSubMenu = cbrcSubMenu.Controls.Add(Type:=msoControlPopup, \_

 Temporary:=True)

 With cbrcSubMenu

 .Caption = "ПодчПодменю1"

 .BeginGroup = True

 End With

 ' Cоздание пункта меню в подменю второго уровня

 With cbrcSubMenu.Controls.Add(Type:=msoControlButton, \_

 Temporary:=True)

 .Caption = "ПослМеню1"

 .OnAction = "CallLastMenu1"

 .Style = msoButtonIconAndCaption

 .FaceId = 71

 .State = msoButtonDown

 End With

 ' Cоздание пункта меню в подменю второго уровня

 With cbrcSubMenu.Controls.Add(Type:=msoControlButton, \_

 Temporary:=True)

 .Caption = "ПослМеню2"

 .OnAction = "CallLastMenu2"

 .Style = msoButtonIconAndCaption

 .FaceId = 72

 .Enabled = True

 End With

 ' Отображение меню

 cbrMenu.Visible = True

 Set cbrcSubMenu = Nothing

 Set cbrcMenu = Nothing

 Set cbrMenu = Nothing

End Sub

Sub DeleteCustomMenu()

 ' Удаление строки меню

 On Error Resume Next

 Application.CommandBars(strMenuName).Delete

 On Error GoTo 0

End Sub

Sub CallMenu1()

 ' Обработка вызова Меню1

 MsgBox "Приветствует меню 1!", vbInformation, ThisWorkbook.Name

End Sub

Sub CallMenu2()

 ' Обработка вызова Меню2

 MsgBox "Приветствует меню 2!", vbInformation, ThisWorkbook.Name

End Sub

Sub CallSubMenu1()

 ' Обработка вызова Подменю1

 MsgBox "Приветствует подменю 1!", vbInformation, ThisWorkbook.Name

End Sub

Sub CallSubMenu2()

 ' Обработка вызова Подменю2

 MsgBox "Приветствует подменю 2!", vbInformation, ThisWorkbook.Name

End Sub

Sub CallLastMenu1()

 ' Обработка вызова Последнего меню1

 MsgBox "Приветствует последнее меню 1!", vbInformation, ThisWorkbook.Name

End Sub

Sub CallLastMenu2()

 ' Обработка вызова Последнего меню2

 MsgBox "Приветствует последнее меню 2!", vbInformation, ThisWorkbook.Name

End Sub

Sub MenuOnOff()

 ' Активация или деактивация пункта "Меню-Подменю1-Подменю2"

 cbrcBar.Enabled = Not cbrcBar.Enabled

End Sub

### Замена стандартного меню пользовательским

Листинг 3.84. Создание пользовательского меню

Sub CreateMenu()

 Dim cbrMenu As CommandBar

 Dim cbrcNewMenu As CommandBarControl

 ' Удаление меню, если оно уже есть

 Call DeleteMenu

 ' Добавление строки пользовательского меню

 Set cbrMenu = CommandBars.Add(MenuBar:=True)

 With cbrMenu

 .Name = "Моя строка меню"

 .Visible = True

 End With

 ' Копирование стандартного меню "Файл"

 CommandBars("Worksheet Menu Bar").FindControl(ID:=30002).Copy \_

 CommandBars("Моя строка меню")

 ' Добавление нового меню - "Дополнительно"

 Set cbrcNewMenu = cbrMenu.Controls.Add(msoControlPopup)

 cbrcNewMenu.Caption = "&Дополнительно"

 ' Добавление команды в новое меню

 With cbrcNewMenu.Controls.Add(msoControlButton)

 .Caption = "&Восстановить обычную строку меню"

 .OnAction = "DeleteMenu"

 End With

 ' Добавление команды в новое меню

 With cbrcNewMenu.Controls.Add(Type:=msoControlButton)

 .Caption = "&Справка"

 End With

End Sub

Sub DeleteMenu()

 ' Пытаемся удалить меню (успешно, если оно ранее создано)

 On Error Resume Next

 CommandBars("Моя строка меню").Delete

 On Error GoTo 0

End Sub

## Склонение фамилии, имени и отчества

Листинг 3.85. Склонение ФИО

Public Sub PossessiveCase()

 ' Склоняем ФИО в родительный падеж

 Dim strName1 As String, strName2 As String, strName3 As String

 strName1 = dhGetName(ActiveCell, 1) ' Выделяем имя

 strName2 = dhGetName(ActiveCell, 2) ' Выделяем фамилию

 strName3 = dhGetName(ActiveCell, 3) ' Выделяем отчество

 ' Если в ячейке менее трех слов - закрытие процедуры

 If strName1 = "" Or strName2 = "" Or strName3 = "" Then Exit Sub

 ' Склоняем

 Cells(ActiveCell.Row, ActiveCell.Column) = dhPossessive( \_

 strName1, strName2, strName3)

End Sub

Public Sub DativeCase()

 ' Объявление переменных

 Dim strName1 As String, strName2 As String, strName3 As String

 strName1 = dhGetName(ActiveCell, 1)

 strName2 = dhGetName(ActiveCell, 2)

 strName3 = dhGetName(ActiveCell, 3)

 ' Если в ячейке менее трех слов - закрытие процедуры

 If Len(strName1) = 0 Or Len(strName2) = 0 Or Len(strName3) = 0 \_

 Then Exit Sub

 Cells(ActiveCell.Row, ActiveCell.Column) = dhDative( \_

 strName1, strName2, strName3)

End Sub

Function dhPossessive(strName1 As String, strName2 As String, \_

 strName3 As String) As String

 Dim fMan As Boolean

 ' Определяем, мужские ФИО или женские

 fMan = (Right(strName3, 1) = "ч")

 ' Склонение фамилии в родительный падеж

 If Len(strName1) > 0 Then

 If fMan Then

 ' Склонение мужской фамилии

 Select Case Right(strName1, 1)

 Case "о", "и", "я", "а"

 dhPossessive = strName1

 Case "й"

 dhPossessive = Mid(strName1, 1, Len(strName1) - 2) + "ого"

 Case Else

 dhPossessive = strName1 + "а"

 End Select

 Else

 ' Склонение женской фамилии

 Select Case Right(strName1, 1)

 Case "о", "и", "б", "в", "г", "д", "ж", "з", "к", "л", \_

 "м", "н", "п", "р", "с", "т", "ф", "х", "ц", "ч", \_

 "ш", "щ", "ь"

 dhPossessive = strName1

 Case "я"

 dhPossessive = Mid(strName1, 1, Len(strName1) - 2) & "ой"

 Case Else

 dhPossessive = Mid(strName1, 1, Len(strName1) - 1) & "ой"

 End Select

 End If

 dhPossessive = dhPossessive & " "

 End If

 ' Склонение имени в родительный падеж

 If Len(strName2) > 0 Then

 If fMan Then

 ' Склонение мужского имени

 Select Case Right(strName2, 1)

 Case "й", "ь"

 dhPossessive = dhPossessive & Mid(strName2, \_

 1, Len(strName2) - 1) & "я"

 Case Else

 dhPossessive = dhPossessive & strName2 & "а"

 End Select

 Else

 ' Склонение женского имени

 Select Case Right(strName2, 1)

 Case "а"

 Select Case Mid(strName2, Len(strName2) - 1, 1)

 Case "и", "г"

 dhPossessive = dhPossessive & Mid( \_

 strName2, 1, Len(strName2) - 1) & "и"

 Case Else

 dhPossessive = dhPossessive & Mid(strName2, \_

 1, Len(strName2) - 1) & "ы"

 End Select

 Case "я"

 If Mid(strName2, Len(strName2) - 1, 1) = "и" Then

 dhPossessive = dhPossessive & Mid(strName2, \_

 1, Len(strName2) - 1) & "и"

 Else

 dhPossessive = dhPossessive & Mid(strName2, \_

 1, Len(strName2) - 1) & "и"

 End If

 Case "ь"

 dhPossessive = dhPossessive & Mid(strName2, \_

 1, Len(strName2) - 1) & "и"

 Case Else

 dhPossessive = dhPossessive & strName2

 End Select

 End If

 dhPossessive = dhPossessive & " "

 End If

 ' Склонение отчества в родительный падеж

 If Len(strName3) > 0 Then

 If fMan Then

 dhPossessive = dhPossessive & strName3 & "а"

 Else

 dhPossessive = dhPossessive & Mid(strName3, 1, \_

 Len(strName3) - 1) & "ы"

 End If

 End If

End Function

Function dhDative(strName1 As String, strName2 As String, \_

 strName3 As String) As String

 Dim fMan As Boolean

 ' Определяем, мужские ФИО или женские

 fMan = (Right(strName3, 1) = "ч")

 ' Склонение фамилии в дательный падеж

 If Len(strName1) > 0 Then

 If fMan Then

 ' Склонение мужской фамилии

 Select Case Right(strName1, 1)

 Case "о", "и", "я", "а"

 dhDative = strName1

 Case "й"

 dhDative = Mid(strName1, 1, Len(strName1) - 2) + "ому"

 Case Else

 dhDative = strName1 + "у"

 End Select

 Else

 ' Склонение женской фамилии

 Select Case Right(strName1, 1)

 Case "о", "и", "б", "в", "г", "д", "ж", "з", "к", "л", \_

 "м", "н", "п", "р", "с", "т", "ф", "х", "ц", "ч", "ш", \_

 "щ", "ь"

 dhDative = strName1

 Case "я"

 dhDative = Mid(strName1, 1, Len(strName1) - 2) & "ой"

 Case Else

 dhDative = Mid(strName1, 1, Len(strName1) - 1) & "ой"

 End Select

 End If

 dhDative = dhDative & " "

 End If

 ' Склонение имени в дательный падеж

 If Len(strName2) > 0 Then

 If fMan Then

 ' Склонение мужского имени

 Select Case Right(strName2, 1)

 Case "й", "ь"

 dhDative = dhDative & Mid(strName2, 1, \_

 Len(strName2) - 1) & "ю"

 Case Else

 dhDative = dhDative & strName2 & "у"

 End Select

 Else

 ' Склонение женского имени

 Select Case Right(strName2, 1)

 Case "а", "я"

 If Mid(strName2, Len(strName2) - 1, 1) = "и" Then

 dhDative = dhDative & Mid(strName2, 1, \_

 Len(strName2) - 1) & "и"

 Else

 dhDative = dhDative & Mid(strName2, 1, \_

 Len(strName2) - 1) & "е"

 End If

 Case "ь"

 dhDative = dhDative & Mid(strName2, 1, \_

 Len(strName2) - 1) & "и"

 Case Else

 dhDative = dhDative & strName2

 End Select

 End If

 dhDative = dhDative & " "

 End If

 ' Склонение отчества в дательный падеж

 If Len(strName3) > 0 Then

 If fMan Then

 dhDative = dhDative & strName3 & "у"

 Else

 dhDative = dhDative & Mid(strName3, 1, Len(strName3) - 1) & "е"

 End If

 End If

End Function

Function dhGetName(strString As String, intNum As Integer)

 ' Функция возвращает слово с номером intNum во входной строке \_

 strString

 Dim strTemp As String

 Dim intWord As Integer

 Dim intSpace As Integer

 ' Удаление пробелов по краям строки

 strTemp = Trim(strString)

 ' Просмотр строки (до слова с нужным номером)

 For intWord = 1 To intNum - 1

 ' Поиск следующего пробела

 intSpace = InStr(strTemp, " ")

 If intSpace = 0 Then

 ' Строка закончилась

 intSpace = Len(strTemp)

 End If

 ' Строка strTemp теперь начинается со слова с номером intWord

 strTemp = Trim(Right(strTemp, Len(strTemp) - intSpace))

 Next intWord

 ' Выделение нужного слова (по пробелу после него)

 intSpace = InStr(strTemp, " ")

 If intSpace = 0 Then

 intSpace = Len(strTemp)

 End If

 dhGetName = Trim(Left(strTemp, intSpace))

End Function

## Получение информации об используемом принтере

Листинг 3.86. Информация о принтере

' Объявление API-функции

Declare Function GetProfileStringA Lib "kernel32" \_

 (ByVal lpAppName As String, ByVal lpKeyName As String, \_

 ByVal lpDefault As String, ByVal lpReturnedString As \_

 String, ByVal nSize As Long) As Long

Sub Принтер()

 Dim strFullInfo As String \* 255 ' Буфер для API-функции

 Dim strInfo As String ' Строка с полной информацией

 Dim strPrinter As String ' Название принтера

 Dim strDriver As String ' Драйвер принтера

 Dim strPort As String ' Порт принтера

 Dim strMessage As String

 Dim intPrinterEndPos As Integer

 Dim intDriverEndPos As Integer

 ' Заполнение буфера пробелами

 strFullInfo = Space(255)

 ' Получение полной информации о принтере

 Call GetProfileStringA("Windows", "Device", "", strFullInfo, 254)

 ' Удаление лишних символов из конца возвращенной строки

 ' Строка strInfo имеет формат <имя\_принтера>,<драйвер>,<порт>:

 strInfo = Trim(strFullInfo)

 ' Поиск запятых в строке (окончаний названий принтера и драйвера)

 intPrinterEndPos = Application.Find(",", strInfo, 1)

 intDriverEndPos = Application.Find(",", strInfo, intPrinterEndPos + 1)

 ' Определение названия принтера

 strPrinter = Left(strInfo, intPrinterEndPos - 1)

 ' Определение драйвера

 strDriver = Mid(strInfo, intPrinterEndPos + 1, intDriverEndPos \_

 - intPrinterEndPos - 1)

 ' Определение порта (его название заканчивается символом ":")

 strPort = Mid(strInfo, intDriverEndPos + 1, InStr(1, strInfo, ":") \_

 - intDriverEndPos - 1)

 ' Формирование информационного сообщения

 strMessage = "Принтер:" & Chr(9) & strPrinter & Chr(13)

 strMessage = strMessage & "Драйвер:" & strDriver & Chr(13)

 strMessage = strMessage & "strPort:" & Chr(9) & strPort

 ' Вывод информационного сообщения

 MsgBox strMessage, vbInformation, "Сведения о принтере по умолчанию"

End Sub

## Вывод текущей даты и времени

Листинг 3.87. Сообщение о дате и времени

Sub TimeAndDate()

 Dim strDate As String, strTime As String

 Dim strGreeting As String

 Dim strUserName As String

 Dim intSpacePos As Integer

 strDate = Format(Date, "Long Date")

 strTime = Format(Time, "Medium Time")

 ' Приветствие - в зависимости от времени суток

 If Time < TimeValue("12:00") Then

 strGreeting = "Доброе утро, "

 ElseIf Time < TimeValue("17:00") Then

 strGreeting = "Добрый день, "

 Else

 strGreeting = "Добрый вечер, "

 End If

 ' В приветствие добавляется имя текущего пользователя

 strUserName = Application.UserName

 intSpacePos = InStr(1, strUserName, " ", 1)

 ' Управление ситуацией, когда в имени нет пробела

 If intSpacePos = 0 Then intSpacePos = Len(strUserName)

 strGreeting = strGreeting & Left(strUserName, intSpacePos)

 ' Вывод на экран информационного сообщения о дате и времени

 MsgBox strDate & vbCrLf & strTime, vbOKOnly, strGreeting

End Sub

## Автоматическое создание документов Word на основе табличных данных Excel

Листинг 3.88. Создание документов Word на основе таблицы Excel

Sub ReportToWord()

 Dim intReportCount As Integer ' Количество сообщений

 Dim strForWho As String ' Получатель сообщения

 Dim strSum As String ' Сумма за товар

 Dim strProduct As String ' Название товара

 Dim strOutFileName As String ' Имя файла для сохранения сообщения

 Dim strMessage As String ' Текст дополнительного сообщения

 Dim rgData As Range ' Обрабатываемые ячейки

 Dim objWord As Object

 Dim i As Integer

 ' Создание объекта Word

 Set objWord = CreateObject("Word.Application")

 ' Информация с рабочего листа

 Set rgData = Range("A1")

 strMessage = Range("E6")

 ' Просмотр записей на листе Лист1

 intReportCount = Application.CountA(Range("A:A"))

 For i = 1 To intReportCount

 ' Динамические сообщения в строке состояния

 Application.StatusBar = "Создание сообщения " & i

 ' Назначение данных переменным

 strForWho = rgData.Cells(i, 1).Value

 strProduct = rgData.Cells(i, 2).Value

 strSum = Format(rgData.Cells(i, 3).Value, "#,000")

 ' Имя файла для сохранения отчета

 strOutFileName = ThisWorkbook.path & "\" & strForWho & ".doc"

 ' Передача команд в Word

 With objWord

 .Documents.Add

 With .Selection

 ' Заголовок сообщения

 .Font.Size = 14

 .Font.Bold = True

 .ParagraphFormat.Alignment = 1

 .TypeText Text:="О Т Ч Е Т"

 ' Дата

 .TypeParagraph

 .TypeParagraph

 .Font.Size = 12

 .ParagraphFormat.Alignment = 0

 .Font.Bold = False

 .TypeText Text:="Дата:" & vbTab & \_

 Format(Date, "mmmm d, yyyy")

 ' Получатель сообщения

 .TypeParagraph

 .TypeText Text:="Кому: менеджеру " & vbTab & strForWho

 ' Отправитель

 .TypeParagraph

 .TypeText Text:="От:" & vbTab & Application.UserName

 ' Сообщение

 .TypeParagraph

 .TypeParagraph

 .TypeText strMessage

 .TypeParagraph

 .TypeParagraph

 ' Название товара

 .TypeText Text:="Продано товара:" & vbTab & strProduct

 .TypeParagraph

 ' Сумма за товар

 .TypeText Text:="На сумму:" & vbTab & \_

 Format(strSum, "$#,##0")

 End With

 ' Сохранение документа

 .ActiveDocument.SaveAs FileName:=strOutFileName

 End With

 Next i

 ' Удаление объекта Word

 objWord.Quit

 Set objWord = Nothing

 ' Обновление строки состояния

 Application.StatusBar = False

 ' Вывод на экран информационного сообщения

 MsgBox intReportCount & " заметки создано и сохранено в папке " \_

 & ThisWorkbook.path

End Sub

## Создание списка панелей инструментов и контекстных меню

Листинг 3.89. Список панелей инструментов и контекстных меню

Sub ListOfMenues()

 Dim intRow As Integer ' Хранит текущую строку

 Dim cbrBar As CommandBar

 ' Очистка всех ячеек текущего листа

 Cells.Clear

 intRow = 1 ' Начинаем запись с первой строки

 ' Просматриваем список панелей инструментов и меню \_

 и записываем информацию о каждом элементе в таблицу

 For Each cbrBar In CommandBars

 ' Порядковый номер

 Cells(intRow, 1) = cbrBar.Index

 ' Название

 Cells(intRow, 2) = cbrBar.Name

 ' Тип

 Select Case cbrBar.Type

 Case msoBarTypeNormal

 Cells(intRow, 3) = "Панель инструментов"

 Case msoBarTypeMenuBar

 Cells(intRow, 3) = "Строка меню"

 Case msoBarTypePopup

 Cells(intRow, 3) = "Контекстное меню"

 End Select

 ' Встроенный элемент или созданный пользователем

 Cells(intRow, 4) = cbrBar.BuiltIn

 ' Переходим на следующую строку

 intRow = intRow + 1

 Next

End Sub

## Создание списка пунктов главного меню Excel

Листинг 3.90. Список содержимого главного меню

Sub ListOfMenues()

 Dim intRow As Integer ' Текущая строка, куда идет запись

 Dim cbrcMenu As CommandBarControl ' Главное меню

 Dim cbrcSubMenu As CommandBarControl ' Подменю

 Dim cbrcSubSubMenu As CommandBarControl ' Подменю в подменю

 ' Очищаем ячейки текущего листа

 Cells.Clear

 ' Начинаем запись с первой строки

 intRow = 1

 ' Просматриваем все элементы строки меню

 On Error Resume Next ' Игнорируем ошибки

 For Each cbrcMenu In CommandBars(1).Controls

 ' Просматриваем элементы выпадающего меню cbrcMenu

 For Each cbrcSubMenu In cbrcMenu.Controls

 ' Просматриваем элементы подменю cbrcSubMenu

 For Each cbrcSubSubMenu In cbrcSubMenu.Controls

 ' Выводим название главного меню

 Cells(intRow, 1) = cbrcMenu.Caption

 ' Выводим название подменю

 Cells(intRow, 2) = cbrcSubMenu.Caption

 ' Выводим название вложенного подменю

 Cells(intRow, 3) = cbrcSubSubMenu.Caption

 ' Переходим на следующую строку

 intRow = intRow + 1

 Next cbrcSubSubMenu

 Next cbrcSubMenu

 Next cbrcMenu

End Sub

## Создание списка пунктов контекстных меню

Листинг 3.91. Список содержимого контекстных меню

Sub ListOfContextMenues()

 Dim intRow As Long

 Dim intControl As Integer

 Dim cbrBar As CommandBar

 ' Очистка ячеек активного листа

 Cells.Clear

 ' Начинаем вывод с первой строки

 intRow = 1

 ' Просмотр списка контекстных меню и вывод информации о них

 For Each cbrBar In CommandBars

 If cbrBar.Type = msoBarTypePopup Then

 ' Порядковый номер

 Cells(intRow, 1) = cbrBar.Index

 ' Название

 Cells(intRow, 2) = cbrBar.Name

 ' Просмотр всех элементов контекстного меню и вывод \_

 названий этих элементов в ячейки текущей строки

 For intControl = 1 To cbrBar.Controls.Count

 Cells(intRow, intControl + 2) = \_

 cbrBar.Controls(intControl).Caption

 Next intControl

 ' Переход на следующую строку таблицы

 intRow = intRow + 1

 End If

 Next cbrBar

 ' Делаем ширину ячеек таблицы оптимальной для просмотра

 Cells.EntireColumn.AutoFit

End Sub

## Отображение панели инструментов при определенном условии

Листинг 3.92. Код в модуле рабочего листа

Sub Worksheet\_SelectionChange(ByVal Target As Excel.Range)

 ' Проверка условия отображения

 If Union(Target, Range("A1:D5")).Address = \_

 Range("A1:D5").Address Then

 ' Условие выполнено - можно показывать панель

 CommandBars("AutoSense").Visible = True

 Else

 ' Условие не выполнено - панель нужно скрыть

 CommandBars("AutoSense").Visible = False

 End If

End Sub

Листинг 3.93. Код в стандартном модуле

Sub CreatePanel()

 Dim cbrBar As CommandBar

 Dim button As CommandBarButton

 Dim i As Integer

 ' Удаление одноименной панели (при ее наличии)

 On Error Resume Next

 CommandBars("AutoSense").Delete

 On Error GoTo 0

 ' Создание панели инструментов

 Set cbrBar = CommandBars.Add

 ' Создание кнопок и их настройка

 For i = 1 To 4

 Set button = cbrBar.Controls.Add(msoControlButton)

 With button

 .OnAction = "ButtonClick" & i

 .FaceId = i + 37

 End With

 Next i

 cbrBar.Name = "AutoSense"

End Sub

Sub ButtonClick3()

 ' Перемещение вниз

 On Error Resume Next

 ActiveCell.Offset(1, 0).Activate

End Sub

Sub ButtonClick1()

 ' Перемещение вверх

 On Error Resume Next

 ActiveCell.Offset(-1, 0).Activate

End Sub

Sub ButtonClick2()

 ' Перемещение вправо

 On Error Resume Next

 ActiveCell.Offset(0, 1).Activate

End Sub

Sub ButtonClick4()

 ' Перемещение влево

 On Error Resume Next

 ActiveCell.Offset(0, -1).Activate

End Sub

## Скрытие и отображение панелей инструментов

Листинг 3.94. Управление отображением панелей инструментов

Sub HidePanels()

 Dim cbrBar As CommandBar

 Dim intRow As Integer ' Номер текущей строки листа

 ' Отключение обновления экрана

 Application.ScreenUpdating = False

 ' Подготовка к сохранению

 Cells.Clear

 ' Скрытие видимых панелей и сохранение их названий

 intRow = 1 ' Запись имен с первой строки

 For Each cbrBar In CommandBars

 If cbrBar.Type = msoBarTypeNormal Then

 If cbrBar.Visible Then

 cbrBar.Visible = False

 Cells(intRow, 1) = cbrBar.Name

 intRow = intRow + 1

 End If

 End If

 Next

 ' Включение обновления экрана

 Application.ScreenUpdating = True

End Sub

Sub ShowPanels()

 Dim cell As Range ' Текущая ячейка листа

 ' Отключение обновления экрана

 Application.ScreenUpdating = False

 ' Отображение скрытых панелей

 On Error Resume Next

 For Each cell In Range("A:A").SpecialCells( \_

 xlCellTypeConstants)

 CommandBars(cell.Value).Visible = True

 Next cell

 ' Включение обновления экрана

 Application.ScreenUpdating = True

End Sub

## Создание меню на основе данных рабочего листа

Листинг 3.95. Код в модуле ЭтаКнига

Sub Workbook\_Open()

 ' Создание меню

 Call CreateCustomMenu

End Sub

Sub Workbook\_BeforeClose(Cancel As Boolean)

 ' Удаление меню перед закрытием книги

 Call DeleteCustomMenu

End Sub

Листинг 3.96. Код в стандартном модуле

Sub CreateMenu()

 Dim sheet As Worksheet ' Лист с описанием меню

 Dim intRow As Integer ' Считываемая строка

 Dim cbrpBar As CommandBarPopup ' Выпадающее меню

 Dim objNewItem As Object ' Элемент меню cbrpBar

 Dim objNewSubItem As Object ' Элемент подменю objNewItem

 Dim intMenuLevel As Integer ' Уровень вложенности пункта меню

 Dim strCaption As String ' Название пункта меню

 Dim strAction As String ' Макрос пункта меню

 Dim fIsDevider As Boolean ' Нужен разделитель

 Dim intNextLevel As Integer ' Уровень вложенности следующего \_

 пункта меню

 Dim strFaceID As String ' Номер значка пункта меню

 ' Расположение данных для меню

 Set sheet = ThisWorkbook.Sheets("ЛистМеню")

 ' Удаление одноименного меню (при его наличии)

 Call DeleteMenu

 ' Данные считываем со второй строки

 intRow = 2

 ' Добавление меню

 Do Until IsEmpty(sheet.Cells(intRow, 1))

 ' Считываем информацию о пункте меню

 With sheet

 ' Уровень вложенности

 intMenuLevel = .Cells(intRow, 1)

 ' Название

 strCaption = .Cells(intRow, 2)

 ' Название макроса для меню

 strAction = .Cells(intRow, 3)

 ' Нужен ли разделитель перед меню?

 fIsDevider = .Cells(intRow, 4)

 ' Номер стандартного значка (если значок нужен)

 strFaceID = .Cells(intRow, 5)

 ' Уровень вложенности следующего меню

 intNextLevel = .Cells(intRow + 1, 1)

 End With

 ' Создаем меню в зависимости от уровня его вложенности

 Select Case intMenuLevel

 Case 1

 ' Создаем меню

 Set cbrpBar = Application.CommandBars(1). \_

 Controls.Add(Type:=msoControlPopup, \_

 Before:=strAction, \_

 Temporary:=True)

 cbrpBar.Caption = strCaption

 Case 2

 ' Создаем элемент меню

 If intNextLevel = 3 Then

 ' Следующий элемент вложен в создаваемый, то есть \_

 создаем раскрывающееся подменю

 Set objNewItem = \_

 cbrpBar.Controls.Add(Type:=msoControlPopup)

 Else

 ' Создаем команду меню

 Set objNewItem = \_

 cbrpBar.Controls.Add(Type:=msoControlButton)

 objNewItem.OnAction = strAction

 End If

 ' Установка названия нового пункта меню

 objNewItem.Caption = strCaption

 ' Установка значка нового пункта меню (если нужно)

 If strFaceID <> "" Then

 objNewItem.FaceId = strFaceID

 End If

 ' Если нужно, то добавим разделитель

 If fIsDevider Then

 objNewItem.BeginGroup = True

 End If

 Case 3

 ' Создание элемента подменю

 Set objNewSubItem = \_

 objNewItem.Controls.Add(Type:=msoControlButton)

 ' Установка его названия

 objNewSubItem.Caption = strCaption

 ' Назначение макроса (или команды)

 objNewSubItem.OnAction = strAction

 ' Установка значка (если нужно)

 If strFaceID <> "" Then

 objNewSubItem.FaceId = strFaceID

 End If

 ' Если нужно, то добавим разделитель

 If fIsDevider Then

 objNewSubItem.BeginGroup = True

 End If

 End Select

 ' Переход на следующую строку таблицы

 intRow = intRow + 1

 Loop

End Sub

Sub DeleteMenu()

 Dim sheet As Worksheet ' Лист с описанием меню

 Dim intRow As Integer ' Считываемая строка

 Dim strCaption As String ' Название меню

 Set sheet = ThisWorkbook.Sheets("ЛистМеню")

 ' Данные начинаются со второй строки

 intRow = 2

 ' Считываем данные, пока есть значения в столбце "A", \_

 и удаляем созданные ранее меню (с уровнем вложенности 1)

 On Error Resume Next

 Do Until IsEmpty(sheet.Cells(intRow, 1))

 If sheet.Cells(intRow, 1) = 1 Then

 strCaption = sheet.Cells(intRow, 2)

 Application.CommandBars(1).Controls(strCaption).Delete

 End If

 intRow = intRow + 1

 Loop

 On Error GoTo 0

End Sub

## Создание контекстного меню

Листинг 3.97. Код в модуле рабочего листа

Sub Worksheet\_BeforeRightClick(ByVal Target As Excel.Range, \_

 Cancel As Boolean)

 ' Проверка, попадает ли выделенная ячейка в диапазон

 If Union(Target.Range("A1"), Range("A2:D5")).Address = \_

 Range("A2:D5").Address Then

 ' Показываем свое контекстное меню

 CommandBars("MyContextMenu").ShowPopup

 Cancel = True

 End If

End Sub

Листинг 3.98. Код в модуле ЭтаКнига

Sub Workbook\_Open()

 ' Создание контекстного меню при открытии книги

 Call CreateCustomContextMenu

End Sub

Sub Workbook\_BeforeClose(Cancel As Boolean)

 ' Удаление меню при закрытии книги

 Call DeleteCustomContextMenu

End Sub

Листинг 3.99. Код в стандартном модуле

Sub CreateCustomContextMenu()

 ' Удаление одноименного меню

 Call DeleteCustomContextMenu

 ' Создание меню

 With CommandBars.Add("MyContextMenu", msoBarPopup, , True).Controls

 ' Создание и настройка кнопок меню

 ' Кнопка "Числовой формат"

 With .Add(msoControlButton)

 .Caption = "&Числовой формат..."

 .OnAction = "ShowFormatNumber"

 .FaceId = 1554

 End With

 ' Кнопка "Выравнивание"

 With .Add(msoControlButton)

 .Caption = "&Выравнивание..."

 .OnAction = "ShowFormatAlignment"

 .FaceId = 217

 End With

 ' Кнопка "Шрифт"

 With .Add(msoControlButton)

 .Caption = "&Шрифт..."

 .OnAction = "ShowFormatFont"

 .FaceId = 291

 End With

 ' Кнопка "Границы"

 With .Add(msoControlButton)

 .Caption = "&Границы..."

 .OnAction = "ShowFormatBorder"

 .FaceId = 149

 .BeginGroup = True

 End With

 ' Кнопка "Узор"

 With .Add(msoControlButton)

 .Caption = "&Узор..."

 .OnAction = "ShowFormatPatterns"

 .FaceId = 1550

 End With

 ' Кнопка "Защита"

 With .Add(msoControlButton)

 .Caption = "&Защита..."

 .OnAction = "ShowFormatProtection"

 .FaceId = 2654

 End With

 End With

End Sub

Sub DeleteCustomContextMenu()

 ' Удаление меню

 On Error Resume Next

 CommandBars("MyContextMenu").Delete

End Sub

Sub ShowFormatNumber()

 ' Число

 Application.Dialogs(xlDialogFormatNumber).Show

End Sub

Sub ShowFormatAlignment()

 ' Выравнивание

 Application.Dialogs(xlDialogAlignment).Show

End Sub

Sub ShowFormatFont()

 ' Шрифт

 Application.Dialogs(xlDialogFormatFont).Show

End Sub

Sub ShowFormatBorder()

 ' Граница

 Application.Dialogs(xlDialogBorder).Show

End Sub

Sub ShowFormatPatterns()

 ' Вид (Узор)

 Application.Dialogs(xlDialogPatterns).Show

End Sub

Sub ShowFormatProtection()

 ' Защита

 Application.Dialogs(xlDialogCellProtection).Show

End Sub

## Просмотр содержимого папки

Листинг 3.100. Просмотр содержимого папки

' Объявление API-функции для отображения стандартного окна \_

 просмотра папок

Declare Function SHBrowseForFolder Lib "shell32.dll" \_

 Alias "SHBrowseForFolderA" (lpBrowseInfo As BROWSEINFO) As Long

' Объявление API-функции для преобразования данных, возвращаемых \_

 функцией SHBrowseForFolder, в строку

Declare Function SHGetPathFromIDList Lib "shell32.dll" \_

 Alias "SHGetPathFromIDListA" (ByVal pidl As Long, ByVal \_

 pszPath As String) As Long

' Структура используется функцией SHBrowseForFolder

Type BROWSEINFO

 hwndOwner As Long ' Родительское окно (для диалога)

 pidlRoot As Long ' Корневая папка для просмотра

 strDisplayName As String

 strTitle As String ' Заголовок окна

 ulFlags As Long ' Флаги для окна

 ' Следующие три параметра в VBA не используются

 lpfn As Long

 lParam As Long

 iImage As Long

End Type

Sub BrowseFolder()

 Dim strPath As String ' Папка, список файлов которой выводится

 Dim strFile As String

 Dim intRow As Long ' Текущая строка таблицы

 ' Выбор папки

 strPath = dhBrowseForFolder()

 If strPath = "" Then Exit Sub

 If Right(strPath, 1) <> "\" Then strPath = strPath & "\"

 ' Оформление заголовка отчета

 ActiveSheet.Cells.ClearContents

 ActiveSheet.Cells(1, 1) = "Имя файла"

 ActiveSheet.Cells(1, 2) = "Размер"

 ActiveSheet.Cells(1, 3) = "Дата/время"

 ActiveSheet.Range("A1:C1").Font.Bold = True

 ' Просмотр объектов в папке...

 ' Первый объект папки

 strFile = Dir(strPath, 7)

 intRow = 2

 Do While strFile <> ""

 ' Запись в столбец "A" имени файла

 ActiveSheet.Cells(intRow, 1) = strFile

 ' Запись в столбец "B" размера файла

 ActiveSheet.Cells(intRow, 2) = FileLen(strPath & strFile)

 ' Запись в столбец "C" времени изменения файла

 ActiveSheet.Cells(intRow, 3) = FileDateTime(strPath & strFile)

 ' Следующий объект папки

 strFile = Dir

 intRow = intRow + 1

 Loop

End Sub

Function dhBrowseForFolder() As String

 Dim biBrowse As BROWSEINFO

 Dim strPath As String

 Dim lngResult As Long

 Dim intLen As Integer

 ' Заполнение полей структуры BROWSEINFO

 ' Корневая папка - Рабочий стол

 biBrowse.pidlRoot = 0&

 ' Заголовок окна

 biBrowse.strTitle = "Выбор папки"

 ' Тип возвращаемой папки

 biBrowse.ulFlags = &H1

 ' Вывод стандартного окна просмотра папок

 lngResult = SHBrowseForFolder(biBrowse)

 ' Обработка результата работы окна

 If lngResult Then

 ' Получение пути (по возвращенным данным)

 strPath = Space$(512)

 If SHGetPathFromIDList(ByVal lngResult, ByVal strPath) Then

 ' Строка пути заканчивается символом Chr(0)

 intLen = InStr(strPath, Chr$(0))

 ' Выделение и возврат пути

 dhBrowseForFolder = Left(strPath, intLen - 1)

 Else

 ' Не удалось получить путь

 dhBrowseForFolder = ""

 End If

 Else

 ' Пользователь нажал кнопку "Отмена"

 dhBrowseForFolder = ""

 End If

End Function

Листинг 3.101. Просмотр содержимого папки с указанием полного пути к файлам

' Объявление API-функции для отображения стандартного окна \_

 просмотра папок

Declare Function SHBrowseForFolder Lib "shell32.dll" \_

 Alias "SHBrowseForFolderA" (lpBrowseInfo As BROWSEINFO) As Long

' Объявление API-функции для преобразования данных, возвращаемых \_

 функцией SHBrowseForFolder, в строку

Declare Function SHGetPathFromIDList Lib "shell32.dll" \_

 Alias "SHGetPathFromIDListA" (ByVal pidl As Long, ByVal \_

 pszPath As String) As Long

' Структура используется функцией SHBrowseForFolder

Type BROWSEINFO

 hwndOwner As Long ' Родительское окно (для диалога)

 pidlRoot As Long ' Корневая папка для просмотра

 strDisplayName As String

 strTitle As String ' Заголовок окна

 ulFlags As Long ' Флаги для окна

 ' Следующие три параметра в VBA не используются

 lpfn As Long

 lParam As Long

 iImage As Long

End Type

Sub BrowseFolder1()

 Dim strPath As String ' Папка, список файлов которой выводится

 Dim strFile As String

 Dim intRow As Long ' Текущая строка таблицы

 ' Выбор папки

 strPath = dhBrowseForFolder()

 If strPath = "" Then Exit Sub

 If Right(strPath, 1) <> "\" Then strPath = strPath & "\"

 ' Оформление заголовка отчета

 ActiveSheet.Cells.ClearContents

 ActiveSheet.Cells(1, 1) = "Имя файла"

 ActiveSheet.Cells(1, 2) = "Размер"

 ActiveSheet.Cells(1, 3) = "Дата/время"

 ActiveSheet.Range("A1:C1").Font.Bold = True

 ' Просмотр объектов в папке...

 ' Первый объект папки

 strFile = Dir(strPath, 7)

 intRow = 2

 Do While strFile <> ""

 ' Запись в столбец "A" имени файла

 ActiveSheet.Cells(intRow, 1) = strPath & strFile

 ' Запись в столбец "B" размера файла

 ActiveSheet.Cells(intRow, 2) = FileLen(strPath & strFile)

 ' Запись в столбец "C" времени изменения файла

 ActiveSheet.Cells(intRow, 3) = FileDateTime(strPath & strFile)

 ' Следующий объект папки

 strFile = Dir

 intRow = intRow + 1

 Loop

End Sub

Function dhBrowseForFolder() As String

 Dim biBrowse As BROWSEINFO

 Dim strPath As String

 Dim lngResult As Long

 Dim intLen As Integer

 ' Заполнение полей структуры BROWSEINFO

 ' Корневая папка - Рабочий стол

 biBrowse.pidlRoot = 0&

 ' Заголовок окна

 biBrowse.strTitle = "Выбор папки"

 ' Тип возвращаемой папки

 biBrowse.ulFlags = &H1

 ' Выводим стандартное окно просмотра папок

 lngResult = SHBrowseForFolder(biBrowse)

 ' Обработка результата работы окна

 If lngResult Then

 ' Получение пути (по возвращенным данным)

 strPath = Space$(512)

 If SHGetPathFromIDList(ByVal lngResult, ByVal strPath) Then

 ' Строка пути заканчивается символом Chr(0)

 intLen = InStr(strPath, Chr$(0))

 ' Выделение и возврат пути

 dhBrowseForFolder = Left(strPath, intLen - 1)

 Else

 ' Не удалось получить путь

 dhBrowseForFolder = ""

 End If

 Else

 ' Пользователь нажал кнопку "Отмена" в окне

 dhBrowseForFolder = ""

 End If

End Function

## Получение информации о состоянии дисков

Листинг 3.102. Просмотр информации о дисках компьютера

Sub DrivesInfo()

 Dim objFileSysObject As Object ' Объект для работы \_

 с файловой системой

 Dim objDrive As Object ' Анализируемый диск

 Dim intRow As Integer ' Заполняемая строка листа

 ' Создание объекта для работы с файловой системой

 Set objFileSysObject = CreateObject("Scripting.FileSystemObject")

 ' Очистка листа

 Cells.Clear

 ' Запись с первой строки

 intRow = 1

 ' Запись на лист информации о дисках компьютера

 On Error Resume Next

 For Each objDrive In objFileSysObject.Drives

 ' Буква диска

 Cells(intRow, 1) = objDrive.DriveLetter

 ' Готовность

 Cells(intRow, 2) = objDrive.IsReady

 ' Тип диска

 Select Case objDrive.DriveType

 Case 0

 Cells(intRow, 3) = "Неизвестно"

 Case 1

 Cells(intRow, 3) = "Съемный"

 Case 2

 Cells(intRow, 3) = "Жесткий"

 Case 3

 Cells(intRow, 3) = "Сетевой"

 Case 4

 Cells(intRow, 3) = "CD-ROM"

 Case 5

 Cells(intRow, 3) = "RAM"

 End Select

 ' Метка диска

 Cells(intRow, 4) = objDrive.VolumeName

 ' Общий размер

 Cells(intRow, 5) = objDrive.TotalSize

 ' Свободное место

 Cells(intRow, 6) = objDrive.AvailableSpace

 intRow = intRow + 1

 Next

End Sub

## Расчет среднего арифметического

Листинг 3.103. Расчет среднего значения

Sub CalculateAverage()

 Dim strFistCell As String

 Dim strLastCell As String

 Dim strFormula As String

 ' Условия закрытия процедуры

 If ActiveCell.Row = 1 Then Exit Sub

 ' Определение положения первой и последней ячеек для расчета

 strFistCell = ActiveCell.Offset(-1, 0).End(xlUp).Address

 strLastCell = ActiveCell.Offset(-1, 0).Address

 ' Формула для расчета среднего значения

 strFormula = "=AVERAGE(" & strFistCell & ":" & strLastCell & ")"

 ' Ввод формулы в текущую ячейку

 ActiveCell.Formula = strFormula

End Sub

## Вывод списка доступных шрифтов

Листинг 3.104. Список шрифтов

Sub ListOfFonts()

 Dim cbrcFonts As CommandBarControl

 Dim cbrBar As CommandBar

 Dim i As Integer

 ' Получение доступа к списку шрифтов (элемент управления в виде \_

 раскрывающегося списка на панели инструментов "Форматирование")

 Set cbrcFonts = Application.CommandBars("Formatting"). \_

 FindControl(ID:=1728)

 If cbrcFonts Is Nothing Then

 ' Панель "Форматирование" не открыта - откроем ее

 Set cbrBar = Application.CommandBars.Add

 Set cbrcFonts = cbrBar.Controls.Add(ID:=1728)

 End If

 ' Подготовка к выводу шрифтов (очистка ячеек)

 Range("A:A").ClearContents

 ' Вывод списка шрифтов в столбец "A" текущего листа

 For i = 0 To cbrcFonts.ListCount - 1

 Cells(i + 1, 1) = cbrcFonts.List(i + 1)

 Next i

 ' Закрытие панели инструментов "Форматирование", если мы были \_

 вынуждены ее открывать

 On Error Resume Next

 cbrBar.Delete

End Sub

## Обработка сразу нескольких внешних файлов

Листинг 3.105. Обработка текстовых файлов

Sub ImportTextFiles()

 Dim fsSearch As FileSearch

 Dim strFileName As String

 Dim strPath As String

 Dim i As Integer

 ' Задание пути и возможного имени файла

 strFileName = ThisWorkbook.path & "\"

 strPath = "text??.txt"

 ' Создание объекта FileSearch

 Set fsSearch = Application.FileSearch

 ' Настройка объекта для поиска

 With fsSearch

 ' Маска для поиска

 .LookIn = strFileName

 ' Путь для поиска

 .FileName = strPath

 ' Поиск всех файлов, удовлетворяющих маске

 .Execute

 ' Выход, если файлы не существуют

 If .FoundFiles.Count = 0 Then

 MsgBox "Файлы не обнаружены"

 Exit Sub

 End If

 End With

 ' Обработка найденных файлов

 For i = 1 To fsSearch.FoundFiles.Count

 Call ImportTextFile(fsSearch.FoundFiles(i))

 Next i

End Sub

Sub ImportTextFile(FileName As String)

 ' Импорт файла

 Workbooks.OpenText FileName:=FileName, \_

 Origin:=xlWindows, \_

 StartRow:=1, \_

 DataType:=xlFixedWidth, \_

 FieldInfo:= \_

 Array(Array(0, 1), Array(3, 1), Array(12, 1))

 ' Ввод формул суммирования

 Range("D1").Value = "A"

 Range("D2").Value = "B"

 Range("D3").Value = "C"

 Range("E1:E3").Formula = "=COUNTIF(B:B,D1)"

 Range("F1:F3").Formula = "=SUMIF(B:B,D1,C:C)"

End Sub

## Запуск таблицы символов из Excel

Листинг 3.106. Вызов таблицы символов

Sub ShowSymbolTable()

 On Error Resume Next

 ' Запуск Charmap.exe - таблицы символов

 Shell "Charmap.exe", vbNormalFocus

 If Err <> 0 Then

 MsgBox "Невозможно запустить таблицу символов.", vbCritical

 End If

End Sub

Листинг 3.107. Таблица символов

' Декларация API-функций:

' для открытия процесса

Declare Function OpenProcess Lib "kernel32" \_

 (ByVal dwDesiredAccess As Long, ByVal bInheritHandle As Long, \_

 ByVal dwProcessId As Long) As Long

' для получения кода завершения процесса

Declare Function GetExitCodeProcess Lib "kernel32" \_

 (ByVal hProcess As Long, lpExitCode As Long) As Long

' для закрытия процесса

Declare Function CloseHandle Lib "kernel32" \_

 (hProcess) As Long

Sub ShowSymbolTable1()

 Dim lProcessID As Long

 Dim hProcess As Long

 Dim lExitCode As Long

 On Error Resume Next

 ' Запуск таблицы символов (Charman.exe). Функция возвращает \_

 идентификатор созданного процесса

 lProcessID = Shell("Charmap.exe", 1)

 If Err <> 0 Then

 MsgBox "Нельзя запустить Charman.exe", vbCritical, "Ошибка"

 Exit Sub

 End If

 ' Открытие процесса по идентификатору (lProcessID). Функция \_

 возвращает дескриптор процесса (handle)

 hProcess = OpenProcess(&H400, False, lProcessID)

 ' Ждем, пока процесс завершится, для этого периодически \_

 получаем код завершения процесса (пока Charman.exe исполняется, \_

 функция GetExitCodeProcess возвращает &H103)

 Do

 GetExitCodeProcess hProcess, lExitCode

 DoEvents

 Loop While lExitCode = &H103

 ' Закрытие процесса

 CloseHandle (hProcess)

 ' Вывод на экран информационного сообщения

 MsgBox "Charmap.exe завершает свою работу"

End Sub

## Создание раскрывающегося списка

Листинг 3.108. Создание панели со списком

Sub CreatePanel()

 Dim i As Integer

 On Error Resume Next

 ' Удаление одноименной панели (если есть)

 CommandBars("Список месяцев").Delete

 On Error GoTo 0

 ' Создание панели "Список месяцев"

 With CommandBars.Add

 .Name = "Список месяцев"

 ' Создание списка месяцев

 With .Controls.Add(Type:=msoControlDropdown)

 ' Настройка (имя, макрос, стиль)

 .Caption = "DateDD"

 .OnAction = "SetMonth"

 .Style = msoButtonAutomatic

 ' Добавление в список названий месяцев

 For i = 1 To 12

 .AddItem Format(DateSerial(1, i, 1), "mmmm")

 Next i

 ' Выделение первого месяца

 .ListIndex = 1

 End With

 ' Показываем созданную панель

 .Visible = True

 End With

End Sub

Sub SetMonth()

 ' Перенос названия выделенного месяца в ячейку

 On Error Resume Next

 With CommandBars("Список месяцев").Controls("DateDD")

 ActiveCell.Value = .List(.ListIndex)

 End With

End Sub

## Добавление команды в меню

### Добавление команды в меню Сервис

Листинг 3.109. Новая команда в меню Сервис

Sub AddMenuItem()

 Dim cbrpMenu As CommandBarPopup

 ' Удаление аналогичной команды (при ее наличии)

 Call DeleteMenuItem

 ' Получение доступа к меню "Сервис"

 Set cbrpMenu = CommandBars(1).FindControl(ID:=30007)

 If cbrpMenu Is Nothing Then

 ' Не удалось получить доступ

 MsgBox "Невозможно добавить элемент."

 Exit Sub

 Else

 ' Добавление новой команды в меню

 With cbrpMenu.Controls.Add(Type:=msoControlButton)

 ' Название команды

 .Caption = "Очистить в&се, кроме формул"

 ' Значок

 .FaceId = 348

 ' Сочетание клавиш (только надпись на кнопке)

 .ShortcutText = "Ctrl+Shift+C"

 ' Сопоставленный макрос

 .OnAction = "ExecuteCommand"

 ' Добавление разделителя перед командой

 .BeginGroup = True

 End With

 End If

 ' Сопоставление с макросом сочетания клавиш Ctrl+Shift+C

 Application.MacroOptions \_

 Macro:="ExecuteCommand", \_

 HasShortcutKey:=True, \_

 ShortcutKey:="C"

End Sub

Sub ExecuteCommand()

 ' Очистка содержимого всех ячеек (кроме формул)

 On Error Resume Next

 Cells.SpecialCells(xlCellTypeConstants, 23).ClearContents

End Sub

Sub DeleteMenuItem()

 ' Удаление команды из меню

 On Error Resume Next

 CommandBars(1).FindControl(ID:=30007). \_

 Controls("Очистить в&се, кроме формул").Delete

End Sub

### Добавление команды в меню Вид

Листинг 3.110. Код в стандартном модуле

Dim AppObject As New Class1

Sub AddCommand()

 Dim cbrpBar As CommandBarPopup

 ' Удаление аналогичной команды (при ее наличии)

 Call DeleteCommand

 ' Получение доступа к меню "Вид"

 Set cbrpBar = CommandBars(1).FindControl(ID:=30004)

 If cbrpBar Is Nothing Then

 ' Не удалось получить доступ к меню

 MsgBox "Невозможно добавить элемент меню."

 Exit Sub

 Else

 ' Добавление команды

 With cbrpBar.Controls.Add(Type:=msoControlButton)

 .Caption = "&Линии сетки"

 .OnAction = "GhangeGridlinesState"

 End With

 End If

 ' Даем объекту AppObject обрабатывать события

 Set AppObject.AppEvents = Application

End Sub

Sub DeleteCommand()

 ' Удаление каманды из меню (если она там есть)

 On Error Resume Next

 CommandBars(1).FindControl(ID:=30004). \_

 Controls("&Линии сетки").Delete

End Sub

Sub GhangeGridlinesState()

 ' Изменение состояния отображения линий сетки \_

 на противоположное (если нет - покажем, если есть - скроем)

 If TypeName(ActiveSheet) = "Worksheet" Then

 ActiveWindow.DisplayGridlines = \_

 Not ActiveWindow.DisplayGridlines

 ' Установка или снятие флажка в меню

 Call CheckGridlines

 End If

End Sub

Sub CheckGridlines()

 Dim button As CommandBarButton

 On Error Resume Next

 ' Поиск команды "Линии сетки" в меню "Вид"

 Set button = CommandBars(1).FindControl(ID:=30004). \_

 Controls("&Линии сетки")

 ' Изменение состояния флажка на противоположное

 If ActiveWindow.DisplayGridlines Then

 ' Установка

 button.State = msoButtonDown

 Else

 ' Снятие

 button.State = msoButtonUp

 End If

End Sub

Листинг 3.111. Код в модуле класса

Public WithEvents AppEvents As Application

' Обработка события активации листа

Sub AppEvents\_SheetActivate(ByVal Sh As Object)

 Call CheckGridlines

End Sub

' Обработка события активации книги

Sub AppEvents\_WorkbookActivate(ByVal Wb As Excel.Workbook)

 Call CheckGridlines

End Sub

' Обработка события активации окна

Sub AppEvents\_WindowActivate \_

 (ByVal Wb As Workbook, ByVal Wn As Window)

 Call CheckGridlines

End Sub

# Глава 4. Трюки и эффекты с помощником

## Мультфильм с помощником в главной роли

Листинг 4.1. «Танцующий» помощник

Sub RunAssistantDance()

 Static intAction As Integer

 ' Заставляем помощника выполнять действие (всего 16)

 DoAssistantAction intAction

 intAction = intAction + 1

 If intAction < 16 Then

 ' Следующее действие через 3 секунды

 Application.OnTime Time + TimeValue("00:00:3"), \_

 "RunAssistantDance"

 End If

End Sub

Sub DoAssistantAction(intAction As Integer)

 Dim astAssistant As Assistant

 Set astAssistant = Application.Assistant

 ' Помещаем помощника в центр активного окна

 astAssistant.Top = Application.ActiveWindow.Top \_

 + Application.ActiveWindow.Height / 2

 astAssistant.Left = Application.ActiveWindow.Left \_

 + Application.ActiveWindow.Width / 2

 ' Показываем помощника

 astAssistant.On = True

 astAssistant.Visible = True

 ' Показываем заданное параметром intAction действие

 Select Case intAction

 Case 0

 astAssistant.Animation = msoAnimationAppear

 Case 1

 astAssistant.Animation = msoAnimationCheckingSomething

 Case 2

 astAssistant.Animation = msoAnimationBeginSpeaking

 Case 3

 astAssistant.Animation = msoAnimationCharacterSuccessMajor

 Case 4

 astAssistant.Animation = msoAnimationEmptyTrash

 Case 5

 astAssistant.Animation = msoAnimationGestureDown

 Case 5

 astAssistant.Animation = msoAnimationGestureLeft

 Case 6

 astAssistant.Animation = msoAnimationGestureRight

 Case 7

 astAssistant.Animation = msoAnimationGestureUp

 Case 8

 astAssistant.Animation = msoAnimationGetArtsy

 Case 9

 astAssistant.Animation = msoAnimationGetAttentionMajor

 Case 10

 astAssistant.Animation = msoAnimationGetAttentionMinor

 Case 11

 astAssistant.Animation = msoAnimationGetTechy

 Case 12

 astAssistant.Animation = msoAnimationGetWizardy

 Case 13

 astAssistant.Animation = msoAnimationGoodbye

 Case 14

 astAssistant.Animation = msoAnimationGreeting

 Case 15

 astAssistant.Animation = msoAnimationDisappear

 End Select

End Sub

## Дополнение помощника текстом, заголовком, кнопкой и значком

Листинг 4.2. Настройка помощника

Sub AssistantMessage()

 Dim strTitle As String ' Заголовок сообщения

 Dim strMessage As String ' Текст сообщения

 ' Содержимое заголовка и текста в окне помощника

 strTitle = "Спрашивайте - ответим"

 strMessage = "{cf 249}{ul 1} Руки мыли{ul 0}?" \_

 & vbCr & "{cf 6} Не забыть обновить антивирус!"

 ' Настраиваем помощника

 With Application.Assistant

 ' Включаем и показываем помощника

 .On = True

 .Visible = True

 ' Создаем окно сообщения

 With .NewBalloon

 .BalloonType = msoBalloonTypeButtons

 ' Кнопка "ОК" в окне помощника

 .button = msoButtonSetOK

 ' Значок в окне помощника

 .Icon = msoIconAlert

 ' Заголовок в окне помощника

 .Heading = strTitle

 ' Текст в окне помощника

 .Text = strMessage

 ' Отображение окна

 .Show

 End With

 End With

End Sub

## Новые параметры помощника

Листинг 4.3. Новые параметры помощника

Sub AssistantCheckboxes()

 Dim i As Integer

 Dim strMessage As String

 With Assistant

 ' Включение и отображение помощника

 .On = True

 .Visible = True

 ' Создание окна сообщения

 With .NewBalloon

 ' Настройка окна...

 ' Тип окна

 .BalloonType = msoBalloonTypeButtons

 ' Заголовок

 .Heading = "Выберите страну"

 ' Добавление флажков

 .CheckBoxes(1).Text = "Россия"

 .CheckBoxes(2).Text = "США"

 .CheckBoxes(3).Text = "Южная Африка"

 .button = msoButtonSetOkCancel

 ' Отображение окна

 If .Show = msoBalloonButtonOK Then

 ' Вывод информационного окна в зависимости \_

 от установленных флажков

 For i = 1 To 3

 If .CheckBoxes(i).Checked Then

 strMessage = strMessage & \_

 .CheckBoxes(i).Text & vbCr

 End If

 Next

 ' Отображение окна сообщения (имеется в виду \_

 стандартное окно)

 If Len(strMessage) = 0 Then

 MsgBox "No choice."

 Else

 MsgBox strMessage

 End If

 End If

 End With

 End With

End Sub

## Использование помощника для выбора цвета заливки

Листинг 4.4. Выбор цвета заливки рабочего листа

Sub AssistantChooseColor()

 Dim intChoise As Integer

 With Assistant

 ' Включение и отображение помощника

 .On = True

 .Visible = True

 With .NewBalloon

 ' Настройка окна...

 ' Тип

 .BalloonType = msoBalloonTypeButtons

 ' Заголовок

 .Heading = "Какой нужен цвет?"

 ' Первый цвет

 .Labels(1).Text = "Красный"

 ' Второй цвет

 .Labels(2).Text = "Желтый"

 ' Третий цвет

 .Labels(3).Text = "Зеленый"

 ' Тип кнопок

 .button = msoButtonSetNone

 ' Оображение окна

 intChoise = .Show

 ' Информационное сообщение о выбранном цвете

 MsgBox "Выбран: " & .Labels(intChoise).Text

 End With

 End With

 ' Настройка цветов ячеек (присвоение выбранного цвета)

 Select Case intChoise

 Case 1

 ' Красный цвет

 ActiveSheet.Cells.Interior.Color = RGB(255, 0, 0)

 Case 2

 ' Желтый цвет

 ActiveSheet.Cells.Interior.Color = RGB(255, 255, 0)

 Case 3

 ' Зеленый цвет

 ActiveSheet.Cells.Interior.Color = RGB(0, 255, 0)

 End Select

End Sub

# Глава 5. Эксперименты с диаграммами

## Построение диаграммы с помощью макроса

Листинг 5.1. Макрос построения диаграммы

Sub CreateChart()

 ' Создание и настройка диаграммы

 With Charts.Add

 ' Данные из первого листа

 .SetSourceData Source:=Worksheets(1).Range("A1:E4")

 ' Заголовок

 .HasTitle = True

 .ChartTitle.Text = "Выручка по магазинам"

 ' Активизируем диаграмму

 .Activate

 End With

End Sub

Листинг 5.2. Построение внедренной диаграммы

Sub CreateEmbeddedChart()

 ' Создание и настройка внедренной диаграммы

 With Worksheets(1).ChartObjects.Add(100, 60, 250, 200)

 ' Объемная диаграмма

 .Chart.ChartType = xl3DArea

 ' Источник данных

 .Chart.SetSourceData Source:=Worksheets(1).Range("A1:E4")

 End With

End Sub

Листинг 5.3. Создание диаграммы на основе выделенных данных

Sub CreateCharOnSelection()

 ' Создание диаграммы (с заданием положения на листе)

 With ActiveSheet.ChartObjects.Add( \_

 Selection.Left + Selection.Width, \_

 Selection.Top + Selection.Height, 300, 200).Chart

 ' Тип диаграммы

 .ChartType = xlColumnClustered

 ' Источник данных - выделение

 .SetSourceData Source:=Selection, PlotBy:=xlColumns

 ' Без легенды

 .HasLegend = False

 ' Без заголовка

 .HasTitle = True

 .ChartTitle.Characters.Text = "Выручка за период"

 ' Выделение диаграммы

 .Parent.Select

 End With

End Sub

## Сохранение диаграммы в отдельном файле

Листинг 5.4. Сохранение диаграммы

Sub SaveChart()

 ' Сохранение выделенной диаграммы в файл

 If ActiveChart Is Nothing Then

 ' Нет выделенных диаграмм

 MsgBox "Выделите диаграмму"

 Else

 ' Сохранение...

 ActiveChart.Export ActiveWorkbook.path & "\Диаграмма.gif", "GIF"

 End If

End Sub

Листинг 5.5. Сохранение диаграммы под указанным именем

Sub InteractiveSaveChart()

 Dim strFileName As String ' Имя файла для сохранения

 ' Проверка, выделена ли диаграмма

 If ActiveChart Is Nothing Then

 ' Нет выделенных диаграмм

 MsgBox "Выделите диаграмму"

 Else

 ' Выбор файла для сохранения

 strFileName = Application.GetSaveAsFilename( \_

 ActiveChart.Name & ".gif", "Файлы GIF (\*.gif), \*.gif", 1, \_

 "Сохранить диаграмму в формате GIF")

 ' Проверка, выбран ли файл

 If strFileName <> "" Then

 ' Сохранение выделенной диаграммы в файл

 ActiveChart.Export strFileName, "GIF"

 End If

 End If

End Sub

## Построение и удаление диаграммы нажатием одной кнопки

Листинг 5.6. Быстрое построение и удаление диаграммы

Sub CreateChart()

 ' Создание диаграммы

 Charts.Add

 ' Параметры диаграммы

 ' Тип диаграммы

 ActiveChart.ChartType = xlLineMarkers

 ' Заголовок

 ActiveChart.SetSourceData Range("B1:E2"), xlRows

 ActiveChart.Location xlLocationAsObject, Name

 ' Остальные параметры

 With ActiveChart

 ' Заголовок

 .HasTitle = True

 .ChartTitle.Characters.Text = Name

 ' Заголовок оси категорий

 .Axes(xlCategory, xlPrimary).HasTitle = True

 .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text \_

 = Sheets(Name).Range("A1").Value

 ' Заголовок оси значений

 .Axes(xlValue, xlPrimary).HasTitle = True

 .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text \_

 = Sheets(Name).Range("A2").Value

 ' Отображение легенды

 .HasLegend = False

 .HasDataTable = True

 .DataTable.ShowLegendKey = True

 ' Настройка отображения сетки

 With .Axes(xlCategory)

 .HasMajorGridlines = True

 .HasMinorGridlines = False

 End With

 With .Axes(xlValue)

 .HasMajorGridlines = True

 .HasMinorGridlines = False

 End With

 End With

End Sub

Sub DeleteChart()

 ' Удаление диаграммы

 ActiveSheet.ChartObjects.Delete

End Sub

## Вывод списка диаграмм в отдельном окне

Листинг 5.7. Внедренные диаграммы

Sub ShowSheetCharts()

 Dim strMessage As String

 Dim i As Integer

 ' Формирование списка диаграмм

 For i = 1 To ActiveSheet.ChartObjects.Count

 strMessage = strMessage & ActiveSheet.ChartObjects(i).Name \_

 & vbNewLine

 Next i

 ' Отображение списка

 MsgBox strMessage

End Sub

Листинг 5.8. Перечень рабочих листов, содержащих обычные диаграммы

Sub ShowBookCharts()

 Dim crt As chart

 Dim strMessage As String

 ' Формирование списка диаграмм

 For Each crt In ActiveWorkbook.Charts

 strMessage = strMessage & crt.Name & vbNewLine

 Next

 ' Отображение списка

 MsgBox strMessage

End Sub

## Применение случайной цветовой палитры

Листинг 5.9. Случайная цветовая палитра

Sub RandomChartColors()

 Dim intGradientStyle As Integer, intGradientVariant As Integer

 Dim i As Integer

 ' Проверка, выделена ли диаграмма

 If ActiveChart Is Nothing Then Exit Sub

 ' Изменение оформления всех категорий

 For i = 1 To ActiveChart.SeriesCollection.Count

 With ActiveChart.SeriesCollection(i)

 ' Вид градиентной заливки (случайный)

 intGradientStyle = Int(Rnd \* 7) + 1

 If intGradientStyle = 6 Then intGradientStyle = 1

 If intGradientStyle = 7 Then

 intGradientVariant = Int(Rnd \* 2) + 1

 Else

 intGradientVariant = Int(Rnd \* 4) + 1

 End If

 ' Применение градиента

 .Fill.TwoColorGradient Style:=intGradientStyle, \_

 Variant:=intGradientVariant

 ' Установка случайных цветов фона и обводки (используются \_

 для градиента)

 .Fill.ForeColor.SchemeColor = Int(Rnd \* 57) + 1

 .Fill.BackColor.SchemeColor = Int(Rnd \* 57) + 1

 End With

 Next i

End Sub

## Эффект прозрачности диаграммы

Листинг 5.10. Эффект прозрачности диаграммы

Sub TransparentChart()

 Dim shpShape As Shape

 Dim dblColor As Double

 Dim srSerie As Series

 Dim intBorderLineStyle As Integer

 Dim intBorderColorIndex As Integer

 Dim intBorderWeight As Integer

 ' Проверка, есть ли выделенная диаграмма

 If ActiveChart Is Nothing Then Exit Sub

 ' Изменение отображения каждой категории

 For Each srSerie In ActiveChart.SeriesCollection

 If (srSerie.ChartType = xlColumnClustered Or \_

 srSerie.ChartType = xlColumnStacked Or \_

 srSerie.ChartType = xlColumnStacked100 Or \_

 srSerie.ChartType = xlBarClustered Or \_

 srSerie.ChartType = xlBarStacked Or \_

 srSerie.ChartType = xlBarStacked100) Then

 ' Сохранение прежнего цвета категории

 dblColor = srSerie.Interior.Color

 ' Сохранение стиля линий

 intBorderLineStyle = srSerie.Border.LineStyle

 ' Цвет границы

 intBorderColorIndex = srSerie.Border.ColorIndex

 ' Толщина линий границы

 intBorderWeight = srSerie.Border.Weight

 ' Создание автофигуры

 Set shpShape = ActiveSheet.shapes.AddShape \_

 (msoShapeRectangle, 1, 1, 100, 100)

 With shpShape

 ' Закрашиваем нужным цветом

 .Fill.ForeColor.RGB = dblColor

 ' Делаем прозрачной

 .Fill.Transparency = 0.4

 ' Убираем линии

 .Line.Visible = msoFalse

 End With

 ' Копируем автофигуру в буфер обмена

 shpShape.CopyPicture Appearance:=xlScreen, \_

 Format:=xlPicture

 ' Вставляем автофигуру в изображения столбцов \_

 категории и настраиваем

 With srSerie

 ' Собственно вставка

 .Paste

 ' Возвращаем на место толщину линий

 .Border.Weight = intBorderWeight

 ' Стиль линий

 .Border.LineStyle = intBorderLineStyle

 ' Цвет границы

 .Border.ColorIndex = intBorderColorIndex

 End With

 ' Автофигура больше не нужна

 shpShape.Delete

 End If

 Next srSerie

End Sub

## Построение диаграммы на основе данных нескольких рабочих листов

Листинг 5.11. Одновременное создание нескольких диаграмм

Sub ManyCharts()

 Dim intTop As Long, intLeft As Long

 Dim intHeight As Long, intWidth As Long

 Dim sheet As Worksheet

 Dim lngFirstRow As Long ' Первая строка с данными

 Dim intSerie As Integer ' Текущая категория диаграммы

 Dim strErrorSheets As String ' Список листов, для которых \_

 не удалось построить диаграммы

 intTop = 1 ' Верхняя точка первой диаграммы

 intLeft = 1 ' Левая точка каждой диаграммы

 intHeight = 180 ' Высота каждой диаграммы

 intWidth = 300 ' Ширина каждой диаграммы

 ' Постоение диаграммы для каждого листа, кроме текущего

 For Each sheet In ActiveWorkbook.Worksheets

 If sheet.Name <> ActiveSheet.Name Then

 ' Первый заполненный ряд

 lngFirstRow = 3

 ' Первая категория

 intSerie = 1

 On Error GoTo DiagrammError

 ' Добавление и настройка диаграммы

 With ActiveSheet.ChartObjects.Add \_

 (intLeft, intTop, intWidth, intHeight).Chart

 Do Until IsEmpty(sheet.Cells(lngFirstRow + intSerie, 1))

 ' Создание ряда

 .SeriesCollection.NewSeries

 ' Значения для ряда

 .SeriesCollection(intSerie).Values = \_

 sheet.Range(sheet.Cells(lngFirstRow + intSerie, 2), \_

 sheet.Cells(lngFirstRow + intSerie, 4))

 ' Диапазон данных для подписей

 .SeriesCollection(intSerie).XValues = \_

 sheet.Range("B3:D3")

 ' Название ряда (берется из столбца "A" таблицы с данными)

 .SeriesCollection(intSerie).Name = sheet.Cells( \_

 lngFirstRow + intSerie, 1)

 intSerie = intSerie + 1

 Loop

 ' Настройка внешнего вида диаграммы

 .ChartType = xl3DColumnClustered

 .ChartGroups(1).GapWidth = 20

 .PlotArea.Interior.ColorIndex = xlNone

 .ChartArea.Font.Size = 9

 ' Диаграмма с легендой

 .HasLegend = True

 ' Заголовок

 .HasTitle = True

 .ChartTitle.Characters.Text = sheet.Range("A1")

 ' Задание диапазона значений на осях

 .Axes(xlValue).MinimumScale = 0

 .Axes(xlValue).MaximumScale = 120000

 ' Стиль линий сетки (прерывистый)

 .Axes(xlValue).MajorGridlines.Border. \_

 LineStyle = xlDot

 End With

 On Error GoTo 0

 ' Сдвиг верхней точки следующей диаграммы на высоту \_

 текущей диаграммы

 intTop = intTop + intHeight

AfterError:

 End If

 Next sheet

 If strErrorSheets <> "" Then

 ' Отобразим список листов, для которых не построили диаграммы

 MsgBox "Не удалось построить диаграммы для листов:" & Chr(13) \_

 & strErrorSheets, vbExclamation

 End If

 Exit Sub

DiagrammError:

 ' Добавление в список имени листа, для которого не смогли \_

 построить диаграмму (ошибка в данных для диаграммы)

 strErrorSheets = strErrorSheets & sheet.Name & Chr(13)

 ' Удаление пустой диаграммы на текущем листе

 ActiveSheet.ChartObjects(ActiveSheet.ChartObjects.Count).Delete

 ' Продолжаем работу с другими листами

 Resume AfterError

End Sub

## Создание подписей к данным диаграммы

Листинг 5.12. Подписи к данным диаграммы

Sub ShowLabels()

 Dim rgLabels As Range ' Диапазон с подписями

 Dim chrChart As Chart ' Диаграмма

 Dim intPoint As Integer ' Точка, для которой добавляется подпись

 ' Определение диаграммы

 Set chrChart = ActiveSheet.ChartObjects(1).Chart

 ' Запрос на ввод диапазона с исходными данными

 On Error Resume Next

 Set rgLabels = Application.InputBox \_

 (prompt:="Укажите диапазон с подписями", Type:=8)

 If rgLabels Is Nothing Then Exit Sub

 On Error GoTo 0

 ' Добавление подписей

 chrChart.SeriesCollection(1).ApplyDataLabels \_

 Type:=xlDataLabelsShowValue, \_

 AutoText:=True, \_

 LegendKey:=False

 ' Просмотр диапазона и назначение подписей

 For intPoint = 1 To chrChart.SeriesCollection(1).Points.Count

 chrChart.SeriesCollection(1). \_

 Points(intPoint).DataLabel.Text = rgLabels(intPoint)

 Next intPoint

End Sub

Sub DeleteLabels()

 ' Удаление подписей диаграммы

 ActiveSheet.ChartObjects(1).Chart.SeriesCollection(1). \_

 HasDataLabels = False

End Sub

# Глава 6. Создание полезных программ

## Программа для составления кроссвордов

Листинг 6.1. Программа для составления кроссворда

Const dhcMinCol = 1 ' Номер первого столбца кроссворда

Const dhcMaxCol = 35 ' Номер последнего столбца кроссворда

Const dhcMinRow = 1 ' Номер первой строки кроссворда

Const dhcMaxRow = 35 ' Номер последней строки кроссворда

Sub Clear()

 ' Выделение и очистка всех используемых для кроссворда ячеек

 Range(Cells(dhcMinRow, dhcMinCol), \_

 Cells(dhcMaxRow, dhcMaxCol)).Select

 Selection.Clear

 ' Удаление сетки всего кроссворда

 ClearGrid

 Range("A1").Select

End Sub

Sub ClearGrid()

 ' Удаление сетки кроссворда (в выделенных ячейках)...

 ' Возврат прежнего цвета ячеек

 Selection.Interior.ColorIndex = xlNone

 ' Задание начертания границ ячеек по умолчанию

 Selection.Borders(xlDiagonalDown).LineStyle = xlNone

 Selection.Borders(xlDiagonalUp).LineStyle = xlNone

 Selection.Borders(xlEdgeLeft).LineStyle = xlNone

 Selection.Borders(xlEdgeTop).LineStyle = xlNone

 Selection.Borders(xlEdgeBottom).LineStyle = xlNone

 Selection.Borders(xlEdgeRight).LineStyle = xlNone

 Selection.Borders(xlInsideVertical).LineStyle = xlNone

 Selection.Borders(xlInsideHorizontal).LineStyle = xlNone

End Sub

Sub DrowCrosswordGrid()

 ' Процедура начертания сетки кроссворда

 ' Задание цвета всех ячеек кроссворда

 Selection.Interior.ColorIndex = 35

 ' Линии по диагонали не нужны

 Selection.Borders(xlDiagonalDown).LineStyle = xlNone

 Selection.Borders(xlDiagonalUp).LineStyle = xlNone

 ' Задание начертания границ всех диапазонов, входящих \_

 в выделение, а также границ между соседними ячейками \_

 всех диапазонов

 On Error Resume Next

 ' Левые границы

 With Selection.Borders(xlEdgeLeft)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

 ' Правые границы

 With Selection.Borders(xlEdgeRight)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

 ' Верхние границы

 With Selection.Borders(xlEdgeTop)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

 ' Нижние границы

 With Selection.Borders(xlEdgeBottom)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

 ' Вертикальные границы между ячейками

 With Selection.Borders(xlInsideVertical)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

 ' Горизонтальные границы между ячейками

 With Selection.Borders(xlInsideHorizontal)

 .LineStyle = xlContinuous

 .Weight = xlThin

 .ColorIndex = xlAutomatic

 End With

End Sub

Sub DisplayGrid()

 ' Включение сетки на листе

 ActiveWindow.DisplayGridlines = True

End Sub

Sub HideGrid()

 ' Выключение сетки на листе

 ActiveWindow.DisplayGridlines = False

End Sub

Sub AutoNumber()

 ' Нумерация клеток, являющихся началом слов

 Dim intRow As Integer ' Текущая строка

 Dim intCol As Integer ' Текущий ряд

 Dim cell As Range ' Текущая ячейка (с координатами \_

 (intRow, intCol))

 Dim fTop As Boolean ' = True, если cell имеет соседей сверху

 Dim fBottom As Boolean ' = True, если cell имеет соседей снизу

 Dim fLeft As Boolean ' = True, если cell имеет соседей слева

 Dim fRight As Boolean ' = True, если cell имеет соседей справа

 Dim intDigit As Integer ' Текущий номер слова в кроссворде

 intDigit = 1 ' Нумерация слов с 1

 ' Проходим по всем клеткам диапазона, используемого \_

 для кроссворда, сверху вниз слева направо и анализируем \_

 каждую угловую и крайнюю (левую и верхнюю) ячейки

 For intRow = dhcMinRow To dhcMaxRow

 For intCol = dhcMinCol To dhcMaxCol

 ' Текущая ячейка

 Set cell = Cells(intRow, intCol)

 ' Проверка, входит ли ячейка в кроссворд (по ее цвету)

 If cell.Interior.ColorIndex = 35 Then

 fLeft = False

 fRight = False

 fTop = False

 fBottom = False

 On Error Resume Next

 ' Определение наличия соседей у ячейки...

 ' сверху

 fTop = cell.Offset(-1, 0).Interior.ColorIndex = 35

 ' снизу

 fBottom = cell.Offset(1, 0).Interior.ColorIndex = 35

 ' слева

 fLeft = cell.Offset(0, -1).Interior.ColorIndex = 35

 ' справа

 fRight = cell.Offset(0, 1).Interior.ColorIndex = 35

 On Error GoTo 0

 ' Анализ положения ячейки

 If (Not fTop And Not fLeft) Or \_

 (Not fBottom And Not fLeft And fRight) Or \_

 (Not fLeft And fRight) Or \_

 (Not fTop And fBottom) Then

 ' Ячейка подходит для начала слова

 SetDigit intDigit, cell

 intDigit = intDigit + 1

 End If

 End If

 Next intCol

 Next intRow

End Sub

Sub SetDigit(intDigit As Integer, cell As Range)

 ' Вставка цифры intDigit в ячейку, заданную параметром cell

 cell.Value = intDigit

 ' Изменение настроек шрифта так, чтобы было похоже \_

 на настоящий кроссворд

 ' Маленький размер шрифта

 cell.Font.Size = 6

 ' Выравнивание текста по левому верхнему углу ячейки

 cell.HorizontalAlignment = xlLeft

 cell.VerticalAlignment = xlTop

End Sub

Sub ToPrint()

 ' Удаление цветовой подсветки кроссворда

 Cells.Interior.ColorIndex = xlNone

End Sub

Sub ToNumber()

 ' Закрытие первой формы и переход ко второй

 UserForm1.Hide

 UserForm2.Show

End Sub

## Игра «Минное поле»

Листинг 6.2. Код в модуле рабочего листа

Sub Worksheet\_SelectionChange(ByVal Target As Range)

 Dim intCol As Integer, intRow As Integer

 Dim intMinesAround As Integer

 Dim fInGameField As Boolean

 ' Определим, попадает ли в игровое поле выделенная ячейка

 fInGameField = (Target.Row >= 2) And (Target.Row <= 7) \_

 And (Target.Column >= 2) And (Target.Column <= 7)

 ' Обрабатываем выделение ячейки

 If Target.Value = "\*" And fInGameField Then

 ' Пользователь выделил ячейку с миной - покажем мину

 Target.Font.Color = RGB(0, 0, 0)

 Target.Interior.Color = RGB(255, 0, 0)

 ' Пользователь проиграл!

 EndGame

 ElseIf fInGameField Then

 ' Пользователь выделил пустую ячейку. Оформим эту ячейку

 Target.Interior.Color = RGB(0, 0, 255)

 Target.Font.Color = RGB(0, 255, 0)

 Target.Font.Size = 16

 ' Подсчитаем количество мин рядом с ячейкой (вокруг ячейки)

 For intCol = Target.Column - 1 To Target.Column + 1

 For intRow = Target.Row - 1 To Target.Row + 1

 If Target.Worksheet.Cells(intRow, intCol).Value = "\*" \_

 Then

 ' Нашли очередную мину

 intMinesAround = intMinesAround + 1

 End If

 Next

 Next

 ' Отображение количества мин

 Target.Value = intMinesAround

 End If

End Sub

Листинг 6.3. Код в стандартном модуле

Sub NewGame()

 ' Начало новой игры

 ' Подготовим поле для игры

 InitGame

 Dim intRow As Integer, intCol As Integer

 Dim intMinesCount As Integer ' Количество мин

 ' Расставляем мины (то есть в случайные ячейки помещаем \_

 значения "\*" и делаем цвет шрифта таким же, как цвет \_

 фона этих ячеек)

 For intMinesCount = 1 To 10

 ' Строка для мины (от 2 до 7)

 intRow = Int((6 \* Rnd) + 1) + 1

 ' Столбец для мины (от 2 до 7)

 intCol = Int((6 \* Rnd) + 1) + 1

 ' Ставим мину, если ячейка пустая

 If Cells(intRow, intCol) <> "\*" Then

 Cells(intRow, intCol).Font.Color = \_

 Cells(intRow, intCol).Interior.Color

 Cells(intRow, intCol).Value = "\*"

 Else

 ' В данной ячейке мина есть - продолжим поиск ячеек

 intMinesCount = intMinesCount - 1

 End If

 Next

 ' Вывод информации о количестве мин в строку состояния

 Application.StatusBar = "Количество мин " & intMinesCount

End Sub

Sub InitGame()

 ' Раскраска (оформление) листа перед началом игры

 Dim intRow As Integer, intCol As Integer

 ' Цвет фона всех ячеек

 Cells.Interior.Color = RGB(0, 200, 75)

 ' Цвет шрифта всех ячеек

 Cells.Font.Color = RGB(0, 0, 0)

 ' Размер шрифта

 Cells.Font.Size = 18

 ' Все надписи - по центру

 Cells.HorizontalAlignment = xlCenter

 ' Всем ячейкам игрового поля назначим особый цвет

 For intRow = 2 To 7

 For intCol = 2 To 7

 Cells(intRow, intCol).Interior.Color = RGB(200, 200, 200)

 Cells(intRow, intCol).Value = ""

 Next

 Next

End Sub

Sub EndGame()

 ' Завершение игры (поражение)

 Dim intRow As Integer, intCol As Integer

 ' Покажем все мины. Для этого сделаем цвет шрифта всех ячеек \_

 черным (ведь во всех ячейках с минами "\*" цвет шрифта и цвет \_

 заливки одинаковы)

 For intRow = 2 To 7

 For intCol = 2 To 7

 If Cells(intRow, intCol).Value = "\*" Then

 Cells(intRow, intCol).Font.Color = RGB(0, 0, 0)

 End If

 Next

 Next

 MsgBox "Проигрыш"

End Sub

## Игра «Угадай животное»

Листинг 6.4. Игра «Угадай животное»

Sub StartGame()

 Dim intLastRow As Integer ' Номер строки для вставки записей

 Dim intRow As Integer ' Номер текущей строки

 Dim intYesRow As Integer ' Номер строки, из которой брать \_

 данные при утвердительном ответе

 Dim intNoRow As Integer ' Номер строки, из которой брать \_

 данные при отрицательном ответе

 Dim strText As String ' Строка с вопросом или названием \_

 животного

 Dim strNewName As String ' Строка с названием нового животного

 Dim strNewQuestion As String ' Строка с новым вопросом

 Dim intRes As Integer

 ' Начало игры

 MsgBox "Начнем игру. Задумайте животное.", vbOKOnly, \_

 "Задумайте животное"

 ' Определение номера ряда для вставки записей. \_

 intLastRow-1 - номер последнего ряда, содержащего данные

 intLastRow = Worksheets("Data").Range("D1").Value + 1

 ' Данные в таблице идут с первого ряда

 intRow = 1

 Do While intRow < intLastRow

 ' Текст вопроса или название животного из столбца "A"

 strText = Worksheets("Data").Cells(intRow, 1).Value

 ' Номер ряда, из которого брать данные при утвердительном \_

 ответе, берем из столбца "B"

 intYesRow = Worksheets("Data").Cells(intRow, 2).Value

 ' Номер ряда, из которого брать данные при отрицательном \_

 ответе, берем из столбца "C"

 intNoRow = Worksheets("Data").Cells(intRow, 3).Value

 If intYesRow > 0 Then

 ' В строке strText содержится вопрос. Зададим его

 intRes = MsgBox(strText, vbYesNo, "Вопрос")

 If intRes = vbYes Then

 ' Переходим по утвердительному ответу

 intRow = intYesRow

 Else

 ' Переходим по отрицательному ответу

 intRow = intNoRow

 End If

 Else

 ' Альтернативы закончились. В строке strText - название \_

 животного. Спросим, его ли загадали

 intRes = MsgBox("Это " & strText & "?", vbYesNo, "Вопрос")

 If intRes = vbYes Then

 ' Животное угадано

 MsgBox "Угадано! Спасибо за игру!", vbOKOnly, \_

 "Игра завершена"

 Exit Do

 Else

 ' Животное не угадали, но данные уже занкончились. \_

 Нужно пополнить наши данные, чтобы отличать животное \_

 с названием strText от загаданного

 ' Ввод названия нового животного

 strNewName = InputBox("Сдаюсь. Кто это?", \_

 "Напечатайте название животного")

 If strNewName <> "" Then

 ' Ввод вопроса, по которому отличать животных

 strNewQuestion = InputBox("Задайте вопрос, по " & \_

 "которому можно отличить '" & strNewName & \_

 "' от '" & strText & "'", "Напечатайте вопрос")

 If strNewQuestion <> "" Then

 ' Определение, какое из животных соответствует \_

 утвердительному ответу на вопрос

 intRes = MsgBox("Правильный ответ на ваш " & \_

 "вопрос - " & strNewName & "'", vbYesNo, \_

 "Какой ответ на вопрос?")

 ' Добавление в таблицу названия нового животного

 Worksheets("Data").Cells(intLastRow, 1). \_

 Value = strNewName

 ' Перемещения названия животного, которое было \_

 ранее, в конец таблицы

 Worksheets("Data").Cells(intLastRow + 1, 1). \_

 Value = strText

 ' Замена названия этого животного вопросом

 Worksheets("Data").Cells(intRow, 1). \_

 Value = strNewQuestion

 ' Корректировка номеров строк для перехода \_

 в зависимости от того, какое животное является \_

 правильным ответом на введенный пользователем вопрос

 If intRes = vbYes Then

 ' Новое животное - правильный ответ

 Worksheets("Data").Cells(intRow, 2). \_

 Value = intLastRow

 Worksheets("Data").Cells(intRow, 3). \_

 Value = intLastRow + 1

 Else

 ' Бывшее ранее животное - правильный ответ

 Worksheets("Data").Cells(intRow, 2). \_

 Value = intLastRow + 1

 Worksheets("Data").Cells(intRow, 3). \_

 Value = intLastRow

 End If

 ' Сохраним номер строки для добавления записей

 Worksheets("Data").Range("D1").Value = \_

 intLastRow + 2

 End If

 End If

 ' Игра завершена. Таблица дополнена

 MsgBox "Спасибо за игру!", vbOKOnly, "Игра завершена"

 Exit Do

 End If

 End If

 Loop

End Sub

## Расчет на основании ячеек определенного цвета

Листинг 6.5. Код в стандартном модуле

Const dhcSum As Integer = 0

Const dhcAvg As Integer = 1

Const dhcMax As Integer = 2

Const dhcMin As Integer = 3

Const dhcCount As Integer = 4

Const dhcSumPlus As Integer = 5

Const dhcSumMinus As Integer = 6

Const dhcCountFull As Integer = 7

Const dhcCountNotNull As Integer = 8

Const dhcCountPlus As Integer = 9

Const dhcCountMinus As Integer = 10

Sub CalcColors()

 ' Отображение формы

 Load frmColorCalc

 frmColorCalc.Show

End Sub

Public Function ColorCalc(strRange As String, \_

 lngColor As Long, fBackBolor As Boolean, \_

 intMode As Integer, Optional fAbsence As Boolean) As Double

 ' Операции над ячейками с установленным цветом шрифта \_

 или заливки

 Dim rgData As Range ' Диапазон ячеек для расчетов

 Dim i As Integer

 Dim Values() As Variant ' Массив со значениями для расчета

 Dim intCount As Integer ' Количество значений в массиве

 Dim cell As Range

 Dim varOut As Variant ' В этой переменной хранятся \_

 результаты промежуточных подсчетов \_

 и окончательный результат

 Set rgData = Range(strRange)

 ReDim Values(1 To rgData.Count)

 ' Просматриваются все ячейки входного диапазона. Значения тех из них, \_

 цвет которых удовлетворяет условию, записываются в массив Values

 For Each cell In rgData.Cells

 ' Если нужно суммировать по заливке:

 If fBackBolor = True Then

 ' Включение ячейки в сумму в зависимости от цвета \_

 заливки и фильтра

 If fAbsence Then

 ' Если ячейка имеет заданный цвет, то она не включается \_

 в вычисления

 If cell.Interior.Color <> lngColor Then

 intCount = intCount + 1

 Values(intCount) = cell.Value

 End If

 Else

 ' Если ячейка имеет заданный цвет, то она включается \_

 в вычисления

 If cell.Interior.Color = lngColor Then

 intCount = intCount + 1

 Values(intCount) = cell.Value

 End If

 End If

 ' В противном случае - суммируется по шрифту

 Else

 ' Включение ячейки в сумму в зависимости \_

 от ее цвета и фильтра

 If fAbsence Then

 ' Если ячейка имеет заданный цвет, то она не включается \_

 в вычисления

 If cell.Font.Color <> lngColor Then

 intCount = intCount + 1

 Values(intCount) = cell.Value

 End If

 Else

 ' Если ячейка имеет заданный цвет, то она включается \_

 в вычисления

 If cell.Font.Color = lngColor Then

 intCount = intCount + 1

 Values(intCount) = cell.Value

 End If

 End If

 End If

 Next cell

 ' Выполнение над собранными значениями операции, заданной в intMode

 For i = 1 To intCount

 Select Case intMode

 Case dhcSum, dhcAvg

 ' Подсчет суммы значений

 varOut = varOut + Values(i)

 Case dhcSumPlus

 ' Подсчет суммы положительных значений

 If Values(i) > 0 Then varOut = varOut + Values(i)

 Case dhcSumMinus

 ' Посчет суммы отрицательных значений

 If Values(i) < 0 Then varOut = varOut + Values(i)

 Case dhcMax

 ' Нахождение максимального значения

 If Values(i) > varOut Then varOut = Values(i)

 Case dhcMin

 ' Нахождение минимального значения

 If i = LBound(Values) Then varOut = Values(i)

 If Values(i) < varOut Then varOut = Values(i)

 Case dhcCount

 ' Подсчет количества значений

 varOut = varOut + 1

 Case dhcCountFull

 ' Подсчет количества заполненных ячеек

 If Not IsEmpty(Values(i)) Then varOut = varOut + 1

 Case dhcCountNotNull

 ' Подсчет количества пустых ячеек

 If Not IsEmpty(Values(i)) And Values(i) <> 0 Then \_

 varOut = varOut + 1

 Case dhcCountPlus

 ' Подсчет количества положительных значений

 If Values(i) > 0 Then varOut = varOut + 1

 Case dhcCountMinus

 ' Подсчет количества отрицательных значений

 If Values(i) < 0 Then varOut = varOut + 1

 End Select

 Next i

 ' Окончательные операции для некоторых видов расчета

 If intMode = dhcAvg Then

 ' Вычисление среднего значения

 ColorCalc = varOut / intCount

 Else

 ColorCalc = varOut

 End If

End Function

Листинг 6.6. Код в модуле формы

Dim lngCurColor As Long ' Выбранный цвет, по которому \_

 идентифицировать (отбирать) ячейки

Dim intMode As Integer ' Номер типа вычисления в списке

Sub cmbApplyColor\_Click()

 If cboOtherColor.Value >= 0 Then

 ' Вычисление с использованием выбранного в списке цвета

 lngCurColor = cboOtherColor.Value

 SetColorSum

 End If

End Sub

Sub cmbColor1\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor1.BackColor

 SetColorSum

End Sub

Sub cmbColor2\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor2.BackColor

 SetColorSum

End Sub

Sub cmbColor3\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor3.BackColor

 SetColorSum

End Sub

Sub cmbColor4\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor4.BackColor

 SetColorSum

End Sub

Sub cmbColor5\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor5.BackColor

 SetColorSum

End Sub

Sub cmbColor6\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor6.BackColor

 SetColorSum

End Sub

Sub cmbColor7\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor7.BackColor

 SetColorSum

End Sub

Sub cmbColor8\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor8.BackColor

 SetColorSum

End Sub

Sub cmbColor9\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor9.BackColor

 SetColorSum

End Sub

Sub cmbColor10\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor10.BackColor

 SetColorSum

End Sub

Sub cmbColor11\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor11.BackColor

 SetColorSum

End Sub

Sub cmbColor12\_Click()

 ' Вычисление с использованием цвета нажатой кнопки

 lngCurColor = cmbColor12.BackColor

 SetColorSum

End Sub

Sub SetColorSum()

 ' Вычисление с использованием заданного цвета

 Dim strFormula As String

 ' Проверка правильности введенных диапазонов и номеров ячеек

 If txtResCell.Value = "" Then

 MsgBox "Введите адрес ячейки вставки функции", \_

 vbCritical, "Внимание!"

 txtResCell.SetFocus

 Exit Sub

 ElseIf txtRange.Value = "" Then

 MsgBox "Введите адрес диапазона суммирования", \_

 vbCritical, "Внимание!"

 txtRange.SetFocus

 Exit Sub

 End If

 ' Формирование формулы

 strFormula = "=ColorCalc(" & """" & txtRange.Value & """" \_

 & "," & lngCurColor & "," & CInt(tglType.Value) & "," \_

 & intMode & "," & CInt(chkVarify.Value) & ")"

 ' Запись формулы в ячейку

 Range(txtResCell.Value).Formula = strFormula

End Sub

Sub cmbExit\_Click()

 ' Закрытие формы

 Unload Me

End Sub

Sub cboCalcTypes\_AfterUpdate()

 ' Изменение режима вычисления - сохраним в переменной \_

 номер вычисления

 intMode = cboCalcTypes.ListIndex

End Sub

Sub cboOtherColor\_Change()

 ' Изменение выделенного цвета в списке "Другой"

 If cboOtherColor.Text <> "" Then

 ' Сохранение выбранного цвета в переменной

 lngCurColor = Val(cboOtherColor.Value)

 End If

End Sub

Sub tglType\_Click()

 ' Изменение типа идентификации ячеек

 If tglType.Value = -1 Then

 ' Идентификация по цвету заливки

 tglType.Caption = "Заливка"

 Else

 ' Идентификация по цвету шрифта

 tglType.Caption = "Шрифт"

 End If

 GetColors

End Sub

Sub txtRange\_AfterUpdate()

 ' Изменение диапазона с исходными данными - покажем \_

 кнопки с цветами, представленными в новом диапазоне

 GetColors

End Sub

Sub txtRange\_BeforeUpdate(ByVal Cancel As MSForms.ReturnBoolean)

 ' Проверка корректности данных, введенных в поле \_

 диапазона исходных данных

 Dim rgData As Range

 Dim cell As Range

 ' Проверка, введен ли диапазон данных

 If txtRange.Text = "" Then

 MsgBox "Введите адрес диапазона суммирования!", \_

 vbCritical, "Ошибка выполнения"

 Cancel = True

 End If

 If txtResCell.Text = "" Then Exit Sub

 On Error GoTo Err1

 ' Проверка отсутствия циклических ссылок (чтобы одна \_

 из входных ячеек не была одновременно и выходной)

 Set rgData = Range(txtRange.Text)

 For Each cell In rgData.Cells

 If cell.Address(False, False) = \_

 Range(txtResCell.Text).Address(False, False) Then

 ' Нашли циклическую ссылку

 MsgBox "Введите другой адрес во избежание " & \_

 "появления циклических ссылок", vbCritical, \_

 "Внимание!"

 Cancel = True

 Exit Sub

 End If

 Next cell

 Exit Sub

Err1:

 ' Обработка ошибок при работе с ячейками

 If Err.Number = 1004 Then

 MsgBox "Введите корректный адрес ячейки", vbCritical, \_

 "Ошибка ввода"

 Cancel = True

 Exit Sub

 Else

 MsgBox Err.Description, vbCritical, "Ошибка ввода"

 Cancel = True

 Exit Sub

 End If

End Sub

Sub txtResCell\_BeforeUpdate(ByVal Cancel As MSForms.ReturnBoolean)

 ' Проверка корректности данных, введенных в поле \_

 адреса выходной ячейки

 Dim rgData As Range

 Dim cell As Range

 ' Проверка, введен ли диапазон данных

 If txtRange.Text = "" Then

 MsgBox "Введите адрес диапазона суммирования!", \_

 vbCritical, "Ошибка выполнения"

 Cancel = True

 End If

 If txtResCell.Text = "" Then Exit Sub

 On Error GoTo Err1

 ' Проверка отсутствия циклических ссылок (чтобы одна \_

 из входных ячеек не была одновременно и выходной)

 Set rgData = Range(txtRange.Text)

 For Each cell In rgData.Cells

 If cell.Address(False, False) = \_

 Range(txtResCell.Text).Address(False, False) Then

 ' Нашли циклическую ссылку

 MsgBox "Введите другой адрес во избежание " & \_

 "появления циклических ссылок", vbCritical, \_

 "Внимание!"

 Cancel = True

 Exit Sub

 End If

 Next cell

 Exit Sub

Err1:

 ' Обработка ошибок при работе с ячейками

 If Err.Number = 1004 Then

 MsgBox "Введите корректный адрес ячейки", vbCritical, \_

 "Ошибка ввода"

 Cancel = True

 Exit Sub

 Else

 MsgBox Err.Description, vbCritical, "Ошибка ввода"

 Cancel = True

 Exit Sub

 End If

End Sub

Sub UserForm\_Activate()

 ' Инициализация формы при активации

 Dim intFunc As Integer

 Dim strFunc As String

 ' Заполение списка доступных операций

 cboCalcTypes.AddItem "0"

 cboCalcTypes.List(0, 1) = "Сумма"

 cboCalcTypes.AddItem "1"

 cboCalcTypes.List(1, 1) = "Среднее"

 cboCalcTypes.AddItem "2"

 cboCalcTypes.List(2, 1) = "Максимум"

 cboCalcTypes.AddItem "3"

 cboCalcTypes.List(3, 1) = "Минимум"

 cboCalcTypes.AddItem "4"

 cboCalcTypes.List(4, 1) = "Количество ячеек"

 cboCalcTypes.AddItem "5"

 cboCalcTypes.List(5, 1) = "Сумма положительных"

 cboCalcTypes.AddItem "6"

 cboCalcTypes.List(6, 1) = "Сумма отрицательных"

 cboCalcTypes.AddItem "7"

 cboCalcTypes.List(7, 1) = "Количество непустых"

 cboCalcTypes.AddItem "8"

 cboCalcTypes.List(8, 1) = "Количество непустых ненулевых"

 cboCalcTypes.AddItem "9"

 cboCalcTypes.List(9, 1) = "Количество положительных"

 cboCalcTypes.AddItem "10"

 cboCalcTypes.List(10, 1) = "Количество отрицательных"

 ' Заполнение списка дополнительных цветов

 cboOtherColor.AddItem "255"

 cboOtherColor.List(0, 1) = "Красный"

 cboOtherColor.AddItem "52479"

 cboOtherColor.List(1, 1) = "Оранжевый"

 cboOtherColor.AddItem "65535"

 cboOtherColor.List(2, 1) = "Желтый"

 cboOtherColor.AddItem "32768"

 cboOtherColor.List(3, 1) = "Зеленый"

 cboOtherColor.AddItem "16776960"

 cboOtherColor.List(4, 1) = "Голубой"

 cboOtherColor.AddItem "16711680"

 cboOtherColor.List(5, 1) = "Синий"

 cboOtherColor.AddItem "16711935"

 cboOtherColor.List(6, 1) = "Фиолетовый"

 cboOtherColor.AddItem "16777215"

 cboOtherColor.List(7, 1) = "Белый"

 cboOtherColor.AddItem "0"

 cboOtherColor.List(8, 1) = "Черный"

 If Selection.Cells.Count = 1 Then

 ' На листе есть выделенная ячейка. Определим, есть ли в этой \_

 ячейке формула с функцией ColorCalc

 intFunc = InStr(Selection.Formula, "ColorCalc(")

 If intFunc > 0 Then

 ' Формула есть, заполним поля формы для вычислений

 ' Адрес ячейки с результатом

 txtResCell.Text = Selection.Address(False, False)

 ' Выделяем аргументы функции...

 ' Номера ячеек с исходными данными

 strFunc = Mid(Selection.Formula, intFunc + 11)

 intFunc = InStr(strFunc, """")

 txtRange.Text = Left(strFunc, intFunc - 1)

 ' Тип идентификации ячеек (по шрифту или цвету)

 strFunc = Mid(strFunc, intFunc + 2)

 intFunc = InStr(strFunc, ",")

 strFunc = Mid(strFunc, intFunc + 1)

 intFunc = InStr(strFunc, ",")

 tglType.Value = Left(strFunc, intFunc - 1)

 ' Режим вычислений

 strFunc = Mid(strFunc, intFunc + 1)

 strFunc = Left(strFunc, Len(strFunc) - 1)

 intFunc = InStr(strFunc, ",")

 cboCalcTypes.Text = cboCalcTypes.List(Val(Left$( \_

 strFunc, intFunc - 1)), 1)

 strFunc = Mid(strFunc, intFunc + 1)

 chkVarify.SetFocus

 chkVarify.Value = CBool(strFunc)

 lblChoose.Visible = True

 GetColors

 Else

 ' Будем применять формулу для выделенной ячейки

 txtRange.Value = Selection.Address(False, False)

 ' В выделенной ячейке конкретная функция не задана. \_

 Выберем первую функцию в списке

 cboCalcTypes.Text = "Сумма"

 End If

 Else

 ' Будем применять формулу для выделенной ячейки

 txtRange.Value = Selection.Address(False, False)

 ' В выделенной ячейке конкретная функция не задана. \_

 Выберем первую функцию в списке

 cboCalcTypes.Text = "Сумма"

 End If

End Sub

Sub GetColors()

 ' Отображение кнопок выбора цвета окрашенными в цвета, \_

 встречающиеся среди ячеек заданного диапазона

 Dim rgCells As Range

 Dim i As Integer

 Dim intColorNumber As Integer ' Номер следующей кнопки \_

 выбора цвета

 Dim lngCurColor As Long ' Анализируемый цвет

 Dim fColorPresented As Boolean ' Кнопка с цветом \_

 lngCurColor уже существует

 Dim ctrl As Control

 Dim strCtrl As String

 Dim fBackColor As Boolean ' = True, если ячейки \_

 идентифицируются по цвету фона, \_

 = False - по цвету шрифта

 fBackColor = tglType.Value

 On Error Resume Next

 ' Скрытие всех кнопок выбора цвета

 For Each ctrl In Me.Controls

 If Left(ctrl.Name, 8) = "cmbColor" Then

 ctrl.Visible = False

 End If

 Next ctrl

 On Error GoTo ErrRange

 Set rgCells = Range(txtRange.Text)

 On Error GoTo 0

 ' Получение цвета первой ячейки

 If fBackColor = False Then

 lngCurColor = rgCells.Cells(i).Font.Color

 Else

 lngCurColor = rgCells.Cells(i).Interior.Color

 End If

 ' Назначения цвета первой ячейки первой кнопке

 cmbColor1.BackColor = lngCurColor

 cmbColor1.Visible = True

 ' Просмотр остальных ячеек и при нахождении новых цветов \_

 отображение кнопок, окрашенных в эти цвета

 intColorNumber = 2

 For i = 2 To rgCells.Cells.Count

 fColorPresented = False

 ' Получение цвета i-й ячейки

 If fBackColor = False Then

 lngCurColor = rgCells.Cells(i).Font.Color

 Else

 lngCurColor = rgCells.Cells(i).Interior.Color

 End If

 ' Проверка, отображается ли уже кнопка с таким цветом

 For Each ctrl In Me.Controls

 If Left(ctrl.Name, 8) = "cmbColor" And \_

 ctrl.Visible = True Then

 If lngCurColor = ctrl.BackColor Then

 ' Кнопка с цветом i-й ячейки уже отображается

 fColorPresented = True

 Exit For

 End If

 End If

 Next ctrl

 If Not fColorPresented Then

 ' Кнопки с цветом lngCurColor еще нет - покажем ее

 intColorNumber = intColorNumber + 1

 strCtrl = "cmbColor" & intColorNumber

 Me.Controls(strCtrl).BackColor = lngCurColor

 Me.Controls(strCtrl).Visible = True

 End If

 Next i

 Exit Sub

ErrRange:

 ' Обработка ошибок при работе с диапазоном

 If txtRange.Text = "" Then

 MsgBox "Введите адрес диапазона суммирования", \_

 vbCritical, "Внимание!"

 Else

 MsgBox "Введен некорректный адрес диапазона суммирования", \_

 vbCritical, "Ошибка!"

 End If

 ' Установка курсора в поле ввода диапазона

 txtRange.SetFocus

End Sub