УДК 004.67

СБОР ДАННЫХ ИЗ ФАЙЛОВ EXCEL И WORD

●●●●●●

COLLECT DATA FROM EXCEL AND WORD FILES

Капитонов Владимир Алексеевич
кандидат технических наук,
доцент кафедры
«Бурение нефтяных и газовых скважин»,
Самарский государственный
технический университет
KapitonovVA@gmail.com

Аннотация. Проведение анализа причин возникновения осложнений при бурении скважин требует сбора информации из различных источников, включая файлы Excel и Word. Данная задача решается с помощью интегрированной среды разработки Visual Basic for Application (VBA), встроенной в офисные приложения. В статье рассмотрено какие данные встречаются в суточных сводках по бурению. Предложен алгоритм для автоматизации сбора данных из файлов excel. Коротко рассмотрено как из кода приложения Excel можно получить доступ к объектной модели Word. Даны ссылки на литературу и статьи интернета.

Ключевые слова: структура данных, автоматизированный сбор данных, раннее связывание, позднее связывание, GetOpenFilename, заливка ячеек цветом.

Kapitonov Vladimir Alekseevich
Ph.D., Associate Professor of Oil and Gas Wells Drilling Department, Samara State Technical University
KapitonovVA@gmail.com

Annotation. An analysis of the causes of complications during well drilling requires the collection of information from various sources, including Microsoft Excel and Microsoft Word files. This problem is solved by use of an integrated development environment Visual Basic for Application (VBA), which is built into the Microsoft Office applications. The types of data found in daily drilling reports are reviewed in the article. An algorithm for automation of data collection from Microsoft Excel files is proposed. It is briefly reviewed how to access the Microsoft Word object model from the code of the Microsoft Excel application. Literature and Internet articles references are given.

Keywords: data structure, automated data collection, early binding, late binding, GetOpenFilename, cell shading.

Бурение скважин сопровождается подготовкой различной документации: проект на бурение скважины, план-программа бурения, рабочая программа по буровым растворам и рабочая документация других сервисных направлений, суточные сводки супервайзера и сервисных подрядчиков, дело скважины и т. д. Перечисленные документы подписываются разработчиками и согласующими, после чего утверждаются заказчиком. Электронный формат перечисленной документации представлен файлами word и excel.

В настоящее время в нефтяных компаниях существуют базы данных, в которых размещается большинство рабочей документации: программы, сводки, акты. В то же время имеются ранее пробуренные скважины, данные по которым ещё не заполнялись в базу данных, а также суточные сводки сервисных подрядчиков могут содержать более полную информацию. В связи с чем, вопрос по сбору данных из файлов excel и word является актуальной задачей.

Пример суточной сводки сервисного подрядчика по буровому раствору приведён на рисунке 1.

При выполнении анализа по одной или нескольким скважинам возникает вопрос, связанный со сбором информации из большого количества файлов, если это делать «вручную», то тратиться много времени. Одним из доступных средств автоматизации является интегрированная среда разработки Visual Basic for Application (VBA).

Для перехода из окна приложения excel или word в редактор Visual Basic Ediror (VBE) достаточно нажать комбинацию клавиш Alt+F11 или выбрать на ленте «Разработчик» – «Visual Basic». С основными принципами программирования на языке VBA можно познакомится в источниках [1, 2, 3].

Перед тем как рассматривать код на языке программирования, обратим внимание на структуру данных, встречающихся в суточных сводках (рис. 2).

Как видим данные могут находится в одной ячейке, например, дата составления сводки. Также может быть перечень данных, который остаётся одинаковым во всех сводках, но может отличаться на разных скважинах, например, перечень израсходованных химических реагентов. Ещё возможен перечень, количество позиций в котором меняется в каждой сводке – перечень с описанием работ. Сразу уточним, что далее будет рассмотрен самый простой случай – извлечение единичных данных.

Рисунок 1 – Пример суточной сводки по буровым растворам

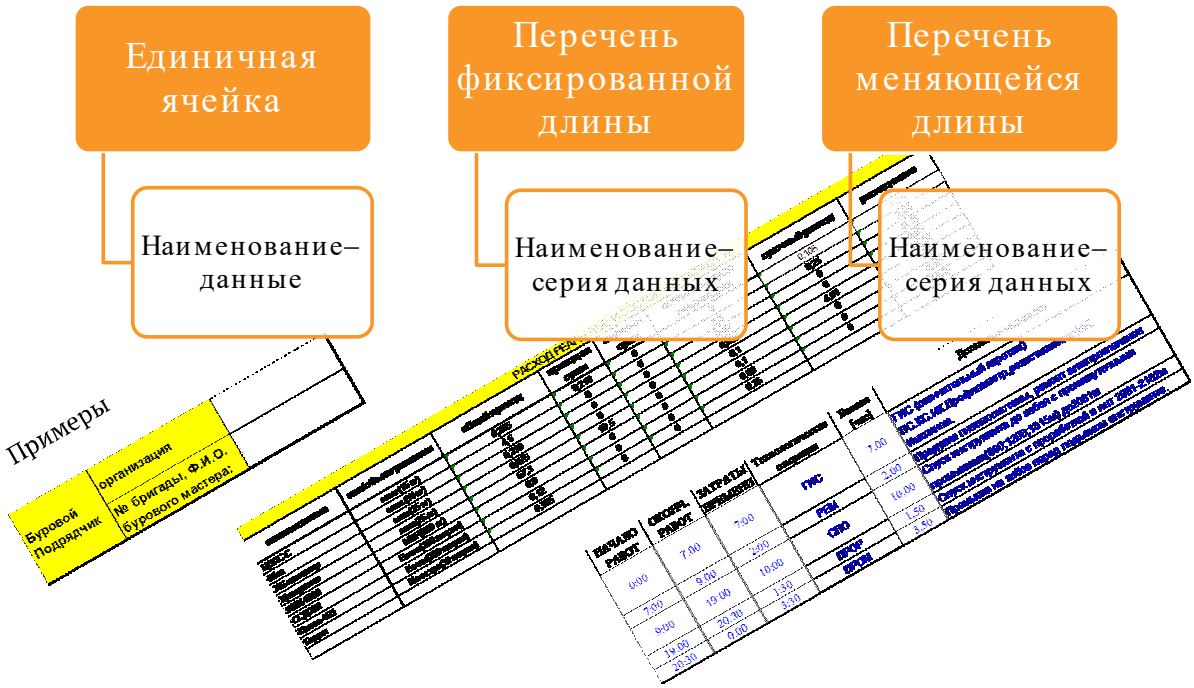

Рисунок 2 – Структура данных, встречающихся в суточных сводках

Т. к. в общем случае структура суточных сводок может меняться и ячейки, из которых надо извлечь данные, могут иметь различные адреса, то предлагается для нахождения требуемой ячейки сначала выполнять поиск на листе excel наименования параметра (аналогично поиску при нажатии комбинации клавиш Ctrl+F). Например, если мы хотим считать значение плотности бурового раствора, то сначала надо на листе excel найти «плотность», затем указать на какое количество ячеек требуется сместиться от найденной ячейки, для попадания в ячейку со значением, которое надо считать. Так, если искомое наименование параметра «плотность» находится в ячейке A57, а его значение в ячейке D58, то чтобы попасть в ячейку, из которой нужно считать данные, надо сместиться от найденной ячейки вправо на 2 (рис. 3).

	B	C	D	E
51	ПАРАМЕТРЫ РАСТВОРА ПО ПРОГРАММЕ			
52	плотность раствора, г/см 3	условная вязкость, с	водоотдача, см³/30 мин	PV, мПа·с
53	1,40 ±0,02	40-65	3,5-4,5	20-35
54	ПАРАМЕТРЫ РАСТВОРА НА СКВАЖИНЕ			
55	время	чч:мм	6.00	12.00
56	глубина	м	2730	2748
57	температура	°С	15,00	15,00
58	плотность	г/см3	1,38	1,38
59	усл.вязк.	сек	49	49
60	СНС 10s/10m	фунт/100фут²	4-6	4-6

Рисунок 3 – Структура данных, встречающихся в суточных сводках

Предлагаемая структура таблицы, содержащая инструкцию по считыванию данных, приведена в табл. 1. На листе excel её нужно располагать горизонтально, в данной статье для удобства восприятия она расположена вертикально.

Таблица 1 – Структура таблицы, задающей что считывать

Наименование результирующей таблицы		рапорты	рапорты
Наименование результирующего столбца		Дата	Скважина
Информация для поиска ячеек для считывания данных	Наименование показателя на рабочем листе	Дата	№ скв.
	Первое смещение считывания данных	вправо	
	На сколько ячеек отстоит ячейка с данными	1	
	Второе смещение считывания данных	вниз	
	На сколько ячеек отстоит ячейка с данными	1	
	Выполнять считывание из фиксированных адресов ячеек (д, н)		
	Адрес ячейки		B40
	Тип данных (т, ч, д, в)	д	т
	Выполнять перезапись ранее считанных данных (д, н)	н	н
Таблица в Access		raporty	raporty
Столбец в Access		Data	Skvazhina
Порядковый номер таблицы		2	2
Порядковый номер столбца в таблице		2	3

В структуру предлагаемой таблицы с заданием для считывания данных включены альтернативные способы получения данных: через поиск и путём задания фиксированного адреса ячеек. Ещё следует предусмотреть возможность проверки считанных данных, т. к. если суточные сводки заполнялись вручную, то в ячейке с числовым значением могут в конце поставить точку или в качестве дробного разделителя использовать то запятую, то точку и т. д. Также в таблицу с заданием для считывания включена информация для наименования таблиц и столбцов в access. Это станет необходимо, когда объём считываемых данных больше, чем может уместиться на один лист excel либо если требуется группировать данные инструментами, доступными в access.

Цветовые заливки ячеек (см. табл. 1) можно использовать для задания наименования файла с исходными данными после считывания. Удобно в процессе обработки заливать ячейки, которые были найдены и ячейки, из которых были считаны данные. Такие файлы с заливкой можно сохранять в отдельном каталоге, а путь с ссылкой на сохранённые файлы указывать в результирующей таблице. Пример считанных данных приведён в таблице 2.

Таблица 2 – Структура таблицы с результатами считывания данных

Путь к исходному файлу	C:\temp\Суточные рапорта	C:\temp\Суточные рапорта
Имя исходного файла	Рапорт 01.02.2013 г..xls	Рапорт 02.02.2013 г..xls
Дата и время создания исходного файла	18.02.2013 9:18	18.02.2013 9:24
Размер исходного файла	80384	80384
Имя листа исходного файла	Рапорт	Рапорт
Ссылка на сохраненный файл	c:\temp\Суточные рапорта\3332\2013_02_01.xls	c:\temp\Суточные рапорта\3332\2013_02_02.xls
Месторождение	xxx	xxx
Скважина	3332	3332
Дата	01.02.2013	02.02.2013
Плотность, г/см³	1,23	1,32

Чтобы обрабатывать большое количество файлов, их можно считывать из указанного каталога [4, 5] или выбирать в открывающемся окне [6, 7]. Приведём для примера часть кода, отображающего диалоговое окно для выбора файлов. Для удобства, вначале примеров с кодом приводится описание типов переменных. Также следует сказать, что при наименовании переменных желательно придерживаться соглашения Реддика. Для запуска приведённого кода надо открыть окно VBE, правой кнопкой мыши щёлкнуть на VBAProject (…) и в открывшемся выпадающем меню выбрать Insert – Module. Скопировать приведённый ниже текст и вставить его в окно редактора. Если после вставки русские символы превращаются в «крокозябры», то убедитесь, что при копировании включена русская раскладка «RU».

```
Option Base 1 'Указываем, чтобы индексация
    'во всех массивах начиналась с "1"
Sub ReadData()
'Модуль для считывания данных из суточных сводок
'Наименование книги, из которой запускается макрос
Dim strActiveWorkbooks As String
'Наименование листа, с которого запускается макрос
Dim strActiveWorksheets As String
'Переменные для открытия файлов
Dim strFilt As String
Dim intFilterIndex As Integer
Dim strTitle As String
Dim varFileName As Variant
Dim intFile As Integer
'Переменные для работы с файлами
Dim lngFileCount As Long
Dim strPathForRead As String 'Путь к обрабатываемому файлу
strActiveWorkbooks = ActiveWorkbook.Name
strActiveWorksheets = ActiveSheet.Name
    'Настройка списка файловых фильтров
strFilt = " Файлы (*.xls*),*.xls*," & _
    " (*.*),*.*"
'По умолчанию отображается *.xls*
intFilterIndex = 1
'Настройка заголовка диалогового окна
strTitle = "Выберите файлы для считывания"
'Получение имени файла
varFileName = Application.GetOpenFilename _
    (FileFilter:=Filt, _
    FilterIndex:=FilterIndex, _
    Title:=Title, _
    MultiSelect:=True)
'Выход, если диалоговое окно не выбрано
If Not IsArray(varFileName) Then
    MsgBox "Файлы не выделены"
```

```
        Exit Sub
    End If
    'Выполним перебор выбранных файлов
    For lngFileCount = LBound(varFileName) _
        To UBound(varFileName)
        strPathForRead = varFileName(lngFileCount)
        'Полный путь к файлу получен
        MsgBox "Полный путь к " & lngFileCount _
          & " выбранному файлу: " & Chr(10) _
          & strPathForRead
      Next lngFileCount
    End Sub
```

Обратите внимание, что при закрытии файла excel его надо сохранить в формате с поддержкой макросов.

После того, как мы в переменной strPathForRead получили полный путь к файлу, его надо открыть. Чтобы работать с листами файла excel добавим объявление новых переменных. Напоминаю, что переменные размещаются в начало модуля, а строки ниже приведённого кода с «'Выполняем открытие файла» – в место, где в предыдущем листинге было указано «'Полный путь к файлу получен». Также не забудьте закомментировать функцию MsgBox поставив перед ней апостроф «'».

```
    'Блок переменных открываемого, для считывания данных, листа
    Dim wbRead As Excel.Workbook
    Dim wsRead As Excel.Worksheet
        'Выполняем открытие файла
        Set wbRead = Workbooks.Open(FileName:= _
            strPathForRead, UpdateLinks:=0)
```

Последнее выполненное действие связано с открытием книги excel, если нам надо считать данные из файлов word, то в макросе excel сначала надо объявить соответствующие переменные. Здесь стоит вспомнить про раннее и позднее связывание [8]. Раннее связывание – это когда сначала задается тип переменной, потом ей присваивается внутренний или внешний объект. Раннее связывание для внешнего объекта можно выполнить только после подключения показанной на рисунке ссылки на библиотеку через главное меню VBA: Tools – References… Для разных версий word ссылка будет отличаться номером.

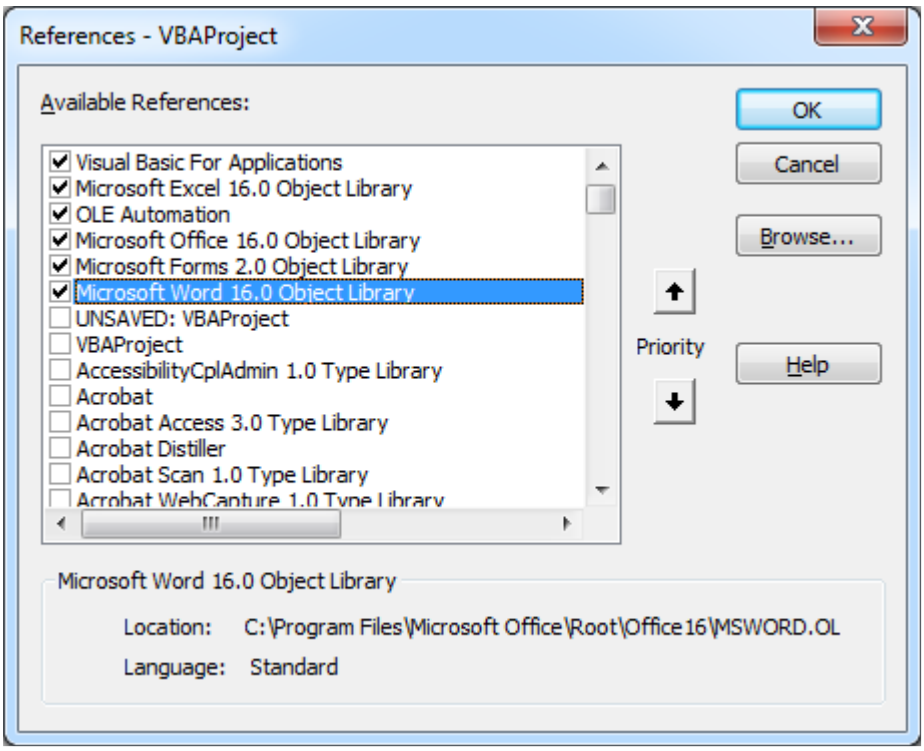

Рисунок 4 – Подключение библиотеки для работы с word из excel

```
'Блок переменных открываемого, для считывания данных, файла
Dim wd As Word.Application
Dim docSource As Word.Document
'Запуск программы "Word" через VBA,
Set wd = CreateObject(Class:="Word.Application")
'Делаем программу Word видимой, чтобы было удобно писать код
wd.Visible = True
```

Главное преимущество раннего связывания заключается в появлении подсказок при написании кода VBA. Лист подсказок отображается автоматически или вызывается сочетанием клавиш «Ctrl+Пробел» или «Ctrl+J». Кроме того, применение ранней привязки для создания объекта с помощью ссылки на библиотеку объектов обеспечивает более высокую производительность приложения.

Позднее связывание – это присвоение переменной, объявленной как Object, экземпляра внутреннего объекта с помощью ключевого слова Set или экземпляра внешнего объекта с помощью ключевого слова Set и функции GetObject (для получения сохраненного объекта) или CreateObject (для создания нового экземпляра). Тип объекта при поздней привязке определяется только в процессе выполнения программы.

```
'Позднее связывание с приложением Word
Dim objWord As Object
Set objWord = CreateObject("Word.Application")
```

Вернёмся к открытому файлу excel wbRead. Нужно уточнить, что не все файлы получится открыть, к таким, например, относятся имеющие пароль. Также при обработки большого количества файлов из разных источников могут возникать ошибки. Чтобы макрос не останавливался на всех ошибках, можно прописать приведённый ниже код.

```
'Проверяем не является ли файл временным
'If Left(strFileName, 2) <> "~$" Then
    'Включаем продолжение выполнения программы,
    'в случае возникновения ошибки
    On Error Resume Next
    'Очищаем все ошибки, чтобы перехватить
    'возникающие ошибки
    Err.Clear
    'Выполняем открытие файла
    Set wbRead = Workbooks.Open(FileName:= _
        strPathForRead, UpdateLinks:=0)
    If Err.Number > 0 Then
        'Очищаем все ошибки
        Err.Clear
        GoTo LastLine
    End If
    On Error GoTo 0 'Отключаем обработку ошибок
'Продолжение надо вставлять сюда
LastLine: 'Метка для перехода на нее при возникновении ошибок
    'Закрываем без сохранения файл, открытый для считывания
    wbRead.Close SaveChanges:=False
    Set wbRead = Nothing
  'End If
```

Как видите выше, сначала приведён код открывающий файл excel, а начиная с метки «LastLine:» приведён код, вставляемый после выполнения алгоритма по считыванию данных из открытой книги.

Следующим действием с открытым файлом является перебор всех его листов. Чтобы однозначно определять лист, на котором имеются требуемые данные для считывания, будем выполнять проверку по задаваемому ключевому слову. Ключевое слово должно быть уникально именно для тех листов, с которых будет считываться информация. В примере, который приводится в данной статье таким ключом является сочетание слов «СУТОЧНЫЙ РАПОРТ».

```
Dim strKeyForRead As String     'Ключ для проверки является ли
                                'открытый файл суточной сводкой
'Переменная для сохранения результатов поиска
Dim rgReadResult As Range
    For Each wsRead In wbRead.Worksheets
      If Err.Number > 0 Then
        'Очищаем все ошибки
        Err.Clear
```

```
            GoTo LastLine
          End If
          On Error GoTo 0 'Отключаем обработку ошибок
          strKeyForRead = "СУТОЧНЫЙ РАПОРТ"
          'Проверяем наличие ключа
          If Len(strKeyForRead) > 1 Then
            'Поиск ключа в книге
            Set rgReadResult = wsRead.Cells.Find(What:= _
              strKeyForRead, LookIn:=xlValues, _
              LookAt:=xlWhole, SearchOrder:=xlByRows, _
              SearchDirection:=xlNext, _
              MatchCase:=False, SearchFormat:=False)
            'Проверяем существование искомого ключа
            If Not rgReadResult Is Nothing Then
'Продолжение надо вставлять сюда
            End If
          End If
        Next wsRead
```

Подробно изучить синтаксис метода Find можно по справке или форумам в интернете. Здесь только уточним, что в указанном коде поиск осуществляется по значениям (LookIn:=xlValues), без учёта регистра (MatchCase:=False) и до полного совпадения (LookAt:=xlWhole). Если на открытом листе wsRead встречается искомое ключевое слово strKeyForRead, то двигаемся далее. Выполним очистку заливки ячеек, цвета шрифта и условного форматирования. Если в сводке значения всех параметров приведены в виде ссылок, то в начале строки «.Value = .Value» надо убрать знак комментирования «'».

```
            'Включаем продолжение выполнения программы,
            'в случае возникновения ошибки
            On Error Resume Next
            'Очищаем все ошибки
            Err.Clear
            With wsRead.UsedRange
              '.Value = .Value 'Заменяем все формулы на значения
              .Interior.ColorIndex = xlNone 'Очистка заливки ячеек
              .Font.ColorIndex = xlAutomatic 'Очистка цвета шрифта
              'Очистка условного форматирования ячеек
              .FormatConditions.Delete
              'Очищаем все ошибки
              Err.Clear
              On Error GoTo 0 'Отключаем обработку ошибок
            End With
            'Отмечаем найденную ячейку цветом
            rgReadResult.Interior.Color = RGB(165, 123, 10)
'Продолжение надо вставлять сюда
```

Мы добрались до нашей цели – далее в код добавляется поиск заданных в таблице 1 наименований. Если они в полном соответствии встречаются на листе wsRead, то от адреса найденной ячейки надо отступить указанное в таблице 1 количество строк и/или столбцов. Затем найденное значение проверяется на соответствие указанному типу данных (число, строка, дата или время). Если выявлено отличие, то следует создать алгоритм корректировки данных. После корректировки считанный результат можно сохранять в таблице 2. В соответствии с нашим примером таблица 2 распложена на листе «рапорты» и имеет только один считываемый параметр «Плотность, г/см³». Удобнее, чтобы на листе excel наименования находились в колонках, а данные считывались в строки, т. к. количество строк существенно больше количества колонок. При размещении наименований в колонках остальные параметры будут следовать за параметром «Плотность, г/см³».

В данной статье нет возможности привести листинг законченной программы, т. к. её объём существенно возрастёт и будет сложен для восприятия потому, что появится много блоков с одинаковой структурой. Данные из таблицы 1 можно считывать непосредственно с листа excel. Для доступа к ячейке существует два распространённых способа. Первый с помощью объекта Range, второй с помощью коллекции Cells [9]. Например, считывание текста из ячейки D5 листа strActiveWorksheets в переменную strReadText выполняется кодом:

```
strReadText = Workbooks(strActiveWorkbooks). _
Worksheets(strActiveWorksheets).Range("D5")
```

или

```
        strReadText = Workbooks(strActiveWorkbooks). _
        Worksheets(strActiveWorksheets).Cells(5, 4)
```

Для записи данных в ячейку меняем порядок:

```
        Workbooks(strActiveWorkbooks). _
        Worksheets(strActiveWorksheets).Range("D5") = strReadText
```

или

```
        Workbooks(strActiveWorkbooks). _
        Worksheets(strActiveWorksheets).Cells(5, 4) = strReadText
```

Нижняя черта «_» в коде позволяет переносить длинные строки для удобства чтения в редакторе VBE. А символ апострофа «'» применяют для комментирования (следующий за ним текст не воспринимаются транслятором).

Для ускорения работы программы можно предварительно сохранить требуемые для считывания данные в массивах. Также в массивы можно разместить структуру таблицы 2, чтобы при нахождении каждого наименования таблицы 1 легко определять наименование листов и адреса ячеек куда их надо сохранять.

```
        Dim strNameForFind As String 'Наименование параметра для поиска
        Dim lngReadRow As Long 'Номер строки
        Dim intReadCol As Integer 'Номер колонки
        Dim strDirection1 As String 'Направление первого смещения
        Dim intDirection1 As Integer 'Величина первого смещения
        Dim strDirection2 As String 'Направление второго смещения
        Dim intDirection2 As Integer 'Величина второго смещения
        Dim strType As String 'Ожидаемый тип данных (т, ч, д, в)
        Dim strReadText As String 'Переменная для считывания текста
        Dim curReadNumber As Currency 'Переменная для считывания числа
        Dim datReadDate As Date 'Переменная для считывания даты
        Dim datReadTime As Date 'Переменная для считывания времени
        'Имя листа для сохранения считанных данных
        Dim strWriteWorksheets As String
        'Номер строки для сохранения данных
        Dim lngWriteRow As Long
        'Номер колонки для сохранения данных
        Dim intWriteCol As Integer
        'Лист «рапорты» должен быть создан файле Excel
        strWriteWorksheets = "рапорты"
        lngWriteRow = 2 'Эту переменную надо менять через цикл
        'каждая строка будет соответствовать одной суточной сводке
        'Данные, которые будут задаваться с учётом табл. 1:
        intWriteCol = 10 'В соответствии с примером в табл. 2
        'Данные, которые будут считываться из табл. 1:
        strType = "ч" 'В соответствии с примером в табл. 2
            'Выполняем поиск считанного наименования параметра
            'в открытой сводке
            Set rgReadResult = wsRead.Cells.Find(What:= _
                strNameForFind, LookIn:=xlValues, _
                LookAt:=xlWhole, SearchOrder:=xlByRows, _
                SearchDirection:=xlNext, MatchCase:=False, _
                SearchFormat:=False)
            'Проверяем существование искомого параметра
            If Not rgReadResult Is Nothing Then
                'strAddressResultFind = rgReadResult.Address
                lngReadRow = rgReadResult.Row
                intReadCol = rgReadResult.Column
                'Отметить найденную ячейку цветом
                rgReadResult.Interior.Color = RGB(165, 123, 10)
                'Учитываем смещение
                Select Case Left(strDirection1, 2)
                Case "вв"
                    lngReadRow = lngReadRow – intDirection1
                Case "вн"
                    lngReadRow = lngReadRow + intDirection1
```

```
Case "вл"
    intReadCol = intReadCol – intDirection1
Case "вп"
    intReadCol = intReadCol + intDirection1
End Select
Select Case Left(strDirection2, 2)
Case "вв"
    lngReadRow = lngReadRow – intDirection2
Case "вн"
    lngReadRow = lngReadRow + intDirection2
Case "вл"
    intReadCol = intReadCol – intDirection2
Case "вп"
    intReadCol = intReadCol + intDirection2
End Select
'Проверяем наличие адреса для считывания данных
If lngReadRow > 0 And intReadCol > 0 Then
    'Заливаем ячейку из которой считываются данные
    wsRead.Cells(lngReadRow, intReadCol). _
    Interior.Color = RGB(0, 255, 0)
'Проверяем наличие заполненных данных
If Workbooks(strActiveWorkbooks). _
  Worksheets(strWriteWorksheets). _
  Cells(lngWriteRow, 1) <> "" Then
  'Сведения о файле были заполнены ранее
Else
  'Сохраняем информацию о считываемом файле
  Workbooks(strActiveWorkbooks). _
    Worksheets(strWriteWorksheets). _
    Cells(lngWriteRow, 1) = wbRead.path
  Workbooks(strActiveWorkbooks). _
    Worksheets(strWriteWorksheets). _
    Cells(lngWriteRow, 2) = wbRead.Name
  Workbooks(strActiveWorkbooks). _
    Worksheets(strWriteWorksheets). _
    Cells(lngWriteRow, 3) = FileDateTime _
      (wbRead.path & "\" & wbRead.Name)
  Workbooks(strActiveWorkbooks). _
    Worksheets(strWriteWorksheets). _
    Cells(lngWriteRow, 4) _
    = FileLen(wbRead.path & "\" & wbRead.Name)
  Workbooks(strActiveWorkbooks). _
    Worksheets(strWriteWorksheets). _
    Cells(lngWriteRow, 5) = wsRead.Name
End If
'Считываем данные в строковую переменную,
'при возникновении ошибки,
'выполняем следующую строку
On Error Resume Next
strReadText = wsRead.Cells(lngReadRow, intReadCol)
'Debug.Print Err.Number
If Err.Number > 0 Then
  strReadText = ""
  Err.Clear 'Очищаем все ошибки
  On Error GoTo 0 'Отключаем обработку ошибок
Else
'Проверяем тип данных
Select Case strType
Case "т"
  'Проверяем наличие заполненных данных
  If Workbooks(strActiveWorkbooks). _
```

```
            Worksheets(strWriteWorksheets). _
            Cells(lngWriteRow, intWriteCol) <> "" Then
          Else
            'Сохраним считанные данные
            Workbooks(strActiveWorkbooks). _
            Worksheets(strWriteWorksheets). _
            Cells(lngWriteRow, intWriteCol) _
            = """ & strReadText
          End If
          'Задаём формат ячейки
          Workbooks(strActiveWorkbooks). _
            Worksheets(strWriteWorksheets). _
            Cells(lngWriteRow, intWriteCol).WrapText _
            = False
        Case "ч"
          'Сюда надо вставить алгоритм, выделяющий
          'число из считанной строки
          curReadNumber = CStr(strReadText)
          'Проверяем наличие заполненных данных
          If Workbooks(strActiveWorkbooks). _
            Worksheets(strWriteWorksheets). _
            Cells(lngWriteRow, intWriteCol) <> "" Then
          Else
            'Сохраним считанные данные
            Workbooks(strActiveWorkbooks). _
              Worksheets(strWriteWorksheets). _
              Cells(lngWriteRow, intWriteCol) _
              = curReadNumber
          End If
          'Задаем формат ячейки
          Workbooks(strActiveWorkbooks). _
            Worksheets(strWriteWorksheets). _
            Cells(lngWriteRow, intWriteCol). _
            NumberFormat = "General"
        Case "д"
        Case "в"
          datReadTime = Empty
        End Select
      End If
    End If
End If
Теперь остаётся сохранить файл из которого выполнялось считывание данных.
'Сохраняем обработанный файл
wbRead.SaveCopyAs _
"c:\temp\Суточные рапорта\3332\2013_01_16.xls"
Также сохраним гиперссылку на файл, для удобства проверки что было считано.
'Сохраняем путь и формируем ссылку
With Workbooks(strActiveWorkbooks). _
    Worksheets(strWriteWorksheets)
    .Hyperlinks.Add Anchor:= _
        .Cells(lngWriteRow, 6), Address:= _
        "c:\temp\Суточные рапорта\3332\2013_01_16.xls"
End With
Workbooks(strActiveWorkbooks).Worksheets(strActiveWorksheets). _
    Activate
Workbooks(strActiveWorkbooks).Save
```

Литература

1. Слепцова Л.Д. Программирование на VBA в Microsoft Office 2010. – М. : ООО «И.Д. Вильямс», 2010. – 432 с.

2. Программирование в пакетах MS Office : учеб. пособие / С.В. Назаров [и др.] ; под общ. ред. С.В. Назарова. – М. : Финансы и статистика, 2007. – 656 с.
3. Биллиг В.А. VBA в Office 2000. Офисное программирование. – М. : Издательско-торговый дом «Русская Редакция», 1999. – 480 с.
4. Функция FilenamesCollection предназначена для получения списка файлов из папки, с учётом выбранной глубины поиска в подпапках : [Электронный ресурс]. – URL : https://excelvba.ru/code/FilenamesCollection (дата обращения: 29.03.2020).
5. Просмотреть все файлы в папке : [Электронный ресурс]. – URL: https://www.excel-vba.ru/chto-umeet-excel/prosmotret-vse-fajly-v-papke/ (дата обращения: 29.03.2020).
6. Уокенбах Д. Excel 2013: профессиональное программирование на VBA / Пер. с англ. – М. : ООО «И.Д. Вильямс», 2014. – С. 395–398.
7. Диалоговое окно выбора файлов/папки : [Электронный ресурс]. – URL : https://www.excel-vba.ru/chto-umeet-excel/dialogovoe-okno-vybora-fajlovpapki/ (дата обращения: 29.03.2020).
8. VBA Excel. Раннее и позднее связывание : [Электронный ресурс]. – URL : https://vremya-ne-zhdet.ru/vba-excel/ranneye-i-pozdneye-svyazyvaniye/ (дата обращения: 29.03.2020).
9. Работа с объектом Range : [Электронный ресурс]. – URL : http://perfect-excel.ru/publ/excel/makrosy_i_programmy_vba/rabota_s_obektom_range/7-1-0-56 (дата обращения: 29.03.2020).

References

1. Sleptsova L.D. Programming on VBA in Microsoft Office 2010. – M. : LLC «I.D. Williams», 2010. – 432 p.
2. Programming in MS Office packages : textbook / S.V. Nazarov [et al.]; under the general. ed. S.V. Nazarova. – M. : Finance and statistics, 2007. – 656 p.
3. Billig V.A. VBA in Office 2000. Office programming. – M. : Publishing and trading house «Russian Edition», 1999. – 480 p.
4. The FilenamesCollection function is designed to obtain a list of files from a folder, taking into account the selected search depth in the subfolders: [Electronic resource]. – URL : https://excelvba.ru/code/FilenamesCollection (accessed date: 03/29/2020).
5. View all files in the folder: [Electronic resource]. – URL: https://www.excel-vba.ru/chto-umeet-excel/prosmotret-vse-fajly-v-papke/ (accessed: 03/29/2020).
6. Walkenbach D. Excel 2013: Professional VBA Programming. – M. : LLC «I.D. Williams», 2014. – 395–398.
7. Dialog box for selecting files / folders: [Electronic resource]. – URL : https://www.excel-vba.ru/chto-umeet-excel/dialogovoe-okno-vybora-fajlovpapki/ (accessed: 03/29/2020).
8. VBA Excel. Early and late binding: [Electronic resource]. – URL : https://vremya-ne-zhdet.ru/vba-excel/ranneye-i-pozdneye-svyazyvaniye/ (accessed: 03/29/2020).
9. Work with the Range object: [Electronic resource]. – URL : http://perfect-excel.ru/publ/excel/makrosy_i_programmy_vba/rabota_s_obektom_range/7-1-0-56 (accessed: 03/29/2020).