Table of Contents

исходники 2

исходники

```
{excel2djvu - конвертор xls файлов с содержанием журнала в HTML формат для создания DJVU архива.

Copyright (C) 2006 by Alexander Sorkin aka Kibi

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-1307, USA.

                                11.11.2006
                                Alexander Sorkin aka Kibi
                                kibizoid@gmail.com
                                http://kibi.ru

*************************************************************************************}

// Исходный код основан на примере работы с Excel человека по имени Злой и опубликованного
//для свободного использования на сайте Королевство Дельфи.

unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ComObj, ActiveX, shellapi, FileCtrl,
{$IFDEF VER120}
  OleCtrls, // Почем-то так написано у меня - по-моему для D5 это лишнее.
            // Это склероз или диабет, не помню!
{$ENDIF}
  Excel8TLB, StdCtrls, ExtCtrls, ComCtrls, SHDocVw;


type
  TForm1 = class(TForm)
```

```
    StatusBar1: TStatusBar;
    OpenDialog1: TOpenDialog;
    PageControl2: TPageControl;
    TabSheet1: TTabSheet;
    Panel1: TPanel;
    html_preview: TWebBrowser;
    TabSheet2: TTabSheet;
    tree_preview: TTreeView;
    SaveDialog1: TSaveDialog;
    Panel2: TPanel;
    PageControl1: TPageControl;
    simple_tab: TTabSheet;
    grp_simple_volume: TGroupBox;
    btn_simple_volume: TButton;
    complex_tab: TTabSheet;
    btn_Initiate_Data: TButton;
    chk_debug_mode: TCheckBox;
    btn_Collect_Data: TButton;
    GroupBox1: TGroupBox;
    Button1: TButton;
    Button2: TButton;
    ErrorLog: TListView;
    TabSheet3: TTabSheet;
    single_volume_tree: TTreeView;
    local_progress: TProgressBar;
    global_progress: TProgressBar;
    edt_volumes: TComboBox;
    btn_Create_Global_Content: TButton;
    Label1: TLabel;
    Bevel1: TBevel;
    chk_make_bookmarks: TCheckBox;
    chk_make_authors: TCheckBox;
    chk_make_rubriks: TCheckBox;
    chk_make_content: TCheckBox;
    procedure FormDestroy(Sender: TObject);
    procedure Initiate_Data(Sender: TObject);
    procedure Local_Content_Create(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Global_Content_Create(Sender: TObject);
    procedure btn_simple_volumeClick(Sender: TObject);
    procedure btn_Collect_DataClick(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
    FIXLSApp: Excel8TLB._Application;
    FIWorkbook,FIWorkbook2: Excel8TLB._Workbook;
    AllContent, fileDJVU_Bookmarks, file_Global_Volumes_Content,
file_All_Authors_Index, file_All_Rubriks_Index:TStream;
    // Все ОНО тут!
    // Оформлено отдельными процедурами из противности.
```

```
    // Впрочем, отсюда проще взять для последующей утилизации.
    procedure CreateExcel (NewInstance: boolean);
    procedure ShowExcel;
    //procedure HideExcel;
    procedure ReleaseExcel;
    procedure ShowTree;
    procedure MakeContent;
    procedure ExportHTML(all:boolean);
    procedure CreateBook;
    procedure CreateTOC;
    procedure Load_complex_rubriks;
    procedure Load_rubriks_groups;
    procedure CreateBatches;
    function compare_rubriks(rubriki, test_string:string; var
other_rubriks:string; var pos_found:integer):boolean;
    procedure sort_articles;
    procedure ShowMessageInLog(year, volume, page, msg:string);
  public
    { Public declarations }
    // Свойство, спросите Вы? Так это же ООП!
    property IXLSApp: Excel8TLB._Application read FIXLSApp;
    property IWorkbook: Excel8TLB._Workbook read FIWorkbook;
    property IWorkbook2: Excel8TLB._Workbook read FIWorkbook2;
  end;
Const Make_Single = false;
      Make_Global = true;
      min_year = 1990;
      max_year = 2005;
      max_toc_triplet = 7;
      max_complex_rubriks = 5;

Type ContentElement = class
     public
        year_vol_str : string;
        str_vol : string;
        rubrika : string;
        author : string;
        title : string;
        page : string;
        link : string;
        volume : string;
        year : string;
        function ColoredResult:string;
        constructor Create(rubrika, author, title, page, link, volume, year:
string);
     end;

Type TMyContent = record
        rubrika : string;
        author : string;
        title : string;
```

```
        page : string;
        link : string;
        volume : string;
        year : string;
        used_as_slave: boolean;
        used_as_article: array [1..max_complex_rubriks] of boolean;
     end;
     index_element = record
        name : string;
        index : integer;
     end;
     rubr_index_element = record
        name : string;
        index : integer;
        other_rubrik : string;
     end;
     authors_element = record
        name:string;
        index: integer;
        count: integer;
        triplet: string;
        surname: string;
     end;
     rubriks_element = record
        name:string;
        index: integer;
        count: integer;
        sinonim: string;
        common: integer;
        group_number: integer;
        number : integer;
     end;
     triplets_element = record
        triplet:string;
        index: integer;
        count: integer;
        letter: string;
        toc_triplets_index: integer;// связь с массивом toc_triplets - к какому
диапазону принадлежит.
     end;
     sinonims_element = record
        rubrika: string;
        sinonim: string;
     end;
     rubriks_type_element = record
        rubrika:string;
        is_table_rubrik:integer; // 1 - если рубрика табличная, 0 - списочная
     end;
     rubriks_group_element = record
        group: string;          // название группы
        rubriks: TStringList;   // перечень рубрик в группе
```

```
     end;
  type TRubr_Info = class(TObject)
       public
          info : rubriks_element;
          constructor Create(init_info: rubriks_element);
       end;
var
  Form1: TForm1;
  VolumeContent: array of TMyContent;
  Rubriki:TStringList;
  Rubriks_groups:array of rubriks_group_element;
  prog_path: string;
  simple_mode: boolean;
implementation

{$R *.DFM}
constructor TRubr_Info.Create(init_info: rubriks_element);
begin
   self.info := init_info;
end;
function ContentElement.ColoredResult:string;
begin
//
end;

constructor ContentElement.Create(rubrika, author, title, page, link,
volume, year: string);
begin
        self.rubrika := rubrika;
        self.author := author;
        self.title := title;
        self.page := page;
        self.link := link;
        self.volume := volume;
        self.year := year;
        if ((year = '1992') and (volume = '05')) then begin
           year_vol_str := '1992_05_06';
           str_vol := '05-06';
        end else begin
           year_vol_str := format('%s_%s',[self.year, self.volume]);
           str_vol := self.volume;
        end;
end;


procedure TForm1.CreateExcel(NewInstance: boolean);
var IU: IUnknown;
    isCreate: boolean;
begin
  // Tricks:
  //    в случает вызова  Excel-а, как сервера автоматизации:
```

```
  //      - все автозагружаемые книги не загружаются (и слава богу!);
  //      - создается процесс, но Excel в случае создания нового процесса,
  //        естественно, невидим.
  if not Assigned(IXLSApp) then begin // а зачем его создавать, если уже есть?
    isCreate := NewInstance or
      (not SUCCEEDED( GetActiveObject(Excel8TLB.CLASS_Application_, nil, IU)
) );
    if isCreate then
      FIXLSApp := CreateComObject(Excel8TLB.CLASS_Application_) as
Excel8TLB._Application
    else
      FIXLSApp := IU as Excel8TLB._Application;
  end;
end;
procedure TForm1.ShowMessageInLog(year, volume, page, msg:string);
var newItem:TListItem;
begin
   newItem := ErrorLog.Items.Add;
   newItem.Caption := (inttostr(ErrorLog.Items.Count));
   newItem.SubItems.Add (format('%s, №%s',[year,volume]));
   newItem.SubItems.Add (page);
   newItem.SubItems.Add (msg);
end;
procedure TForm1.ShowExcel;
begin
  // Tricks:
  //    Приведение к TOLEEnum сделано только для того, чтобы не видеть
  //         warning-ов типа "Constant expression violates subrange bounds" и
т.п.
  //
  //         lcid - что такое? Что-то насчет локализации, уже не помню. Надо уточнить!
  //    Но вы всегда можете спокойно передать 0 вместо lcid.
  //
  //    Так насчет показать Excel... Показать просто. Главное корректно показать.
  //    Вроде бы, достаточно просто Visible := true. Черта с два!
  //    Комментируем строки с условием на минимизацию и ScreenUpdating.
  //    Делаем CreateExcel, ShowExcel, минимизируем его (мышью - чем еще?),
  //    возвращаемся в приложение и делаем еще раз ShowExcel! И что?
  //    Насколько я понял его поведение, Visible в таком случае сфокусирует туда,
  //    но окно-то не раскроется. Поэтому проверка на xlMinimize.
  //
  //    Что здесь делает ScreenUpdating? Многие спросят. Делаем:
  //    - раскомментим строки с условием на минимайз;
  //         - CreateExcel;
  //         - ShowExcel;
  //    - закрываем (мышью) Excel (можно через Файл\Закрыть);
  //    - смотрим в процессы - Excel не выгрузился (так оно и понятно - интерфейс не
освободили);
  //    - Главное! Делаем ShowExcel - имеем право.
  //    При последнем действии Excel у меня активизируется, но не перерисовывается.
  //    Эту проблему последняя строка и решает. Этот баг я называю "прозрачный Excel".
```

```
  //    А мы тут полупрозрачные окна рисуем. Наверняка, MS скоро в WinAPI это
положит!
  if Assigned(IXLSApp) then begin
    IXLSApp.Visible[0] := true;
    //IXLSApp.Visible[0] := false;
    if IXLSApp.WindowState[0] = TOLEEnum(Excel8TLB.xlMinimized) then
      IXLSApp.WindowState[0] := TOLEEnum(Excel8TLB.xlNormal);
    IXLSApp.ScreenUpdating[0] := true;
    IXLSApp.DisplayAlerts[0] := true;
  end;
end;

(*procedure TForm1.HideExcel;
begin
  if Assigned(IXLSApp) then begin
    IXLSApp.Visible[0] := false;
  end;
end;*)

procedure TForm1.ReleaseExcel;
begin
  // Tricks:
  //    Верни пользователю его Excel!!!
  //    Считаем, что если остается хотя бы одна открытая книга, значит пользователь ее
хотел
  //    бы потом увидеть. А то ж можно сделать Excel-у Hide и Release. И что?
  //    Если там осталась книга, которая правилась - процесс останется, но в списке
приложений
  //    его не увидишь - только в списке процессов.
  //    Так решим эту проблему следующим образом:
  if Assigned(IXLSApp) then begin
    // если есть книга и мы невидимы
    if (IXLSApp.Workbooks.Count > 0) and (not IXLSApp.Visible[0]) then begin
      // положим аккуратненько (правильно написал это слово?) его вниз,
      IXLSApp.WindowState[0] := TOLEEnum(xlMinimized);
      // а потом покажем,
      IXLSApp.Visible[0] := true;
      // ну, и после вернем фокус форме и приложению, а то после Visible - у эксела
SetFocus
      // происходит.
      // Собственно говоря, набор свойств Visible, ScreenUpdating, Activate (еще
остановимся) и
      // WindowState меня периодически пугает своей неоднозначностью.
      // Дельфийная альтернатива Visible, SetFocus, WindowState работает четко.
      if not(csDestroying in ComponentState) then Self.SetFocus; // а это
объяснять необходимо?
      // Объясню вопросом. Какой SetFocus, если приложение (форма) закрывается?
      Application.BringToFront;
    end;
  end;
  // Tricks:
```

```
  // Главное мое правило: поюзал интерфейс - сам его и освободи.
  // Эта привычка выработана давно. Освобождаю даже локально объявленные переменные-интерфейсы.
  // То есть блок finally всегда присутствует с такими присваиваниями.
  // Совет начинающим от начинающего: если чего-то не выгружается - проверь, освобождаются ли
  // все использованные интерфейсы, особенно те, которые в полях классов и глобальных переменных.
  // Кстати, последнее - дурной стиль.
  // Максималист, я знаю. По мне, дурной стиль всякая лишняя глобальная переменная, будь то
  // экземпляр класса, целое число или интерфейс. Я даже сношу описания типа var Form1: TForm1.
  // Для использования Excel у меня существует несколько классов.
  // Это классы XL Report - низкоуровневый набор компонентов, обеспечивающий
  // создание одного отчета по нескольким TDataSet. А вся подсистема отчетов описана еще одним
  // классом, который умеет создать эти самые наборы данных под конкретный отчет, хранить
  // описания этих запросов (Это запросы. Параметризованные, конечно). К тому же, все это
  // завязано на COM и ActiveX Scripting. Это не оттого, что я такой крутой. Просто пора
  // давно понять, что начинать строить настоящие приложения надо с создания интерфейса
  // МоеПриложение.Application. Это не мода, это, по-моему, уже необходимость.
  FIXLSApp := nil;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
  prog_path:=ExtractFilePath(ParamStr(0));
  if not DirectoryExists(prog_path+'bookmarks/') then ForceDirectories(prog_path+'bookmarks/');
  CreateExcel(false);
  ShowExcel;
  Application.BringToFront;
  if fileexists(prog_path+'!simple_mode.txt') then simple_mode := true else simple_mode := false;
  if not simple_mode then CreateBook;
  if simple_mode then begin
     Simple_tab.TabVisible:=true;
     complex_tab.TabVisible:=false;
     pagecontrol1.ActivePage:=Simple_tab;
  end else begin
     Simple_tab.TabVisible:=false;
     complex_tab.TabVisible:=true;
     pagecontrol1.ActivePage:=complex_tab;
  end;
  Rubriki:=TStringList.Create;
//  Rubriki.LoadFromFile(prog_path+'rubriki.txt');
end;
```

```
procedure TForm1.FormDestroy(Sender: TObject);
begin
  FIWorkbook.Close(EmptyParam, EmptyParam, EmptyParam,0);
  FIWorkbook:= nil;
  FIWorkbook2:= nil;
  ReleaseExcel; // святое дело
  Rubriki.Free;
end;

procedure TForm1.CreateBook;
var FullFileName: string;
    x:integer;
    Value: OLEVariant;
begin
  if simple_mode then begin
     OpenDialog1.InitialDir:=prog_path;
     if OpenDialog1.Execute then begin
       FullFileName:=OpenDialog1.FileName;
     end;
  end else FullFileName := prog_path + 'content.xls';
  if Assigned(IXLSApp) and (not Assigned(IWorkbook) ) then
    try
      try
        FIWorkbook := IXLSApp.Workbooks.Open(FullFileName,
             EmptyParam, EmptyParam, EmptyParam, EmptyParam, EmptyParam,
EmptyParam,
             EmptyParam, EmptyParam, EmptyParam, EmptyParam, EmptyParam,
false, 0);
        edt_volumes.Items.Clear;
        edt_volumes.Sorted:=false;
        for x:=1 to IWorkbook.Names.Count do begin // прочесть из экселя список
уже существующих именованных диапазонов
           Value:=IWorkbook.Names.Item(x,EmptyParam,EmptyParam).Name;
           edt_volumes.Items.Add(Copy(VarToStr(Value), length('vol_')+1,
length(VarToStr(Value))-length('vol_')));
           // префикс к номеру в списке нам не нужен - чтобы оставались чистые  yyyy_mm
(имя в экселе не может начинаться с цифры)
        end;
        edt_volumes.Sorted:=true;
        if edt_volumes.Items.Count>0 then edt_volumes.ItemIndex:=0;
      finally
        ShowExcel;
        Application.BringToFront;
      end;
    except
      raise Exception.Create('Не могу открыть книгу!');
    end;
end;

procedure TForm1.Initiate_Data(Sender: TObject);
{
```

```
  Простановка именованных диапазонов
  Исходные данные - уже открытый  XLS  файл, лист "Содержание"
  Конечный результат - именованные диапазоны в  XLS  файле
  Происходит сброс всех именных диапазонов и сохранение обработанного файла

  Именно тут надо сделать разбор списка авторов и построение списка "Автор - ссылки на статьи",
  а так же списка "Рубрика - ссылки на статьи"

  Сам процесс разбора - поэтапный
    - построение неупорядоченного списка 1 автор - 1 ссылка на статью
    - сортировка созданного списка по авторам / статьям
    + экспорт в текстовый файл статистики - автор/количество статей - для проверки и построения списка исключений
    + статистика, сколько ссылок у одной рубрики - нужна при составлении дерева рубрик
    + регалии хранить не требуется - автор имеет только фамилию, имя и отчество

  Запользование этого списка идёт уже на этапе построения авторского указателя и рубрикатора соответственно
  потому как в одной строке не получается хранить больше 1000 символов и 500 ссылок в одну строку на укладываются :(

  Оглавление по авторам - также три уровня
    - первый уровень - таблица на одну страницу - алфавит с уточнением - диапазонами из трёх букв
      Количество диапазонов - фиксированное - для равномерного заполнения первой страницы
    - второй уровень - полный перечень авторов для каждого диапазона
    - третий - перечень статей для каждого автора (возможно с фоткой и краткой биографией)



}
var vYear, vMonth, vAuthor, vRubrika, Values, prevValue, outValues:
OLEVariant;
    ISheet, ISheet_auth_index, ISheet_rubr_index, ISheet_authors,
ISheet_rubriks,
    ISheet_triplet, ISheet_sinonims, ISheet_rubriks_type:
Excel8TLB._Worksheet;
    IRange: Excel8TLB.Range;
    //IR1, IR2: IxlRange;
    articles: array of TMyContent;
    auth_index:array of index_element;
    authors: array of authors_element;
    triplets: array of triplets_element;
    rubr_index: array of rubr_index_element;
    rubriks: array of rubriks_element;
    sinonims: array of sinonims_element;
    rubriks_type: array of rubriks_type_element;
    x, first,last:integer;
    stroka,prevStroka,prevVolume:string;
    auth_index_counter, // счётчик авторских статей
```

```
    authors_counter, // счётчик уникальных авторов
    triplet_counter:integer; // счётчик триплетов
    rubr_index_counter, // счётчик статей с рубриками
    rubriks_counter: integer; // счётчик уникальных рубрик

  function get_sinonim(rubrika:string):string;
  // вычисляем синоним рубрики (по предварительно составленному вручную списку)
  // список синонимов находится в массиве  sinonims
  var x:integer;
  begin
     for x:=0 to length(sinonims)-1 do begin
        if sinonims[x].rubrika=rubrika then begin
           result:=sinonims[x].sinonim;
           exit;
        end;
     end;
     result:=rubrika;
  end;
  function get_rubrik_type(calculated_type:integer; rubrika:string):integer;
  // получаем тип рубрики (по вычисленному типу и предварительно составленному вручную
списку)
  // список типов рубрик от пользователя находится в массиве  rubriks_type
  var x:integer;
  begin
     for x:=0 to length(rubriks_type)-1 do begin
        if rubriks_type[x].rubrika = rubrika then begin
           result := rubriks_type[x].is_table_rubrik;
           exit;
        end;
     end;
     result := calculated_type;
  end;
procedure process_rubriks(rubriki:string; link:integer);
  function other_rubriks(rubriks:TStringList; i: integer):string;
  var x: integer;
      temp_str:string;
  begin
     temp_str := '';
     for x:=0 to rubriks.Count-1 do begin
        if x<i then begin // запомнить все рубрики до  i  -  "материнские" рубрики -
рубрики более высокого уровня
           if temp_str = ''
              then temp_str := rubriks[x]
              else temp_str := temp_str + ' | ' + rubriks[x];
        end;
     end;
     result := temp_str;
  end;
  var x:integer;
      temp:string;
      mas:TStringList;
```

```
  begin
     if rubriki='' then exit; // пропускаем статьи без рубрик

     rubriki:=StringReplace(rubriki, ' | ', '@', [rfReplaceAll]); // получено
в вычищенном виде при сборке

     mas:=TStringList.Create;
     temp:='';
     for x:=1 to length(rubriki) do begin // разбиваем строку на подстроки,
разделитель "@"
       if rubriki[x]='@' then begin
          mas.Add(temp);
          temp:='';
       end else temp:=temp+rubriki[x];
     end;
     mas.Add(temp); // добавить последний найденный элемент
     for x := 0 to mas.Count-1 do begin
        setlength(rubr_index,rubr_index_counter+1); // завести место под новый
элемент
        rubr_index[rubr_index_counter].name:= get_sinonim(mas[x]);
        rubr_index[rubr_index_counter].index:= link;
        rubr_index[rubr_index_counter].other_rubrik := other_rubriks(mas,x);
// сопутствующие рубрики
        inc(rubr_index_counter);
     end;
     mas.Free;
  end;
function extract_surname(name:string):string;
  {фамилия вычленяется следующим образом - берётся конец строки до пробела "М.
Иванов", "Михаил Иванов".
 Итого, алгоритм критичен к отсутствию пробела после точки - это можно исправить, но
тогда не будут появляться разные авторы
 "М.Иванов" и "М. Иванов"}
  var temp:string;
      x:integer;
  begin
     temp:='';
     for x:=length(name) downto 1 do begin
        if name[x]=' ' then break;
        temp := name[x] + temp;
     end;
     extract_surname:=temp;
  end;
function check_name(name:string):boolean;
// проверка качества имени - имя должно быть только из русских букв
var x:integer;
    test_str:string;
begin
   for x:=1 to length(name) do begin
      if not (name[x] in ['а'..'я','А'..'Я','ё','Ё','.',' ','''','-']) then begin
         result := false;
```

```
         exit;
      end;
   end;
   test_str := extract_surname(name);
   if length(test_str)<3 then begin // триплет должен иметь три символа
      result := false;
      exit;
   end;
   for x:=1 to length(test_str) do begin
      if not (test_str[x] in ['А'..'Я','Ё','''','-']) then begin // ограничения на
символы в фамилии
         result := false;
         exit;
      end;
   end;

   result := true;
end;
function process_authors(rubrika, authors:string; link:integer):boolean;
//link - номер статьи в основном списке (на листе "Содержание")
  var x:integer;
      temp:string;
      mas:TStringList;
      ok: boolean;
  begin
     ok := true;
     if authors='' then begin result := ok; exit end; // пропускаем статьи без
авторов
     if pos('Рефераты',rubrika)>0 then begin result := ok; exit end; //
пропускаем авторов рефератов
     if pos('Вкладк',rubrika)>0 then begin result := ok; exit end; //
пропускаем вкладки
     if pos('Обложк',rubrika)>0 then begin result := ok; exit end; //
пропускаем обложки
     temp := copy (authors,1,5);

     authors := StringReplace(authors,'и','@',[rfReplaceAll, rfIgnoreCase]);
     authors := StringReplace(authors,', ','@',[rfReplaceAll,
rfIgnoreCase]);

     mas:=TStringList.Create;
     temp:='';
     for x:=1 to length(authors) do begin // разбиваем строку на подстроки,
разделитель "@"
       if authors[x]='@' then begin

          if (length(temp)>0) and (temp[1]=AnsiUpperCase(temp[1])) and  //
добавлять только имена (начинаются с заглавной буквы)
             (pos('кадемии',temp)=0) and // никаких "кадемии" в имени нет
             (pos('осударствен',temp)=0) and // никаких "осударствен" в имени нет
             (pos('Украины',temp)=0) // никаких "Украины" в имени нет
```

```
            then mas.Add(temp);
          temp:='';
       end else temp:=temp+authors[x];
     end;
     if (temp<>'') and (temp[1]=AnsiUpperCase(temp[1])) and  // добавлять
только имена (начинаются с заглавной буквы)
        (pos('кадемии',temp)=0) and // никаких "кадемии" в имени нет
        (pos('осударствен',temp)=0) and // никаких "осударствен" в имени нет
        (pos('Украины',temp)=0) // никаких "Украины" в имени нет
        then mas.Add(temp);
     for x := 0 to mas.Count-1 do begin
        setlength(auth_index,auth_index_counter+1); // завести место под новый
элемент
        if not check_name(mas[x]) then ok := false;
        auth_index[auth_index_counter].name:= mas[x];
        auth_index[auth_index_counter].index:= link;
        inc(auth_index_counter);
     end;
     mas.Free;
     result := ok;
  end;
  function detect_rubriks_group(rubrik_name:string):integer;
  var x:integer;
      n: integer;
      results: integer;
  begin
     results:=0;
     for x:=1 to length(Rubriks_Groups)-1 do begin
        if Rubriks_Groups[x].rubriks.Find(rubrik_name, n) then begin
           results := x;
           break;
        end
     end;
     detect_rubriks_group := results;
  end;
  function detect_common_rubriks(i:integer; rubrik_name:string):integer;
     Const min_vol = 1;
           max_vol = 12;
           test_vol = max_vol + 1;
           magic_number = 6; // количество номеров с рубрикой в году,
                             // при котором рубрика считается регулярной
     var x:integer;
         user_year, user_vol:integer;
         all_volumes:array[min_year..max_year, min_vol..max_vol+1] of
integer;
  begin
    // Заполнить таблицу ссылок (all_volumes) для рубрики
    FillChar(all_volumes,sizeof(all_volumes),0); // инициализировать таблицу
номеров
    for x := rubriks[i].index to rubriks[i].index+rubriks[i].count-1 do
begin // прочесать все статьи для рубрики x
```

```
        try
           user_year := strtoint(articles[rubr_index[x].index].year); //
заполучить год, в котором статья
           user_vol := strtoint(articles[rubr_index[x].index].volume); //
заполучить номер, в котором статья
        except
           user_year := 0;
           user_vol := 0;
        end;
        if (user_year >= min_year) and (user_year <= max_year) and
        (user_vol >= min_vol) and (user_vol <= max_vol)
        then begin
           if all_volumes[user_year, user_vol] = 0 then begin
              all_volumes[user_year, user_vol] := 1; // в этом номере есть статьи
нужной рубрики
              inc(all_volumes[user_year, test_vol]); // увеличить годовой счётчик
           end;
        end;
     end;
     for x:=min_year to max_year do begin
        if all_volumes[x, test_vol] >= magic_number then begin

           detect_common_rubriks:=get_rubrik_type(1,rubrik_name); exit;

        end;
     end;
     detect_common_rubriks:=get_rubrik_type(0,rubrik_name);
  end;
begin
  sort_articles; // отсортировать статьи
  Load_rubriks_groups; // загрузить список групп рубрик
  // загрузить полный список статей

  auth_index_counter := 1;
  rubr_index_counter := 1;
  global_progress.Min:=0;
  global_progress.max:=15; // основные этапы
  global_progress.Position:=local_progress.Min;
  global_progress.Step:=1;
  if Assigned(IWorkbook) then
    try
      ISheet := IWorkbook.Worksheets.Item['Содержание'] as
Excel8TLB._Worksheet;
      if not simple_mode then begin
         ISheet_auth_index := IWorkbook.Worksheets.Item['Авторы'] as
Excel8TLB._Worksheet;
         ISheet_auth_index.Cells.Clear; // очистить лист "Авторы"
         ISheet_rubr_index := IWorkbook.Worksheets.Item['Рубрики'] as
Excel8TLB._Worksheet;
         ISheet_rubr_index.Cells.Clear; // очистить лист "Рубрики"
         ISheet_authors := IWorkbook.Worksheets.Item['Авторы (свод)'] as
```

```
Excel8TLB._Worksheet;
         ISheet_authors.Cells.Clear; // очистить лист "Авторы (свод)"
         ISheet_rubriks := IWorkbook.Worksheets.Item['Рубрики (свод)'] as
Excel8TLB._Worksheet;
         ISheet_rubriks.Cells.Clear; // очистить лист "Рубрики (свод)"
         ISheet_triplet := IWorkbook.Worksheets.Item['Триплеты'] as
Excel8TLB._Worksheet;
         ISheet_triplet.Cells.Clear; // очистить лист "Триплеты"
         // загрузить список типов рубрик (табличная/списочная)
         ISheet_rubriks_type := IWorkbook.Worksheets.Item['Типы рубрик'] as
Excel8TLB._Worksheet;
         IRange := ISheet_rubriks_type.UsedRange[0];
         Values := IRange.Value;
         SetLength(Rubriks_type,IRange.Rows.Count+1);
         global_progress.StepIt;
         local_progress.Min:=0;
         local_progress.max:=IRange.Rows.Count;
         local_progress.Position:=local_progress.Min;
         local_progress.Step:=1;
         statusbar1.simpletext:='Загрузка листа "Типы рубрик"...';
         for x := 1 to IRange.Rows.Count do begin
            local_progress.stepit;
            with Rubriks_type[x] do begin
               rubrika := Values[x, 1];
               if (Values[x, 2] = 'таблица') then is_table_rubrik := 1 else
is_table_rubrik := 0;
            end;
         end;
         ISheet_rubriks_type := nil;

         // загрузить список синонимов рубрик
         ISheet_sinonims := IWorkbook.Worksheets.Item['Синонимы'] as
Excel8TLB._Worksheet;
         IRange := ISheet_sinonims.UsedRange[0];
         Values := IRange.Value;
         SetLength(Sinonims,IRange.Rows.Count+1);
         global_progress.StepIt;
         local_progress.Min:=0;
         local_progress.max:=IRange.Rows.Count;
         local_progress.Position:=local_progress.Min;
         local_progress.Step:=1;
         statusbar1.simpletext:='Загрузка листа "Синонимы"...';
         for x := 1 to IRange.Rows.Count do begin
            local_progress.stepit;
            with Sinonims[x] do begin
               rubrika := Values[x, 1];
               sinonim := Values[x, 2];
            end;
         end;
         ISheet_sinonims := nil;
      end;
```

```
      try
        for x:=1 to IWorkbook.Names.Count do begin
           IWorkbook.Names.Item(1,EmptyParam,EmptyParam).delete; {удалить все
именованные диапазоны}
        end;
        edt_volumes.Items.Clear;
        edt_volumes.Sorted:=false;


        // загрузить лист "Содержание"
        IRange := ISheet.UsedRange[0];  // прочесть весь лист
        Values := IRange.Value;
        SetLength(Articles,IRange.Rows.Count+1); // установить длину массива
статей
        // загрузить массив статей
        global_progress.StepIt;
        local_progress.Min:=1;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Загрузка массива статей...';
        for x:=1 to IRange.Rows.Count do begin
          local_progress.StepIt;
          with Articles[x] do begin
             rubrika := Values[x, 1];
             author := Values[x, 2];
             title := Values[x, 3];
             page := Values[x, 4];
             link := Values[x, 5];
             volume := Values[x, 6];
             year := Values[x, 7];
          end;
        end;
        with Articles[2] do begin
           stroka := year + '_' + volume; first:=2;
           if not simple_mode then begin
              if not process_authors(rubrika,author,2) then // вычленить
авторов из описания и положить в список авторов
                 ShowMessageInLog(year,volume,page,format('Некорректное имя
автора - "%s"',[author]));
              process_rubriks(rubrika,2); // вычленить рубрики из описания и
положить в список рубрик
           end;
        end;

        global_progress.StepIt;
        local_progress.Min:=3;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Построение списка номеров...';
```

```
        for x:=3 to IRange.Rows.Count do begin
           local_progress.StepIt;
           statusbar1.simpletext:='Построение списка номеров (строка
'+inttostr(x)+')...';
           with Articles[x] do begin
              prevVolume := articles[x-1].volume;
              prevStroka := stroka;
              if not simple_mode then begin
                 if not process_authors(rubrika,author,x) then // вычленить
авторов из описания и положить в список авторов
                    ShowMessageInLog(year, volume, page, format('Некорректное
имя автора - "%s"',[author]));
                 process_rubriks(Rubrika,x); // вычленить рубрики из описания и
положить в список рубрик
              end;
              stroka := Year + '_' + volume;
              if (stroka <> prevStroka) then begin // сменился блок без
разделителя
                 statusbar1.simpletext:='Построение списка номеров...
('+prevStroka+')';
                 last:=x-1;
                 IWorkbook.Names.Add('vol_'+prevStroka,
EmptyParam,EmptyParam,EmptyParam,EmptyParam,EmptyParam,EmptyParam,EmptyParam
,EmptyParam,
'=Содержание!R'+IntToStr(first)+'C1:R'+IntToStr(last)+'C7',EmptyParam);
                 edt_volumes.Items.Add(prevStroka);
                 first:=x;
              end;
              if (prevVolume = '') and (volume = '') then break;
           end;
        end;
        last:=IRange.Rows.Count;
        if prevStroka<>'_' then begin IWorkbook.Names.Add('vol_'+prevStroka,
EmptyParam,EmptyParam,EmptyParam,EmptyParam,EmptyParam,EmptyParam,EmptyParam
,EmptyParam,
'=Содержание!R'+IntToStr(first)+'C1:R'+IntToStr(last)+'C7',EmptyParam);
           edt_volumes.Items.Add(prevStroka);
        end;
        if not simple_mode then begin

        // выгрузить инфу на лист ISheet_auth_index
        outValues := VarArrayCreate([1, length(auth_index), 1, 2],
varVariant);
        for x:=1 to length(auth_index)-1 do begin
           outValues[x, 1] := auth_index[x].name;
           outValues[x, 2] := auth_index[x].index;
        end;
        IRange:=
ISheet_auth_index.Range['A1','B'+inttostr(length(auth_index)-1)];
        IRange.Value := outValues;
        VarClear(outValues);
```

```
        global_progress.StepIt;
        sort_articles;

        // выгрузить инфу на лист ISheet_rubr_index
        outValues := VarArrayCreate([1, length(rubr_index), 1, 3],
varVariant);
        for x:=1 to length(rubr_index)-1 do begin
           outValues[x, 1] := rubr_index[x].name;
           outValues[x, 2] := rubr_index[x].index;
           outValues[x, 3] := rubr_index[x].other_rubrik;
        end;
        IRange:=
ISheet_rubr_index.Range['A1','C'+inttostr(length(rubr_index)-1)];
        IRange.Value := outValues;
        VarClear(outValues);

        global_progress.StepIt;
        sort_articles;

        // построение списка уникальных авторов
        IRange := ISheet_auth_index.UsedRange[0];  // прочесть весь лист
        Values := IRange.Value;

        // то самое место, где жутко тормозит
        vAuthor := Values[1, 1]; first:=1; //last:=0;
        statusbar1.simpletext:='Построение списка авторов...';
        authors_counter := 1;

        global_progress.StepIt;
        local_progress.Min:=2;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        for x:=2 to IRange.Rows.Count do begin // построить список авторов
            local_progress.StepIt;
            prevValue := vAuthor;
            vAuthor := Values[x, 1];
            if (vartostr(vAuthor) <> vartostr(prevValue)) then begin //
сменился блок
               last := x-1;
               stroka:=extract_surname(VarToStr(prevValue));

               setlength(authors,authors_counter+1); // завести место под новый
элемент
               with authors[authors_counter] do begin
                  name := vartostr(prevValue);
                  index := first;
                  count := last-first+1;
                  triplet:= copy(stroka,1,3);
                  surname:= stroka;
```

```
            end;
            inc(authors_counter);
            first := x;
        end;
    end;
    last := IRange.Rows.Count;
    stroka:=extract_surname(VarToStr(vAuthor));

    setlength(authors,authors_counter+1); // завести место под новый элемент
    with authors[authors_counter] do begin
       name := vartostr(vAuthor);
       index := first;
       count := last-first+1;
       triplet:= copy(stroka,1,3);
       surname:= stroka;
    end;
    //inc(authors_counter);

    // выгрузить инфу на лист ISheet_authors
    outValues := VarArrayCreate([1, length(authors), 1, 5], varVariant);
    for x:=1 to length(authors)-1 do begin
       outValues[x, 1] := authors[x].name;
       outValues[x, 2] := authors[x].index;
       outValues[x, 3] := authors[x].count;
       outValues[x, 4] := authors[x].triplet;
       outValues[x, 5] := authors[x].surname;
    end;
    IRange:= ISheet_authors.Range['A1','E'+inttostr(length(authors)-1)];
    IRange.Value := outValues;
    VarClear(outValues);

    sort_articles;
    global_progress.StepIt;

    // построение списка уникальных рубрик
    IRange := ISheet_rubr_index.UsedRange[0];  // прочесть весь лист
    Values := IRange.Value;

    // загрузить индекс рубрик назад
    global_progress.StepIt;
    local_progress.Min:=2;
    local_progress.max:=IRange.Rows.Count;
    local_progress.Position:=local_progress.Min;
    local_progress.Step:=1;
    Setlength(Rubr_index,IRange.Rows.Count+1);
    for x := 1 to IRange.Rows.Count do begin
      local_progress.stepit;
      with Rubr_index[x] do begin
         name := Values[x, 1];
         index := Values[x, 2];
      end;
```

```
        end;

        vRubrika := Values[1, 1]; first:=1; //last:=0;
        statusbar1.simpletext:='Построение списка рубрик...';
        rubriks_counter := 1;

        global_progress.StepIt;
        local_progress.Min:=2;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        for x:=2 to IRange.Rows.Count do begin // построить список рубрик
            local_progress.StepIt;
            prevValue := vRubrika;
            vRubrika := Values[x, 1];
            if (vartostr(vRubrika) <> vartostr(prevValue)) then begin //
сменился блок
               last := x-1;
               stroka:=get_sinonim(VarToStr(prevValue));
               setlength(rubriks,rubriks_counter+1); // завести место под новый
элемент
               with rubriks[rubriks_counter] do begin
                  name := vartostr(prevValue);
                  index := first;
                  count := last-first+1;
                  common := detect_common_rubriks(rubriks_counter, stroka);
// определить, регулярная ли рубрика
                  group_number := detect_rubriks_group(stroka); // определить
группу рубрики
               end;
               inc(rubriks_counter);
               first := x;
            end;
        end;
        last := IRange.Rows.Count;
        stroka:=get_sinonim(VarToStr(vRubrika));
        // обработать последний элемент
        setlength(rubriks,rubriks_counter+1); // завести место под новый элемент
        with rubriks[rubriks_counter] do begin
           name := vartostr(vRubrika);
           index := first;
           count := last-first+1;
           common := detect_common_rubriks(rubriks_counter, stroka); //
определить, регулярная ли рубрика
           group_number := detect_rubriks_group(stroka); // определить группу
рубрики
        end;
        //inc(rubriks_counter);

        // выгрузить инфу на лист ISheet_rubriks
        outValues := VarArrayCreate([1, length(rubriks), 1, 5], varVariant);
```

```
        for x:=1 to length(rubriks)-1 do begin
           outValues[x, 1] := rubriks[x].name;
           outValues[x, 2] := rubriks[x].index;
           outValues[x, 3] := rubriks[x].count;
           outValues[x, 4] := rubriks[x].common;
           outValues[x, 5] := rubriks[x].group_number;

        end;
        IRange:= ISheet_rubriks.Range['A1','E'+inttostr(length(rubriks)-1)];
        IRange.Value := outValues;
        VarClear(outValues);

        sort_articles;
        global_progress.StepIt;

        // построение списка триплетов (трёхбуквенных комбинаций)
        IRange := ISheet_authors.UsedRange[0];  // прочесть весь лист
        Values := IRange.Value;

        vAuthor := Values[1, 4]; first:=1; //last:=0;
        statusbar1.simpletext:='Построение списка триплетов...';
        triplet_counter:=1;

        global_progress.StepIt;
        local_progress.Min:=2;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        for x:=2 to IRange.Rows.Count do begin // построить список триплетов
            local_progress.StepIt;
            prevValue := vAuthor;
            vAuthor := Values[x, 4];
            if vartostr(vAuthor) <> vartostr(prevValue) then begin // сменился
блок
               last := x-1;
               stroka:=extract_surname(VarToStr(prevValue));

               setlength(triplets,triplet_counter+1); // завести место под новый
элемент
               with triplets[triplet_counter] do begin
                  triplet := vartostr(prevValue);
                  index := first;
                  count := last-first+1;
                  letter:= copy(stroka,1,1);
               end;
               inc(triplet_counter);
               first := x;
            end;
        end;
        last := IRange.Rows.Count;
        stroka:=extract_surname(VarToStr(vAuthor));
```

```
        setlength(triplets,triplet_counter+1); // завести место под новый элемент
        with triplets[triplet_counter] do begin
           triplet := vartostr(vAuthor);
           index := first;
           count := last-first+1;
           letter:= copy(stroka,1,1);
        end;
        //inc(triplet_counter);

        // выгрузить инфу на лист ISheet_triplet
        outValues := VarArrayCreate([1, length(triplets), 1, 4],
varVariant);
        for x:=1 to length(triplets)-1 do begin
           outValues[x, 1] := triplets[x].triplet;
           outValues[x, 2] := triplets[x].index;
           outValues[x, 3] := triplets[x].count;
           outValues[x, 4] := triplets[x].letter;
        end;
        IRange:=
ISheet_triplet.Range['A1','D'+inttostr(length(triplets)-1)];
        IRange.Value := outValues;
        VarClear(outValues);

        sort_articles;
        end; // simple_mode
        global_progress.StepIt;
        statusbar1.simpletext:='Сохранение файла...';

        local_progress.Position := local_progress.max;
        edt_volumes.Sorted:=true;
        if edt_volumes.Items.Count>0 then edt_volumes.ItemIndex:=0;
          //0: Value := ISheet.Cells.Item[2, 1].Value;
          //1: Value := ISheet.Range['A2', EmptyParam].Value;
          //2: Value := ISheet.Range['TestCell', EmptyParam].Value;
          //3: Value := IWorkbook.Names.Item('TestCell', EmptyParam,
EmptyParam).RefersToRange.Value;
          //ShowMessageInLog(Value);
      finally
        IRange := nil;
        ISheet := nil;
        ISheet_auth_index := nil;
        ISheet_rubr_index := nil;
        ISheet_authors := nil;
        ISheet_rubriks := nil;
        ISheet_triplet := nil;
        IWorkBook.Save(0);
        setlength(auth_index,0);
        setlength(rubr_index,0);
        setlength(authors,0);
        setlength(rubriks,0);
```

```
        setlength(triplets,0);
        setlength(sinonims,0);
        statusbar1.simpletext:='Готово';
        global_progress.StepIt;
      end;
    except
      raise Exception.Create('Не могу прочитать данные!');
    end;
    local_progress.Position := local_progress.Max;
    global_progress.Position := global_progress.Max;
end;

Procedure TForm1.sort_articles;
var ISheet: Excel8TLB._Worksheet;
begin
  if Assigned(IWorkbook) then begin
     ISheet := IWorkbook.Worksheets.Item['Содержание'] as
Excel8TLB._Worksheet;
     try
       try
ISheet.Columns.Sort(ISheet.Range['G1',emptyparam],xlAscending,ISheet.Range['
F1',emptyparam],EmptyParam,
         xlAscending,ISheet.Range['D1',emptyparam],xlAscending,
         xlYes, EmptyParam, EmptyParam,
         xlTopToBottom, xlStroke,
         true, true,
         true);
       finally
         ISheet := nil;
       end;
     except
       raise Exception.Create('Не могу отсортировать статьи!');
     end;

     if not simple_mode then begin
     ISheet := IWorkbook.Worksheets.Item['Авторы'] as Excel8TLB._Worksheet;
     try
       try
ISheet.Columns.Sort(ISheet.Range['A1',emptyparam],xlAscending,ISheet.Range['
B1',emptyparam],EmptyParam,
         xlAscending,ISheet.Range['C1',emptyparam],xlAscending,
         xlNo, EmptyParam, EmptyParam,
         xlTopToBottom, xlStroke,
         true, true,
         true);
       finally
         ISheet := nil;
       end;
     except
       raise Exception.Create('Не могу отсортировать авторов!');
     end;
```

```
     ISheet := IWorkbook.Worksheets.Item['Рубрики'] as Excel8TLB._Worksheet;
     try
       try
ISheet.Columns.Sort(ISheet.Range['A1',emptyparam],xlAscending,ISheet.Range['
B1',emptyparam],EmptyParam,
         xlAscending,ISheet.Range['C1',emptyparam],xlAscending,
         xlNo, EmptyParam, EmptyParam,
         xlTopToBottom, xlStroke,
         true, true,
         true);
       finally
         ISheet := nil;
       end;
     except
       raise Exception.Create('Не могу отсортировать рубрики!');
     end;


     ISheet := IWorkbook.Worksheets.Item['Авторы (свод)'] as
Excel8TLB._Worksheet;
     try
       try
ISheet.Columns.Sort(ISheet.Range['D1',emptyparam],xlAscending,ISheet.Range['
E1',emptyparam],EmptyParam,
         xlAscending,ISheet.Range['F1',emptyparam],xlAscending,
         xlNo, EmptyParam, EmptyParam,
         xlTopToBottom, xlStroke,
         true, true,
         true);
       finally
         ISheet := nil;
       end;
     except
       raise Exception.Create('Не могу отсортировать свод авторов!');
     end;

     ISheet := IWorkbook.Worksheets.Item['Рубрики (свод)'] as
Excel8TLB._Worksheet;
     try
       try
ISheet.Columns.Sort(ISheet.Range['E1',emptyparam],xlAscending,ISheet.Range['
A1',emptyparam],EmptyParam,
         xlAscending,ISheet.Range['F1',emptyparam],xlAscending,
         xlNo, EmptyParam, EmptyParam,
         xlTopToBottom, xlStroke,
         true, true,
         true);
       finally
         ISheet := nil;
       end;
```

```
     except
       raise Exception.Create('Не могу отсортировать свод рубрик!');
     end;


     end; // simple_mode
  end;
end;

procedure TForm1.Load_complex_rubriks;
var Values: OLEVariant;
    ISheet: Excel8TLB._Worksheet;
    IRange: Excel8TLB.Range;
    i: integer;
begin
   // загрузить список сборных рубрик
   ISheet := IWorkbook.Worksheets.Item['Сборные рубрики'] as
Excel8TLB._Worksheet;
   IRange := ISheet.UsedRange[0];
   local_progress.Min:=0;
   local_progress.max:=IRange.Rows.Count;
   local_progress.Position:=local_progress.Min;
   local_progress.Step:=1;
   statusbar1.simpletext:='Загрузка списка сборных рубрик...';
   Values := IRange.Value;
   //global_progress.StepIt;
   Rubriki.Clear;
   for i := 1 to IRange.Rows.Count do begin
      local_progress.stepit;
      Rubriki.Add(Values[i,1]);
   end;
   ISheet := nil;
   statusbar1.simpletext:='';
end;


{заполняет структуру Rubriks_groups на основе информации с листа "Группы рубрик"}
procedure TForm1.Load_rubriks_groups;
var Values: OLEVariant;
    ISheet: Excel8TLB._Worksheet;
    IRange: Excel8TLB.Range;
    i,j: integer;
    temp_str: string;
begin
   ISheet := IWorkbook.Worksheets.Item['Группы рубрик'] as
Excel8TLB._Worksheet;
   IRange := ISheet.UsedRange[0];
   local_progress.Min:=0;
   local_progress.max:=IRange.Columns.Count;
   local_progress.Position:=local_progress.Min;
   local_progress.Step:=1;
```

```
   statusbar1.simpletext:='Загрузка списка типов рубрик...';
   Values := IRange.Value;
   setlength(Rubriks_groups,IRange.Columns.Count+1);
   Rubriks_groups[0].group := 'Другие рубрики';
   Rubriks_groups[0].rubriks := TStringList.Create; // создать контейнер для
рубрик
   for i := 1 to IRange.Columns.Count do begin
      local_progress.stepit;
      with Rubriks_groups[i] do begin
         group := Values[1,i]; // получить название группы
         rubriks := TStringList.Create; // создать контейнер для рубрик
         rubriks.Sorted := true;
         for j:=2 to IRange.Rows.Count do begin // загрузить рубрики
            temp_str := Values[j,i];
            if (temp_str <> '') then rubriks.Add(temp_str) else break;
         end;
      end;
   end;
   ISheet := nil;
   statusbar1.simpletext:='';
end;

procedure TForm1.Local_Content_Create(Sender: TObject);
var year_vol:string;
    x:integer;
    year, load_year_vol:string;
   procedure LoadContent(year_vol:string);
   var Values: OLEVariant;
       ISheet: Excel8TLB._Worksheet;
       IRange: Excel8TLB.Range;
       i,j: integer;
       stroka: string;
   begin
      try
         ISheet := IWorkbook.Worksheets.Item['Содержание'] as
Excel8TLB._Worksheet;
         IRange := ISheet.Range['vol_'+year_vol, EmptyParam];
         Values := IRange.Value;
         SetLength(VolumeContent,IRange.Rows.Count);
         for i := 1 to IRange.Rows.Count do begin
            with VolumeContent[i-1] do begin
              rubrika := Values[i, 1];
              author := Values[i, 2];
              title := Values[i, 3];
              page := Values[i, 4];
              link := Values[i, 5];
              volume := Values[i, 6];
              year := Values[i, 7];
              used_as_slave := false;
              for j:=1 to max_complex_rubriks do used_as_article[j] :=
false;
```

```
              stroka:= format('%s * %s * %s * %s * %s * %s * %s', [rubrika,
author, title, page, link, volume, year]);
           end;
         end;
      finally
         IRange := nil;
         ISheet := nil;
      end;
   end;
begin
  if Assigned(IWorkbook) then begin
     Load_complex_rubriks; // загрузить список сборных рубрик
     local_progress.Min:=0;
     local_progress.max:=edt_volumes.Items.Count;
     local_progress.Position:=local_progress.Min;
     local_progress.Step:=1;
     for x:=0 to edt_volumes.Items.Count-1 do begin
        AllContent:=TMemoryStream.Create;
        local_progress.StepIt;
        load_year_vol:=edt_volumes.Items[x];
        year_vol:=load_year_vol;
        year := copy(year_vol, 1, 4);
        if (year_vol = '1992_05') then year_vol := '1992_05_06';
        statusbar1.simpletext:=format('Создание закладок для
%s.djvu...',[year_vol]);
fileDJVU_Bookmarks:=TFileStream.Create(prog_path+'bookmarks\'+year_vol+'.htm
', fmCreate);
        LoadContent(load_year_vol);
        MakeContent;
        ExportHTML(make_single);
        fileDJVU_Bookmarks.CopyFrom(AllContent,0); // скопировать всё оглавление
в файл с оглавлением
        fileDJVU_Bookmarks.Free;
        AllContent.Free;
     end;
  end;
  statusbar1.simpletext:='Готово';
end;

procedure TForm1.CreateBatches;
var bat_file: TStringList;
    sed_path: string;
    x:integer;
    year, year_vol: string;
function test_path:boolean;
var x:integer;
    results:boolean;
begin
  results := true;
  for x:=1 to length(prog_path) do begin
     if (prog_path[x] in ['А'..'Я', 'а'..'я', 'Ё', 'ё', ' ']) then
```

```
      begin
         results := false;
         break;
      end;
  end;
  test_path := results;
end;
begin
  if not test_path then begin
  Application.MessageBox(
    'Путь к рабочей папке содержит нежелательные символы - автоматическая конвертация
в DJVU будет невозможна.'+#13#10+#13#10+
    'Имена папок должны быть набраны латиницей и не содержать пробелов',
    'Предупреждение о проблемах',
    MB_OK);
  end;
  bat_file:=TStringList.Create;

  // !insert_bookmarks.bat

  for x:=0 to edt_volumes.Items.Count-1 do begin
     year_vol:=edt_volumes.Items[x];
     year := copy(year_vol, 1, 4);
     if (year_vol = '1992_05') then year_vol := '1992_05_06';
     bat_file.Add(format('@call !insert_one.bat %s %s',[year, year_vol]));
  end;
  bat_file.SaveToFile(prog_path+'!insert_bookmarks.bat');

  // !insert_one.bat

  bat_file.Clear;
  bat_file.Add('@echo off');
  bat_file.Add('echo Inserting bookmarks into %1\%2.djvu');
  bat_file.Add('bookmark.exe "bookmarks\%2.htm" "%1\%2.djvu"');
  bat_file.Add('del "%1\%2.djvu"');
  bat_file.Add('ren "%1\%2.new.djvu" "%2.djvu"');
  bat_file.SaveToFile(prog_path+'!insert_one.bat');

  // !do_all.bat

  bat_file.Clear;
  bat_file.Add('@call !process_one.bat all_authors');
  bat_file.Add('@call !process_one.bat all_rubriks');
  bat_file.Add('@call !process_one.bat all_content');
  bat_file.Add('@call !insert_bookmarks.bat');
  bat_file.SaveToFile(prog_path+'!do_all.bat');

  // !Process_one.bat

  bat_file.Clear;
  bat_file.Add('@ECHO OFF');
```

```
  bat_file.Add('echo Converting PDF to DJVU...');
  bat_file.Add('"pdftodjvu.exe" --dpi=300 --threshold-level=80 --fg-
quality=quasilossless --bg-subsample=3 '+
  '--quality=75 --words --convert-links --fgbg-sep --fg-colors=256 --fg-
image-colors=40 --pages-per-dict=20 '+
  '--force --profile=clean300 --ignore-color-profiles "' + prog_path +
'%1.pdf" "' + prog_path + '%1.djvu"');
  bat_file.Add('echo Extracting links from DJVU...');
  bat_file.Add('djvused.exe "%1.djvu" -e output-ant > "%1.txt"');
  bat_file.Add('echo Correcting links...');

  // вычислить строку для коррекции ссылок внутри DJVU - коррекция бага Acrobat'a
  sed_path := prog_path;                                          //
G:\NiZh\read_excel\
  sed_path:=StringReplace(sed_path, ':', '|', [rfReplaceAll]);  //
G|\NiZh\read_excel\
  sed_path:=StringReplace(sed_path, '\', '/', [rfReplaceAll]);  //
G|/NiZh/read_excel/
  sed_path:='../file:///' + sed_path;                            //
../file:///G|/NiZh/read_excel/
  sed_path:=StringReplace(sed_path, '/', '\/', [rfReplaceAll]); //
..\/file:\/\/\/G|\/NiZh\/read_excel\/

  bat_file.Add('sed -e "s/' + sed_path + '//" "%1.txt" > "%1.temp.txt"');
  bat_file.Add('echo Inserting links into DJVU...');
  bat_file.Add('djvused.exe "%1.djvu" -f "%1.temp.txt" -s');
  bat_file.Add('echo Inserting bookmarks into DJVU...');
  bat_file.Add('bookmark.exe all_bookmarks.htm %1.djvu');
  bat_file.Add('del "%1.djvu"');
  bat_file.Add('ren "%1.new.djvu" "%1.djvu"');
  bat_file.Add('echo Deleting temporary files...');
  bat_file.Add('del "%1.txt"');
  bat_file.Add('del "%1.temp.txt"');
  bat_file.Add('echo %1.djvu is ready.');
  bat_file.SaveToFile(prog_path+'!process_one.bat');
  bat_file.Free;
end;

procedure TForm1.ShowTree;
var i: integer;
    srcNode:TTreeNode;
    stroka:string;
begin
        tree_preview.Items.Clear;
        tree_preview.Items:=single_volume_tree.Items; // перенести из дерева
оглавления структуру
        srcNode:=tree_preview.TopItem;
        while srcNode <> nil do begin
           if pos('_',srcNode.Text)=0 then
              i:=StrToInt(srcNode.text)
           else
```

```
              i := strToInt(copy(SrcNode.Text,1,pos('_',SrcNode.Text)-1));
           stroka := '';
           if (VolumeContent[i].rubrika <> '') then stroka := '[' +
VolumeContent[i].rubrika + '] ';
           if (VolumeContent[i].author <> '') then stroka := stroka + '(' +
VolumeContent[i].author + ') — ';
           if (VolumeContent[i].title <> '') then stroka := stroka +
VolumeContent[i].title;
           if length(stroka)>255 then stroka:=copy(stroka,1,250)+'...';
           srcNode.text:=stroka;     // и подменить цифры на осмысленный текст
           srcNode:=srcNode.GetNext;
         end;
end;

procedure TForm1.MakeContent;
var i,y: integer;
    Node, lastNode:TTreeNode;
    stroka, stroka_spec:string;
    special_rubrika, slave, master:boolean;
    other_rubriks:string;
    pos_found:integer;

   procedure connect_slaves(Node:TTreeNode; master_link: string);
   // собирает все подчинённые статьи (подвёрстки) для данной мастер-статьи
   // на входе - узел мастер-статьи и мастер-ссылка
   var x:integer;
       stroka : string;
       master_flag : boolean;
       counter : integer;
       article_number:integer;
   begin
      master_flag:=true;
      stroka:=master_link;
      counter:=0;
      Delete(stroka,1,1); // получить соответствующий слэйв-линк
      stroka:= 'д' + stroka;
      for x:=0 to length(VolumeContent)-1 do begin
         with VolumeContent[x] do begin
            if link = master_link then inc(counter); // подсчитать кол-во мастер-
линков (должен быть один)
            if link = stroka then begin  // найден соответсвующий слэйв
               single_volume_tree.Items.AddChild(Node, IntToStr(x));
               used_as_slave:=true;
               master_flag:=false;
            end;
         end;
      end;
      // получить номер статьи для вывода сообщения
      if pos('_',Node.Text)=0 then begin
         article_number := StrToInt(Node.Text);
      end else begin
```

```
         article_number := strToInt(copy(Node.Text,1,pos('_',Node.Text)-1));
      end;

      if master_flag then begin // проверка на "orphan master" - одинокие мастер-статьи
         with VolumeContent[article_number] do begin
            ShowMessageInLog(year, volume, page, format('Не найдено ни одной подвёрстки (%s) для [%s %s %s] (%s)',[stroka, rubrika, author, title, link]));
         end;
      end;
      if counter<>1 then begin
         with VolumeContent[article_number] do begin
            ShowMessageInLog(year, volume, page, format('Главная статья имеет уже использованный номер связи %s [%s %s %s] (%s)',[master_link, rubrika, author, title, link]));
         end;
      end;
   end;

   procedure connect_subrubriks(Node:TTreeNode; root_rubrika: string);
   // собирает все статьи из сборной рубрики
   // на входе - узел главной статьи и имя сборной рубрики
   var x:integer;
       stroka:string;
       lastNode:TTreeNode;
       master:boolean;
       other_rubriks:string;
       rubr_position:integer;
   begin
      rubr_position := 0;
      for x:=0 to length(VolumeContent)-1 do begin
         with VolumeContent[x] do begin
            if compare_rubriks(rubrika,root_rubrika,other_rubriks,rubr_position) then begin // найдена статья для спецрубрики
               lastNode:=single_volume_tree.Items.AddChild(Node, IntToStr(x)+'_'+root_rubrika);
               used_as_article[rubr_position] := true;
               if (link='') then begin // определить мастер/слэйв
                  master:=false;
               end else begin
                  stroka:=Copy(link,1,1);
                  master:=stroka='a';
               end;
               if master then connect_slaves(lastNode,link);
            end;
         end;
      end;
   end;
    {
```

```
  Нужно модифицировать алгоритм сборки оглавления:
  статья, у которой две сборные рубрики, появляется в оглавлении дважды.
  => признак used_as_article недостаточен для решения.
  нужен массив флагов для всех сборных рубрик "used_as_article #x"

  Для того, чтобы не делать массив флагов зависимым от списка сборных рубрик,
  надо использовать принцип - "использовано как сборная рубрика для первой рубрики" и
  ограничить количество контролируемых сборных рубрик, например пятью.

    used_as_article: array [0..5] of boolean;
    for i:=0 to 5 do used_as_slave[i]:=false;
    used_as_article[0] - использована как обычная статья (не из сборных рубрик)
    used_as_article[1] - использована как статья первой спецрубрики и т.д.

    }
begin
        single_volume_tree.Items.Clear;
        Node:=single_volume_tree.TopItem;
        for i:=0 to length(VolumeContent)-1 do begin
           with VolumeContent[i] do begin
              // определить мастер/слэйв
              if (link='') then begin
                 slave:=false;
                 master:=false;
              end else begin
                 stroka:=Copy(link,1,1);
                 master:=stroka='а';
                 slave:=stroka='д';
              end;
              // Узнать, относится ли рубрика к списку специальных
              special_rubrika:=false;
              for y:=0 to Rubriki.Count-1 do begin
                 if
compare_rubriks(rubrika,Rubriki.Strings[y],other_rubriks, pos_found) then
begin
                    special_rubrika:=true;
                    // проверить, что найдена сборная рубрика, по которой статья ещё не
собиралась
                    if not used_as_article[pos_found] then begin
                       stroka_spec:=Rubriki.Strings[y]; // запомнить название
спецрубрики, т.к. рубрика может быть сложносочинённой
                       break;
                    end else begin pos_found:=0; stroka_spec:=''; end;
                 end;
              end;
              // на выходе - признак, что есть спецрубрика.
              // stroka_spec - только для случая, когда найдена неиспользованная
рубрика
              // pos_found - тоже только тогда, когда найдена неиспользованная
рубрика
              if special_rubrika then begin
```

```
                    // если найдена спецрубрика и она неиспользованная - подключить
спецрубрики
                    if pos_found > 0 then begin
                       if not used_as_article[pos_found] then begin //
спецрубрики строятся даже по слэйвам (обложки, вкладки)
                          lastNode:=single_volume_tree.Items.Add(Node,
IntToStr(i)+'_'+stroka_spec); // добавить спецрубрику
                          connect_subrubriks(lastNode,stroka_spec);
                       end;
                    end;
                 end else begin
                    if not (used_as_article[1]) and not (used_as_slave) and not
slave then begin
                       lastNode:=single_volume_tree.Items.Add(Node,
IntToStr(i)); // добавить статью
                       used_as_article[1]:=true;
                       if master then connect_slaves(lastNode,link);
                    end;
                 end;
              end;
           end;
           // проверка на "orphan slaves" - одинокие подвёрстки
           for i:=0 to length(VolumeContent)-1 do begin
              with VolumeContent[i] do begin
                 if (link='') then begin // определить мастер/слэйв
                    slave:=false;
                 end else begin
                    stroka:=Copy(link,1,1);
                    slave:=stroka='д';
                 end;
                 if not (used_as_slave) and slave then begin
                    stroka:=link;
                    Delete(stroka,1,1); // получить соответствующий мастер-линк
                    stroka:= 'a' + stroka;
                    ShowMessageInLog(year, volume, page, format('Нет главной
статьи (%s) для подвёрстки [%s %s %s] (%s)',[stroka, rubrika, author, title,
link]));
                 end;
              end;
           end;
end;

function TForm1.compare_rubriks(rubriki, test_string:string; var
other_rubriks:string; var pos_found:integer):boolean;
//сравнение рубрик - rubriki - набор рубрик для статьи (разделён " | ")
//                   test_string - искомая рубрика
//    возвращает other_rubriks - набор рубрик без искомой рубрики (разделитель " | ")
//    возвращает pos_found - порядковый номер найденной рубрики в наборе рубрик
var temp:string;
    position:integer;
    counter:integer;
```

```
begin
   other_rubriks:='';
   pos_found := 0;
   counter := 0;
   if rubriki=test_string then begin result:=true; pos_found := 1; exit end;
   if Pos('|',rubriki)=0 then begin result:=false; exit end;

   rubriki:=StringReplace(rubriki, ' | ', '|', [rfReplaceAll]); // получено в
вычищенном виде при сборке

   result:=false;
   repeat
      position:=Pos('|',rubriki);
      if position>0 then begin
         temp:=Copy(rubriki,1,position-1); // скопировать кусок от начала до
найденного "|"
         Delete(Rubriki,1,position); // удалить в рубриках всё, до найденного "|",
включая разделитель
         inc(counter);
         if temp=test_string
            then begin // рубрика обнаружена
               pos_found := counter;
               result:=true;
            end else begin // записать найденную рубрику в список "другие рубрики"
               if other_rubriks=''
                  then other_rubriks:=temp
                  else other_rubriks:=format('%s | %s',[other_rubriks,
temp]);
            end;
      end;
   until position=0;
   temp:=rubriki; // проверить оставшийся кусок
   inc(counter);
   if temp=test_string
      then begin  // рубрика обнаружена
         pos_found := counter;
         result:=true
      end else begin // записать найденную рубрику в список "другие рубрики"
         if other_rubriks=''
            then other_rubriks:=temp
            else other_rubriks:=format('%s | %s',[other_rubriks, temp]);
      end;
end;

procedure TForm1.ExportHTML(all:boolean);
var oldLevel:integer;
    curNode:TTreeNode;
    x, y: integer;
    pos_found:integer;
    file_content_name: string;
    special_rubrika:boolean;
```

```
    stroka,stroka_spec,other_rubriks:string;
    Flags: OLEVariant;
    One_Volume_Content: TStringList;
    article_number: integer;
    rubrika_name: string;
    year_vol_str, str_vol : string;
    CurElement : ContentElement;
begin
    curNode:=single_volume_tree.TopItem;
    oldLevel:=-1;
    One_Volume_Content := TStringList.Create;
    if all then begin
       with VolumeContent[1] do begin
          if (year = '1992') and  (volume = '05') then begin
             year_vol_str := '1992_05_06';
             str_vol := '05-06';
          end else begin
             year_vol_str := year + '_' + volume;
             str_vol := volume;
          end;
          One_Volume_Content.Add(format('<ul><h1 STYLE="{clear: left; PAGE-
BREAK-BEFORE: always}"><a href="%1:s/%2:s.djvu" name="nomer_%2:s">Наука и
Жизнь №%0:s, %1:s</a></h1></ul>',[str_vol,year,year_vol_str]));
          if FileExists(prog_path+format('face/%0:s.jpg',[year_vol_str]))
             then One_Volume_Content.Add(format('<a
href="%0:s/%1:s.djvu"><img src="face/%1:s.jpg" align=right hspace=10
border=0></a>',[year, year_vol_str]))
             else One_Volume_Content.Add(format('<a
href="%0:s/%1:s.djvu"><img src="face/unknown.jpg" align=right hspace=10
border=0></a>',[year, year_vol_str]));
       end;
    end else begin
       One_Volume_Content.Add('<LINK REL="stylesheet" TYPE="text/css"
HREF="all_content.css" TITLE="style">');
    end;
    while curNode <> nil do begin
       if oldLevel<curNode.Level then begin // проставить <ul>
          for x:=1 to curNode.Level-oldLevel do begin
             One_Volume_Content.Add('<ul>');
          end;
       end;
       if oldLevel>curNode.Level then begin // проставить </ul>
          for x:=1 to oldLevel-curNode.Level do begin
             One_Volume_Content.Add('</ul>');
          end;
       end;
       // тут будет передаваться инфа о том, какая рубрика в данной статье является
сборной.
       // передаваться в виде s1 (s2, s3, s4, s5 и т.д.)
       if pos('_',curNode.Text)=0 then begin
          article_number := StrToInt(curNode.Text);
```

```
          rubrika_name :='';
       end else begin
          article_number :=
strToInt(copy(CurNode.Text,1,pos('_',curNode.Text)-1));
          rubrika_name :=
copy(CurNode.Text,pos('_',curNode.Text)+1,length(curnode.text)-
pos('_',curNode.Text)+1);
       end;
       with VolumeContent[article_number] do begin
          CurElement := ContentElement.Create(rubrika, author, title, page,
link, volume, year);
          if (year = '1992') and  (volume = '05') then begin
             year_vol_str := '1992_05_06';
             str_vol := '05-06';
          end else begin
             year_vol_str := year + '_' + volume;
             str_vol := volume;
          end;
          if rubrika_name <> '' then begin
             special_rubrika:=true; // рубрика относится к списку специальных
             compare_rubriks(rubrika,rubrika_name,other_rubriks,pos_found);
             stroka_spec:=rubrika_name; // запомнить название спецрубрики, т.к.
рубрика может быть сложносочинённой
          end else begin
             special_rubrika:=false;
          end;
          if chk_debug_mode.Checked then begin // режим вывода отладочного
оглавления
             if special_rubrika and (curNode.Level=0) then begin
                stroka:=format('[%s] <font color=red> - сборная
рубрика</font>',[stroka_spec])
             end else begin; // вывести всю инфу по ссылке
                stroka:='';
                if (rubrika <> '') then stroka := '[<font color=blue>' +
rubrika + '</font>] ';
                if (author <> '') then stroka := stroka + '(<font
color=green>' + author + '</font>) – ';
                if (title <> '') then stroka := stroka + title + ' ';
                if stroka = '' then stroka := '<font color=blue><i>нет
информации</i></font> ';
                if (link <> '')
                   then stroka := stroka + '(стр.'+ page + ', связь "<font
color=red>' + link + '</font>")'
                   else stroka := stroka + '(стр.'+ page + ')';
             end;
          end else begin // обычный режим вывода оглавления
             if all then begin // вывод сборного оглавления
                if special_rubrika then begin // спецрубрика
                   if (curNode.Level=0) then // спецрубрика в корне
                      stroka:=format('[<font
color=blue>%s</font>]',[stroka_spec])
```

```
                    else begin // статья в спецрубрике
                       if stroka_spec <> rubrika then begin // статья со
сложносочинённой рубрикой
                          if (author='') then begin // other_rubriks
                             if (title='')
                                then stroka:=format('[<font
color=blue>%s</font>]',[other_rubriks])
                                else stroka:=format('[<font
color=blue>%s</font>] — %s',[other_rubriks, title]);
                          end else begin
                             if (title='')
                                then stroka:=format('[<font
color=blue>%s</font>] (<font color=green>%s</font>)',[other_rubriks,
author])
                                else stroka:=format('[<font
color=blue>%s</font>] (<font color=green>%s</font>) — %s',[other_rubriks,
author, title]);
                          end;
                       end else begin
                          if (author='') then begin
                             if (title='')
                                then stroka:=stroka_spec
                                else stroka:=title;
                          end else begin
                             if (title='')
                                then stroka:=format('(<font
color=green>%s</font>)',[author])
                                else stroka:=format('(<font
color=green>%s</font>) — %s',[author, title]);
                          end;
                       end;
                    end;
                 end else begin // обычная рубрика
                    if (rubrika='') then begin
                       if (author='') then begin
                          if (title='') then stroka:='-=*=-' // заменитель для
пустой ссылки
                          else stroka:=title;
                       end else begin
                          if (title='') then stroka:=format('(<font
color=green>%s</font>)',[author])
                          else stroka:=format('(<font color=green>%s</font>)
— %s',[author, title]);
                       end;
                    end else begin // есть название рубрики
                       if (author='') then begin // но нет автора
                          if (title='') then stroka:=format('[<font
color=blue>%s</font>]',[rubrika])
                          else stroka:=format('[<font color=blue>%s</font>] —
%s',[rubrika,title]);
                       end else begin
```

```
                        if (title='') then stroka:=format('(<font
color=green>%s</font>)',[author])
                        else stroka:=format('(<font color=green>%s</font>)
— %s',[author, title]);
                     end;
                  end;
               end;
            end else begin // вывод оглавления для одного номера
               if special_rubrika then begin // спецрубрика
                  if (curNode.Level=0) then // спецрубрика в корне
                     stroka:=format('[%s]',[stroka_spec])
                  else begin // статья в спецрубрике
                     if stroka_spec <> rubrika then begin // статья со
сложносочинённой рубрикой
                        if (author='') then begin // other_rubriks
                           if (title='')
                              then stroka:=format('[%s]',[other_rubriks])
                              else stroka:=format('[%s] —
%s',[other_rubriks, title]);
                        end else begin
                           if (title='')
                              then stroka:=format('[%s] —
%s',[other_rubriks, author])
                              else stroka:=format('[%s] — %s —
%s',[other_rubriks, author, title]);
                        end;
                     end else begin
                        if (author='') then begin
                           if (title='')
                              then stroka:=stroka_spec
                              else stroka:=title;
                        end else begin
                           if (title='')
                              then stroka:=author
                              else stroka:=format('%s — %s',[author,
title]);
                        end;
                     end;
                  end;
               end else begin // обычная рубрика
                  if (rubrika='') then begin
                     if (author='') then begin
                        if (title='') then stroka:='-=*=-' // заменитель для
пустой ссылки
                        else stroka:=title;
                     end else begin
                        if (title='') then stroka:=author
                        else stroka:=format('%s — %s',[author, title]);
                     end;
                  end else begin // есть название рубрики
                     if (author='') then begin // но нет автора
```

```
                        if (title='') then stroka:=format('[%s]',[rubrika])
                        else stroka:=format('[%s] – %s',[rubrika,title]);
                     end else begin
                        if (title='') then stroka:=author
                        else stroka:=format('%s – %s',[author, title]);
                     end;
                  end;
               end;
            end;
            if all
               then stroka:=format('<li><a
href="%0:s/%1:s.djvu#%2:s">%3:s</a></li>',[year,year_vol_str,page,stroka])
               else stroka:=format('<li><a
href="#%s">%s</a></li>',[page,stroka]);
            One_Volume_Content.Add(stroka);
            CurElement.Free;
         end;
         oldLevel:=curNode.Level;
         curNode:=curNode.GetNext;
      end;
      for x:=1 to oldLevel+1 do begin // проставить завершающие </ul>
         One_Volume_Content.Add('</ul>');
      end;
      //if all then begin
         One_Volume_Content.SaveToStream(AllContent);
      {end else begin
file_content_name:=prog_path+edt_volumes.Items[edt_volumes.ItemIndex]+'.htm'
;
         One_Volume_Content.SaveToFile(file_content_name);
     // открыть созданный Html во встроенном редакторе
         Flags := 0;
         html_preview.Navigate(WideString(file_content_name), Flags, Flags,
Flags, Flags);
         if chk_embed_bookmarks.Checked then begin
            ShellExecute(GetDesktopWindow(), 'open', PChar(prog_path +
'make.bat'),
            PChar(file_content_name + ' ' +
edt_volumes.Items[edt_volumes.ItemIndex] + '.djvu ' +
edt_volumes.Items[edt_volumes.ItemIndex] + '.new.djvu'),
            nil, SW_HIDE);
         end;
      end;  }
      One_Volume_Content.Free;
end;

procedure TForm1.CreateTOC; // создание содержания для глобального оглавления по
номерам
                            // используется инфа из edt_volumes
                            // результат записывается в
file_Global_Volumes_Content
```

```
Const min_vol = 1;
      max_vol = 12;
      test_vol = max_vol + 1;
var MyContentTOC:TStringList;
    x,y,year_count:integer;
    user_year, user_vol:integer;
    str_vol, img_vol: string;
    style:string;
    all_volumes:array[min_year..max_year, min_vol..max_vol+1] of boolean;
    djvu_name:string;
    year_vol_str:string;
begin
    MyContentTOC:=TStringList.Create;
    FillChar(all_volumes,sizeof(all_volumes),0); // инициализировать таблицу номеров
    for x := 0 to edt_volumes.Items.Count-1 do begin // загрузить инфу по номерам и годам
       try
          user_year := StrToInt(copy(edt_volumes.Items[x],1,4));
          user_vol := StrToInt(copy(edt_volumes.Items[x],6,2));
       except
          ShowMessageInLog('----', '--', '---', format('Год или номер журнала являются некорректными (%s)',[edt_volumes.Items[x]]));
       end;
       if (user_year >= min_year) and (user_year <= max_year) and
       (user_vol >= min_vol) and (user_vol <= max_vol)
       then begin
          all_volumes[user_year, user_vol] := true; // есть конкретный номер
          all_volumes[user_year, test_vol] := true; // есть хоть один номер за определённый год
       end;
    end;
    // Создать заголовок
    MyContentTOC.Add('<LINK REL="stylesheet" TYPE="text/css" HREF="all_content.css" TITLE="style">');
    MyContentTOC.Add(format('<ul><H1>Наука и Жизнь %d - %d</H1>',[min_year, max_year]));
    if chk_debug_mode.Checked then begin
       MyContentTOC.Add('<div align=right><font size=-1><i>Оглавление создано в режиме проверки, формат статей:');
       MyContentTOC.Add('<br><b>[<font color=blue>Рубрика 1 | Рубрика 2</font>] - (<font color=green>Автор</font>) - Название статьи (страница, <font color=red>связь</font>)</b></i></font></div>');
    end;
    // создать оглавление первого уровня (таблица номеров)
    MyContentTOC.Add('<table class="toc" style="font-size=-2" width=100% height=80% cellspacing=0 cellpadding=0>');
    MyContentTOC.Add('<tr align=center><td class="hd"> </td>');
    for y := min_vol to max_vol do begin
       if y<10
          then MyContentTOC.Add(format('<td class="hd">№0%d</td>',[y]))
```

```
          else MyContentTOC.Add(format('<td class="hd">№%d</td>',[y]));
    end;
    MyContentTOC.Add('</tr>');
    for x := max_year downto min_year do begin
       if (x mod 2) = 0 then style := 'odd' else style := 'even';
       if all_volumes[x,test_vol]
          then MyContentTOC.Add(format('<tr align=center class="%0:s"><td
class="toc"><a href="#year_%1:d">%1:d год</a></td>',[style,x]))
          else MyContentTOC.Add(format('<tr align=center class="%0:s"><td
class="toc">%1:d год</td>',[style,x]));
       for y := min_vol to max_vol do begin
          if y<10
             then year_vol_str := format('%d_0%d',[x,y])
             else year_vol_str := format('%d_%d',[x,y]);
          if (year_vol_str = '1992_05') and (all_volumes[x,y]) then begin //
сдвоенный номер 1992 05-06
             year_vol_str := '1992_05_06';
             MyContentTOC.Add(format('<td class="toc" colspan=2><a
href="#nomer_%s">-=&gt;&lt;=-</a></td>',[year_vol_str]));
             continue; // перейти к следующему номеру
          end;

          if (x = 1992) and (y = 6) and (all_volumes[x,y-1]) then continue;
// пропустить номер 1992 6

          if all_volumes[x,y] then begin
             MyContentTOC.Add(format('<td class="toc"><a href="#nomer_%s">-
=&gt;&lt;=-</a></td>',[year_vol_str]))
          end else begin
             djvu_name := format('%s%d/%s.djvu',[prog_path,x,year_vol_str]);
             if (chk_debug_mode.checked) and (fileexists(djvu_name)) // в
режиме проверки для имеющихся в  DJVU  номеров создавать ссылки в таблице номеров
                then MyContentTOC.Add(format('<td class="toc"><a
href="%s">djvu</a></td>',[djvu_name]))
                else MyContentTOC.Add('<td class="toc"> </td>');
          end;
       end;
       MyContentTOC.Add('</tr>');
    end;
    MyContentTOC.Add('</table></ul>');

    // создать оглавление второго уровня (годовые подшивки)
    MyContentTOC.Add('<ul>');
    year_count:=0;
    for x:=max_year downto min_year do begin
       if all_volumes[x, test_vol] then begin // если есть номера для данного года
          inc(year_count);
          if (year_count mod 2) = 1
             then MyContentTOC.Add(format('<H2 STYLE="{clear: left; PAGE-
BREAK-BEFORE: always;}">Наука и Жизнь, %0:d год</H2><A
NAME="year_%0:d"></A>',[x]))
```

```
            else MyContentTOC.Add(format('<H2 STYLE="{clear: left;}">Наука
и Жизнь, %0:d год</H2><A NAME="year_%0:d"></A>',[x]));
          for y:=min_vol to max_vol do begin
             if y=7 then MyContentTOC.Add('<br STYLE="{clear: left;}">');
             if (x = 1992) and (y = 6) and (all_volumes[x,y-1]) then
continue; // пропустить номер 1992 6
             if all_volumes[x,y] then begin // если есть конкретный номер, вывести
для него ссылку
                if y<10
                   then year_vol_str := format('%d_0%d',[x,y])
                   else year_vol_str := format('%d_%d',[x,y]);

                if y < 10 then str_vol := format('0%d',[y]) else str_vol :=
format('%d',[y]);

                if (year_vol_str = '1992_05') then begin
                   str_vol := '05-06';
                   year_vol_str := '1992_05_06';
                end;

                if
FileExists(format('%sface/%s_mini.jpg',[prog_path,year_vol_str]))
                   then img_vol := format('%s_mini.jpg',[year_vol_str])
                   else img_vol:='unknown_mini.jpg';
                MyContentTOC.Add(format(
                   '<table border=0 align=left><tr align=center><td><a
href="#nomer_%3:s">'+
                   '<img src="face/%2:s" border=0></a></td></tr><tr
align=center><td>'+
                   '<a href="#nomer_%3:s">№%1:s,
%0:d</a></td></tr></table>',
                   [x,str_vol,img_vol,year_vol_str]));
             end else begin // номера нет - вывести затычку
                if y < 10 then str_vol := format('0%d',[y]) else str_vol :=
format('%d',[y]);
                img_vol := 'absent_mini.gif';
                MyContentTOC.Add(format(
                   '<table border=0 align=left><tr align=center><td>'+
                   '<img src="face/%2:s" border=0></td></tr><tr
align=center><td>'+
                   ' </td></tr></table>',
                   [x,str_vol,img_vol]));
             end;
          end;
       end;
    end;
    MyContentTOC.Add('</ul>');
    MyContentTOC.SaveToStream(file_Global_Volumes_Content);
    MyContentTOC.Free;
end;
```

```
procedure TForm1.Global_Content_Create(Sender: TObject);
  Procedure per_volumes; // создание глобального оглавления по номерам
(all_content.htm)
  var Values: OLEVariant;
    ISheet: Excel8TLB._Worksheet;
    IRange: Excel8TLB.Range;
    i,z,j: integer;
    Flags: OLEVariant;
  begin
  if Assigned(IWorkbook) then
    //try
      ISheet := IWorkbook.Worksheets.Item['Содержание'] as
Excel8TLB._Worksheet;
file_Global_Volumes_Content:=TFileStream.Create(prog_path+'all_content.htm',
fmCreate);
      AllContent:=TMemoryStream.Create;
      tree_preview.items.Clear;
      try
        local_progress.Min:=0;
        local_progress.max:=edt_volumes.Items.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        for z:=edt_volumes.Items.Count-1 downto 0 do begin // пройтись по всем
выпускам
           local_progress.StepIt;
           IRange := ISheet.Range['vol_'+edt_volumes.Items[z], EmptyParam];
           Values := IRange.Value;
           SetLength(VolumeContent,IRange.Rows.Count);
           for i := 1 to IRange.Rows.Count do begin
             with VolumeContent[i-1] do begin
                rubrika := Values[i, 1];
                author := Values[i, 2];
                title := Values[i, 3];
                page := Values[i, 4];
                link := Values[i, 5];
                volume := Values[i, 6];
                year := Values[i, 7];
                used_as_slave := false;
                for j :=1 to max_complex_rubriks do used_as_article[j] :=
false;
                //stroka:= format('%s | %s | %s | %s | %s | %s | %s',
[rubrika, author, title, page, link, volume, year]);
             end;
           end;
           MakeContent;
           ExportHTML(make_global);
        end;
        local_progress.StepIt;
        CreateTOC;
        local_progress.Position:=local_progress.Min;
        file_Global_Volumes_Content.CopyFrom(AllContent,0); // скопировать всё
```

```
оглавление в файл с оглавлением
      finally
        IRange := nil;
        ISheet := nil;
        AllContent.Free;
        file_Global_Volumes_Content.Free;
        tree_preview.Items.Clear;
        single_volume_tree.Items.Clear;
        // открыть созданный Html во встроенном редакторе
        Flags := 0;
        html_preview.Navigate(WideString(prog_path+'all_content.htm'),
Flags, Flags, Flags, Flags);
      end;
    //except
      //raise Exception.Create('Не могу прочитать данные в массив!');
    //end;
  end;
  Procedure per_authors; // создание глобального авторского указателя
(all_authors.htm)
//  type all_triplets_element:
  var Values: OLEVariant;
    ISheet, ISheet_auth_index, ISheet_authors, ISheet_triplet:
Excel8TLB._Worksheet;
    articles: array of TMyContent;
    auth_index:array of index_element;
    authors: array of authors_element;
    triplets: array of triplets_element;
    odd_even:boolean;
    min_tripl, max_tripl: integer; // индексы для простановки диапазонов триплетов
    toc_triplets:array [ord('A')..ord('Я'), 0..max_toc_triplet] of string; //
столбец '0' - признак, что столбец заполнен
    prevIndex, page_counter, rows_counter : integer;

    vYear, vMonth, vAuthor, prevValue, vRubrika: OLEVariant;
    x,first,last:integer;
    stroka, prevStroka:string;
    auth_index_counter, // счётчик авторских статей
    authors_counter, // счётчик уникальных авторов
    triplet_counter:integer; // счётчик триплетов
    Authors_TOC:TStringList; // место для создания главной таблицы (верхний уровень)
    Authors_Names:TStringList; // место для создания списка фамилий (средний
уровень)
    Authors_Articles:TStringList;// место для создания блоков ссылок на статьи
(нижний уровень)
    IRange: Excel8TLB.Range;
    i,j,z: integer;
    Flags: OLEVariant;
    year_vol_str, str_vol: string;

  begin
  if Assigned(IWorkbook) then
```

```
    try
      ISheet := IWorkbook.Worksheets.Item['Содержание'] as
Excel8TLB._Worksheet;
      ISheet_auth_index := IWorkbook.Worksheets.Item['Авторы'] as
Excel8TLB._Worksheet;
      ISheet_authors := IWorkbook.Worksheets.Item['Авторы (свод)'] as
Excel8TLB._Worksheet;
      ISheet_triplet := IWorkbook.Worksheets.Item['Триплеты'] as
Excel8TLB._Worksheet;

      Authors_TOC := TStringList.Create; // место для создания главной таблицы
(верхний уровень)
      Authors_Names := TStringList.Create; // место для создания списка фамилий
(средний уровень)
      Authors_Articles := TStringList.Create; // место для создания блоков ссылок
на статьи (нижний уровень)

file_All_Authors_Index:=TFileStream.Create(prog_path+'all_authors.htm',
fmCreate);
      try
        // создать конечный перечень - имя автора (заголовок, ссылка вида  #author_),
перечень статей этого автора.

        // загрузить лист "Содержание"
        IRange := ISheet.UsedRange[0];
        Values := IRange.Value;
        SetLength(Articles,IRange.Rows.Count+1);
        local_progress.Min:=0;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Загрузка листа "Содержание"...';
        for i := 1 to IRange.Rows.Count do begin
          local_progress.stepit;
          with Articles[i] do begin
             rubrika := Values[i, 1];
             author := Values[i, 2];
             title := Values[i, 3];
             page := Values[i, 4];
             link := Values[i, 5];
             volume := Values[i, 6];
             year := Values[i, 7];
          end;
        end;

        // загрузить лист "Авторы"
        IRange := ISheet_auth_index.UsedRange[0];
        Values := IRange.Value;
        Setlength(auth_index,IRange.Rows.Count+1);
        local_progress.Min:=0;
        local_progress.max:=IRange.Rows.Count;
```

```
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Загрузка листа "Авторы"...';
        for i := 1 to IRange.Rows.Count do begin
          local_progress.stepit;
          with auth_index[i] do begin
             name := Values[i, 1];
             index := Values[i, 2];
          end;
        end;

        // загрузить лист "Авторы (свод)"
        IRange := ISheet_authors.UsedRange[0];
        Values := IRange.Value;
        Setlength(authors,IRange.Rows.Count+1);
        local_progress.Min:=0;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Загрузка листа "Авторы (свод)"...';
        for i := 1 to IRange.Rows.Count do begin
          local_progress.stepit;
          with authors[i] do begin
             name := Values[i, 1];
             index := Values[i, 2];
             count:= Values[i, 3];
             triplet:= Values[i, 4];
             surname:= Values[i, 5];
          end;
        end;

        // загрузить лист "Триплеты"
        IRange := ISheet_triplet.UsedRange[0];
        Values := IRange.Value;
        Setlength(triplets,IRange.Rows.Count+1);
        local_progress.Min:=0;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Загрузка листа "Триплеты"...';
        for i := 1 to IRange.Rows.Count do begin
          local_progress.stepit;
          with triplets[i] do begin
             triplet := Values[i, 1];
             index := Values[i, 2];
             count:= Values[i, 3];
             letter:= Values[i, 4];
             if not (letter[1] in ['А'..'Я']) then letter:='Ъ'; // подменить
ошибочные триплеты на Ъ
          end;
        end;
```

```
        Authors_TOC.add('<LINK REL="stylesheet" TYPE="text/css"
HREF="all_content.css" TITLE="style">');
        Authors_TOC.add('<ul>'); // ведущий UL
         // построить список всех триплетов
        local_progress.Min:=0;
        local_progress.max:=length(triplets)-1;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Построение списка триплетов...';
        x:=0;
        //odd_even:=true;
        // загрузить массив триплетов
        for i:= ord('A') to ord('Я') do begin
           for x:=0 to max_toc_triplet do begin
              toc_triplets[i,x] := ''; // обнулить массив.
           end;
        end;
        { найти диапазон для очередной буквы
     по найденному диапазону триплетов построить 8 интервалов
     если кол-во триплетов меньше или равно max_toc_triplet - "тупой" алгоритм
     если кол-во триплетов больше max_toc_triplet - "хитрый" алгоритм
     приписать триплетам соответствующие индексы интервалов
        }
        first:=1; prevStroka:=triplets[1].letter;
        for i := 1 to length(triplets)-1 do begin
           if (prevStroka <> triplets[i].letter) then begin // пошла новая
буква
              last:=i-1; // конец диапазона - предыдущий триплет
              toc_triplets[ord(prevStroka[1]),0] := '*'; // проставить признак,
что такая буква есть.
              if (last-first+1)<=max_toc_triplet then begin // кол-во триплетов
меньше или равно max_toc_triplet - "тупой" алгоритм
                 for x:=1 to last-first+1 do begin
toc_triplets[ord(prevStroka[1]),x]:=triplets[first+x-1].triplet;
                    triplets[first+x-1].toc_triplets_index:=x; // приписать
триплетам соответствующие индексы интервалов
                 end;
              end else begin // хитрый алгоритм
                 for x:=1 to max_toc_triplet do begin
                    min_tripl:=first+round((last-
first+1)/max_toc_triplet*(x-1));
                    max_tripl:=first+round((last-
first+1)/max_toc_triplet*(x))-1;
                    for z:=min_tripl to max_tripl do begin
                       triplets[z].toc_triplets_index:=x; // приписать
триплетам соответствующие индексы интервалов
                    end;
                    if min_tripl = max_tripl
                       then toc_triplets[ord(prevStroka[1]),x] :=
triplets[min_tripl].triplet
```

```
                        else toc_triplets[ord(prevStroka[1]),x] :=
triplets[min_tripl].triplet + ' - ' + triplets[max_tripl].triplet;
                    end;
                end;
                first:=i; // запомнить начало нового диапазона
            end;
            prevStroka := triplets[i].letter;
        end;
        i:=length(triplets);
        last:=i-1; // конец диапазона - последний элемент
        toc_triplets[ord(prevStroka[1]),0] := '*'; // проставить признак, что
такая буква есть.
        if (last-first+1)<=max_toc_triplet then begin // кол-во триплетов меньше
или равно max_toc_triplet - "тупой" алгоритм
            for x:=1 to last-first+1 do begin
toc_triplets[ord(prevStroka[1]),x]:=triplets[first+x-1].triplet;
                triplets[first+x-1].toc_triplets_index:=x; // приписать триплетам
соответствующие индексы интервалов
            end;
        end else begin // хитрый алгоритм
            for x:=1 to max_toc_triplet do begin
                min_tripl:=first+round((last-first+1)/max_toc_triplet*(x-1));
                max_tripl:=first+round((last-first+1)/max_toc_triplet*(x))-1;
                for z:=min_tripl to max_tripl do begin
                    triplets[z].toc_triplets_index:=x; // приписать триплетам
соответствующие индексы интервалов
                end;
                if min_tripl = max_tripl
                    then toc_triplets[ord(prevStroka[1]),x] :=
triplets[min_tripl].triplet
                    else toc_triplets[ord(prevStroka[1]),x] :=
triplets[min_tripl].triplet + ' - ' + triplets[max_tripl].triplet;
            end;
        end;
        odd_even := true;
        Authors_TOC.add('<h1>Авторский указатель</h1>');
        Authors_TOC.Add('<table class="toc" style="font-size=-2" width=100%
cellspacing=0 cellpadding=5>');
        for i:= ord('А') to ord('Я') do begin
            if toc_triplets[i,0] = '' then continue; // пропустить пустые буквы
            odd_even := not odd_even;
            if odd_even
                then Authors_TOC.Add('<tr class="even"><td class="hd">' +
chr(i) + '</td>')
                else Authors_TOC.Add('<tr class="odd"><td class="hd">' +
chr(i) + '</td>');
            for x:=1 to max_toc_triplet do begin
                if (toc_triplets[i,x] <> '')
                    then Authors_TOC.Add(format('<td class="toc"><a
href="#triplet_%1:d_%2:d">%0:s</a></td>',[toc_triplets[i,x],i,x]))
                    else Authors_TOC.Add('<td class="toc"> </td>');
```

```
            end;
            Authors_TOC.Add('</tr>');
            local_progress.StepIt;
         end;
         Authors_TOC.Add('</table>');

         // построить список всех фамилий
         Authors_Names.add('<div STYLE="{PAGE-BREAK-BEFORE:
always}"></div>');
         local_progress.Min:=0;
         local_progress.max:=length(triplets)-1;
         local_progress.Position:=local_progress.Min;
         local_progress.Step := 1;
         statusbar1.simpletext := 'Построение списка фамилий...';
         z := 0;
         page_counter := 0;
         odd_even := false;
         i := 1; prevStroka := triplets[i].letter + '_' +
IntToStr(triplets[i].toc_triplets_index);
         Authors_Names.Add(format('<h2><a
name="triplet_%1:d_%2:d">%0:s</a></h2>',[toc_triplets[ord(triplets[i].letter
[1]),triplets[i].toc_triplets_index],ord(triplets[i].letter[1]),triplets[i].
toc_triplets_index]));
         inc(page_counter,27); // высота заголовка - 27 пунктов
         Authors_Names.Add('<table class="toc" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5><tr class="odd">');
         for i:= 1 to length(triplets)-1 do begin
            if (prevStroka <> triplets[i].letter + '_' +
IntToStr(triplets[i].toc_triplets_index)) then begin // пошёл новый интервал
               for x:=z to 2 do begin
                  Authors_Names.Add('<td class="toc"> </td>'); // дописать
пустые ячейки до трёх
               end;
               Authors_Names.Add('</tr></table>');
               if (page_counter>220-27) then begin // высота страницы - 290
пунктов
                  z := 0;
                  page_counter := 0;
                  odd_even:=false;
                  Authors_Names.Add(format('<h2 style="{page-break-before:
always}"><a
name="triplet_%1:d_%2:d">%0:s</a></h2>',[toc_triplets[ord(triplets[i].letter
[1]),triplets[i].toc_triplets_index],ord(triplets[i].letter[1]),triplets[i].
toc_triplets_index]));
               end else
                  Authors_Names.Add(format('<h2><a
name="triplet_%1:d_%2:d">%0:s</a></h2>',[toc_triplets[ord(triplets[i].letter
[1]),triplets[i].toc_triplets_index],ord(triplets[i].letter[1]),triplets[i].
toc_triplets_index]));
               inc(page_counter,27); // высота заголовка - 27 пунктов
               rows_counter := 0;
```

```
              Authors_Names.Add('<table class="toc" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5><tr class="odd">');
              z := 0;
              odd_even:=false;
           end;
           for x := triplets[i].index to (triplets[i].index +
triplets[i].count - 1) do begin // вывести все фамилии для триплета
              if z = 3 then begin // запустить новую строку
                 inc(page_counter,9);   // высота строки - 9 пунктов
                 inc(rows_counter);
                 z := 0;
                 if odd_even
                    then Authors_Names.Add('</tr><tr class="odd">')
                    else Authors_Names.Add('</tr><tr class="even">');
                 odd_even:=not odd_even;
              end;
              if (page_counter>220) and (rows_counter>1) then begin // высота
страницы - 290 пунктов
                 Authors_Names.Add('</tr></table>');
                 z := 0;
                 page_counter := 0;
                 odd_even:=false;
                 Authors_Names.Add(format('<h2 style="{page-break-before:
always}">%0:s</h2>',[toc_triplets[ord(triplets[i].letter[1]),triplets[i].toc
_triplets_index]]));
                 inc(page_counter,27); // высота заголовка - 27 пунктов
                 Authors_Names.Add('<table class="toc" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5><tr class="odd">');
              end;
              inc(z);
              Authors_Names.Add(format('<td class="toc" width="33%%"><a
href="#author_%1:d">%0:s</a></td>',[authors[x].name, x]));
           end;
           local_progress.StepIt;
           prevStroka := triplets[i].letter + '_' +
IntToStr(triplets[i].toc_triplets_index);
        end;
        Authors_Names.Add('</tr></table>');

        // построить список нижнего уровня
        local_progress.Min:=0;
        local_progress.max:=length(authors)-1;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Построение поавторского списка статей...';
        Authors_Articles.add('<div STYLE="{PAGE-BREAK-BEFORE:
always}"></div>');
        for i := 1 to length(authors)-1 do begin
           stroka := format('<h2><a name="author_%0:d">%1:s</a></h2>',[i,
authors[i].name]);;
           Authors_Articles.add(stroka);
```

```
           local_progress.StepIt;
           Authors_Articles.add('<ul>');
           for x:=authors[i].index + authors[i].Count - 1 downto
authors[i].index do begin // пройтись по всем записям индекса автора
              with articles[auth_index[x].index] do begin

                 if (year = '1992') and  (volume = '05') then begin
                    year_vol_str := '1992_05_06';
                    str_vol := '05-06';
                 end else begin
                    year_vol_str := year + '_' + volume;
                    str_vol := volume;
                 end;

                 stroka:=format('<li><a href="%0:s/%1:s.djvu#%2:s">%0:s
№%3:s – ',[year, year_vol_str, page, str_vol]);
                 if (rubrika <> '') then stroka := stroka + '[<font
color=blue>' + rubrika + '</font>] ';
                 if (author <> '') then stroka := stroka + '(<font
color=green>' + author + '</font>) – ';
                 if (title <> '') then stroka := stroka + title + ' ';
                 stroka := stroka + '</a>';

                 Authors_Articles.Add(stroka);
              end;
           end;
           Authors_Articles.add('</ul>');
        end;
        Authors_Articles.add('</ul>'); // конечный /UL
        statusbar1.simpletext:='Готово';

        Authors_TOC.SaveToStream(file_All_Authors_Index);
        Authors_Names.SaveToStream(file_All_Authors_Index);
        Authors_Articles.SaveToStream(file_All_Authors_Index);
      finally
        IRange := nil;
        ISheet := nil;
        ISheet_auth_index := nil;
        ISheet_authors := nil;
        ISheet_triplet := nil;
        setlength(articles,0);
        setlength(auth_index,0);
        setlength(authors,0);
        setlength(triplets,0);
        file_All_Authors_Index.Free;
        Authors_TOC.Free; // освободить место для создания главной таблицы (верхний
уровень)
        Authors_Names.Free; // освободить место для создания списка фамилий
(средний уровень)
        Authors_Articles.Free; // освободить место для создания блоков ссылок на
статьи (нижний уровень)
```

```
        {
    // открыть созданный Html во встроенном редакторе
        Flags := 0;
        html_preview.Navigate(WideString(prog_path+'all_content.htm'),
Flags, Flags, Flags, Flags);
        }
      end;
    except
      raise Exception.Create('Не могу прочитать данные в массив!');
    end;
  end;
  Procedure per_rubrics; // создание глобального рубрикатора (all_rubrics.htm)
  var Values: OLEVariant;
    ISheet, ISheet_rubr_index, ISheet_rubriks: Excel8TLB._Worksheet;
    articles: array of TMyContent;
    rubr_index: array of rubr_index_element;
    rubriks: array of rubriks_element;

    Sorted_rubriks:TStringList;
    Rubrik_Object:TRubr_Info;

    odd_even:boolean; // хрень для полосатых таблиц
    page_counter, rows_counter : integer;
    x:integer;
    user_max_year, user_min_year : integer; // пределы для отображения таблиц
ссылок
    stroka:string;
    Rubriks_TOC:TStringList; // место для создания главной таблицы (верхний уровень)
    Rubriks_Names:TStringList; // место для создания списка фамилий (средний
уровень)
    Rubriks_Articles:TStringList;// место для создания блоков ссылок на статьи
(нижний уровень)
    IRange: Excel8TLB.Range;
    i,z: integer;
    year_vol_str, str_vol:string;
    prev_group: integer;
    procedure show_table_rubrik(i:integer); // вывод таблицы со ссылками для
регулярных рубрик
       Const min_vol = 1;
             max_vol = 12;
             test_vol = max_vol + 1;
       var x,y:integer;
           user_year, user_vol:integer;
           style:string;
           all_volumes:array[min_year..max_year, min_vol..max_vol+1] of
integer;
               // если рубрики в номере нет, то значение равно нулю,
               // в противном случае - это номер страницы
               // последний столбец - количество номеров с рубриками в определённом
году.
    begin
```

```
    // определить min_year, max_year для конкретной рубрики
    //

    // Заполнить таблицу ссылок (all_volumes) для рубрики
    FillChar(all_volumes,sizeof(all_volumes),0); // инициализировать таблицу
номеров
    for x := rubriks[i].index to rubriks[i].index+rubriks[i].count-1 do
begin // прочесать все статьи для рубрики x
       user_year := strtoint(articles[rubr_index[x].index].year); //
заполучить год, в котором статья
       user_vol := strtoint(articles[rubr_index[x].index].volume); //
заполучить номер, в котором статья
       if (user_year >= min_year) and (user_year <= max_year) and
       (user_vol >= min_vol) and (user_vol <= max_vol)
       then begin
          if all_volumes[user_year, user_vol] = 0 then // если это первая статья
данной рубрики в номере
             all_volumes[user_year, user_vol] :=
strtoint(articles[rubr_index[x].index].page); // запомнить номер страницы
          inc(all_volumes[user_year, test_vol]); // увеличить счётчик количества
номеров в этом году
       end;
    end;

    // Создать заголовок таблицы ссылок для рубрики
    //Rubriks_Articles.Add(format('<H1 STYLE="{PAGE-BREAK-BEFORE:
always}">%s %d - %d</H1>',[rubriks[i].name, min_year, max_year]));
    Rubriks_Articles.Add('<table class="toc" style="font-size=-2" width=100%
cellspacing=0 cellpadding=0>');
    Rubriks_Articles.Add('<tr align=center><td class="hd"> </td>');
    for y := min_vol to max_vol do begin
       if y<10
          then Rubriks_Articles.Add(format('<td class="hd">№0%d</td>',[y]))
          else Rubriks_Articles.Add(format('<td class="hd">№%d</td>',[y]));
    end;
    Rubriks_Articles.Add('</tr>');
    // просчитать пределы годов отображения
    for x := max_year downto min_year do begin
       user_max_year := x;
       if (all_volumes[x, test_vol] >0) then break;
    end;
    for x := min_year to max_year do begin
       user_min_year := x;
       if (all_volumes[x, test_vol] >0) then break;
    end;
    // вывести основную часть таблицы
    for x := user_max_year downto user_min_year do begin
       if (x mod 2) = 0 then style := 'odd' else style := 'even'; // полосатая
таблица
       Rubriks_Articles.Add(format('<tr align=center class="%0:s"><td
class="toc">%1:d год</td>',[style,x]));
```

```
       for y := min_vol to max_vol do begin
          if y < 10
             then year_vol_str := format('%d_0%d',[x,y])
             else year_vol_str := format('%d_%d',[x,y]);
          if (year_vol_str = '1992_05') then begin // сдвоенный номер 1992 05-06
             year_vol_str := '1992_05_06';
             if (all_volumes[x,y]>0)
                then Rubriks_Articles.Add(format('<td class="toc"
colspan=2><a href="%0:d/%1:s.djvu#%2:d">-=&gt;&lt;=-</a></td>',[x,
year_vol_str, all_volumes[x,y]]))
                else Rubriks_Articles.Add('<td class="toc"
colspan=2> </td>');
             continue; // перейти к следующему номеру
          end;
          if (x = 1992) and (y = 6) then continue; // пропустить номер 1992 6
          if (all_volumes[x,y]>0)
             then Rubriks_Articles.Add(format('<td class="toc"><a
href="%0:d/%1:s.djvu#%2:d">-=&gt;&lt;=-</a></td>',[x, year_vol_str,
all_volumes[x,y]]))
             else Rubriks_Articles.Add('<td class="toc"> </td>');
       end;
       Rubriks_Articles.Add('</tr>');
    end;
    Rubriks_Articles.Add('</table>');

    end;
  begin // собственно начало процедуры per_rubriks
  if Assigned(IWorkbook) then
    try
      ISheet := IWorkbook.Worksheets.Item['Содержание'] as
Excel8TLB._Worksheet;
      ISheet_Rubr_index := IWorkbook.Worksheets.Item['Рубрики'] as
Excel8TLB._Worksheet;
      ISheet_Rubriks := IWorkbook.Worksheets.Item['Рубрики (свод)'] as
Excel8TLB._Worksheet;

      Rubriks_TOC := TStringList.Create; // место для создания главной таблицы
(верхний уровень)
      Rubriks_Names := TStringList.Create; // место для создания списка рубрик
(средний уровень)
      Rubriks_Articles := TStringList.Create; // место для создания блоков ссылок
на статьи (нижний уровень)

file_All_Rubriks_Index:=TFileStream.Create(prog_path+'all_rubriks.htm',
fmCreate);
      try
        // создать конечный перечень - имя автора (заголовок, ссылка вида #Rubrika_),
перечень статей в этой рубрике.

        // загрузить лист "Содержание"
        IRange := ISheet.UsedRange[0];
```

```
        Values := IRange.Value;
        SetLength(Articles,IRange.Rows.Count+1);
        local_progress.Min:=0;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Загрузка листа "Содержание"...';
        for i := 1 to IRange.Rows.Count do begin
          local_progress.stepit;
          with Articles[i] do begin
             rubrika := Values[i, 1];
             author := Values[i, 2];
             title := Values[i, 3];
             page := Values[i, 4];
             link := Values[i, 5];
             volume := Values[i, 6];
             year := Values[i, 7];
          end;
        end;

        // загрузить лист "Рубрики"
        IRange := ISheet_Rubr_index.UsedRange[0];
        Values := IRange.Value;
        Setlength(Rubr_index,IRange.Rows.Count+1);
        local_progress.Min:=0;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Загрузка листа "Рубрики"...';
        for i := 1 to IRange.Rows.Count do begin
          local_progress.stepit;
          with Rubr_index[i] do begin
             name := Values[i, 1];
             index := Values[i, 2];
          end;
        end;

        // загрузить лист "Рубрики (свод)"
        IRange := ISheet_Rubriks.UsedRange[0];
        Values := IRange.Value;
        Setlength(Rubriks,IRange.Rows.Count+1);
        local_progress.Min:=0;
        local_progress.max:=IRange.Rows.Count;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Загрузка листа "Рубрики (свод)"...';
        for i := 1 to IRange.Rows.Count do begin
          local_progress.stepit;
          with Rubriks[i] do begin
             name := Values[i, 1];
             index := Values[i, 2];
```

```
            count := Values[i, 3];
            common := Values[i, 4];
            group_number := Values[i, 5];
            number := i;
          end;
        end;

        // создать список рубрик, упорядоченный по алфавиту  "название рубрики" |
"Исходный номер рубрики"
        Sorted_rubriks := TStringList.Create;
        for i := 1 to length(rubriks)-1 do begin
           Rubrik_Object := TRubr_Info.Create(rubriks[i]);
           sorted_rubriks.AddObject(Rubrik_Object.info.name, Rubrik_Object);
        end;
        sorted_rubriks.Sort;

        // построить первую страницу (перечень групп рубрик и ссылка на алфавитный
список рубрик)

        Rubriks_TOC.add('<LINK REL="stylesheet" TYPE="text/css"
HREF="all_content.css" TITLE="style">');

        Rubriks_TOC.add('<ul>'); // ведущий UL
        odd_even := true;
        //Rubriks_TOC.add('<h1>Рубрикатор</h2>');
        Rubriks_TOC.add('<h2><a href="#Group_1">Группы рубрик</a></h2>');
        Rubriks_TOC.add('<table class="stealth" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5>');
        x := 1;
        repeat
           Rubriks_TOC.add('<tr align="center" valign="bottom"
class="even">');
           for i:=0 to 2 do begin
              if (x+i > length(Rubriks_Groups)-1) then continue;
              Rubriks_TOC.add(format('<td><a href="#Group_%0:d"><img
src="face/group_%0:d.png" border=0></a></td>', [x+i]));
           end;
           Rubriks_TOC.add('</tr><tr align="center" valign="top"
class="even">');
           for i:=0 to 2 do begin
              if (x+i > length(Rubriks_Groups)-1) then continue;
              Rubriks_TOC.add(format('<td><a href="#Group_%0:d">%1:s</a>',
[x+i, Rubriks_Groups[x+i].group]));
           end;
           Rubriks_TOC.add('</tr>');
           inc(x,3);
        until (x > length(Rubriks_Groups)-1);
        Rubriks_TOC.add('</table>');
        Rubriks_TOC.add('<h2><a href="#alphabet">Алфавитный список
рубрик</a></h2>');
        Rubriks_TOC.add('<table class="stealth" style="font-size=-2"
```

```
width=100% cellspacing=0 cellpadding=5>');
        Rubriks_TOC.add('<tr class="even"><td>');

        // построить список всех рубрик (группированный по темам)
        Rubriks_Names.add('<div STYLE="{PAGE-BREAK-BEFORE:
always}"></div>');
        local_progress.Min:=0;
        local_progress.max:=length(rubriks)-1;
        local_progress.Position:=local_progress.Min;
        local_progress.Step := 1;
        statusbar1.simpletext := 'Построение списка рубрик по группам...';
        z := 0;
        page_counter := 0;
        odd_even := false;
        (*// на странице уже есть заголовок и список групп - надо учесть.
        inc(page_counter, 27); // высота заголовка - 27 пунктов
        inc(page_counter, 5 * (length(Rubriks_Groups)-1) ); // строки в списке
- 5 пунктов
        *)
        prev_group:=0; // первая группа

        Rubriks_Names.Add('<table class="toc" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5><tr class="odd">');
        for i:= 1 to length(rubriks)-1 do begin
           if (Rubriks[i].group_number<>prev_group) then begin // сменилась
группа рубрик
              if (rows_counter>0) then // игнорируем пустые таблицы
                 for x:=z to 2 do begin // закончить последнюю строку в таблице
                    Rubriks_Names.Add('<td class="toc"> </td>'); //
дописать пустые ячейки до трёх
                 end;
              Rubriks_Names.Add('</tr></table>'); // закрыть предыдущую таблицу
              z := 0; // снова инициализировать счётчик столбцов
              rows_counter := 0;
              //page_counter := 0; // снова инициализировать счётчик высоты
              odd_even := false; // снова инициализировать полосатость
              prev_group := Rubriks[i].group_number; // сменить текущую группу
              Rubriks_Names.add(format('<h2 style="{page-break-before:
always}"><a name="Group_%d">%s</a></h2>',[prev_group,
Rubriks_Groups[prev_group].group]));
              //inc(page_counter,27); // высота заголовка - 27 пунктов
              page_counter := 27;
              Rubriks_Names.Add('<table class="toc" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5><tr class="odd">'); // открыть новую
таблицу
           end;

           if z = 3 then begin // запустить новую строку
              inc(page_counter,9);  // высота строки - 9 пунктов
              inc(rows_counter);
              z := 0;
```

```
              if odd_even
                 then Rubriks_Names.Add('</tr><tr class="odd">')
                 else Rubriks_Names.Add('</tr><tr class="even">');
              odd_even:=not odd_even;
           end;
           if (page_counter>210) and (rows_counter>1) then begin // высота
страницы - 210 пунктов
              Rubriks_Names.Add('</tr></table>');
              z := 0;
              rows_counter := 0;
              page_counter := 0;
              odd_even:=false;
              Rubriks_Names.add(format('<h2 style="{page-break-before:
always}">%s</h2>',[Rubriks_Groups[prev_group].group])); // повторить заголовок
              inc(page_counter,27); // высота заголовка - 27 пунктов
              Rubriks_Names.Add('<table class="toc" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5><tr class="odd">');
           end;
           inc(z);
           Rubriks_Names.Add(format('<td class="toc" width="33%%"><a
href="#Rubrika_%1:d">%0:s</a></td>',[Rubriks[i].name, i]));
           local_progress.StepIt;
        end;
        for x:=z to 2 do begin // закончить последнюю строку в таблице
           Rubriks_Names.Add('<td class="toc"> </td>'); // дописать пустые
ячейки до трёх
        end;
        Rubriks_Names.Add('</tr></table>');

        // построить список всех рубрик, без разбивки по группам рубрик (просто по
алфавиту)

        local_progress.Min:=0;
        local_progress.max:=sorted_rubriks.Count-1;
        local_progress.Position:=local_progress.Min;
        local_progress.Step := 1;
        statusbar1.simpletext := 'Построение алфавитного списка рубрик...';
        z := 0;
        page_counter := 0;
        odd_even := false;
        prev_group :=ord(upcase(sorted_rubriks.Strings[0][1]));
        Rubriks_Names.add('<h2 style="{page-break-before: always}"><a
name="alphabet">Алфавитный список рубрик</a></h2>');
        inc(page_counter,27); // высота заголовка - 27 пунктов
        Rubriks_Names.add(format('<h2><a
name="abc_%d">%s</a></h2>',[prev_group,
upcase(sorted_rubriks.Strings[0][1])]));
        inc(page_counter,27); // высота заголовка - 27 пунктов
        Rubriks_TOC.add(format('Рубрики на: <a
href="#abc_%d">%s</a>',[ord(upcase(sorted_rubriks.Strings[0][1])),upcase(sor
ted_rubriks.Strings[0][1])]));
```

```
        Rubriks_Names.Add('<table class="toc" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5><tr class="odd">');
        for i:= 0 to sorted_rubriks.Count-1 do begin
           if (ord(upcase(sorted_rubriks.Strings[i][1]))<>prev_group) then
begin // сменилась группа рубрик
              Rubriks_TOC.add(format(', <a
href="#abc_%d">%s</a>',[ord(upcase(sorted_rubriks.Strings[i][1])),upcase(sor
ted_rubriks.Strings[i][1])]));
              for x:=z to 2 do begin // закончить последнюю строку в таблице
                 Rubriks_Names.Add('<td class="toc"> </td>'); // дописать
пустые ячейки до трёх
              end;
              Rubriks_Names.Add('</tr></table>'); // закрыть предыдущую таблицу
              //inc(page_counter,9);  //  высота строки - 9 пунктов
              z := 0; // снова инициализировать счётчик столбцов
              rows_counter := 0;
              odd_even := false; // снова инициализировать полосатость
              prev_group := ord(upcase(sorted_rubriks.Strings[i][1])); //
сменить текущую группу
              if (page_counter>220) then begin
                 page_counter := 0;
                 Rubriks_Names.add(format('<h2 STYLE="{PAGE-BREAK-BEFORE:
always}"><a name="abc_%d">%s</a></h2>',[prev_group,
upcase(sorted_rubriks.Strings[i][1])]));
              end else Rubriks_Names.add(format('<h2><a
name="abc_%d">%s</a></h2>',[prev_group,
upcase(sorted_rubriks.Strings[i][1])]));
              inc(page_counter,30); // высота заголовка - 27 пунктов
              Rubriks_Names.Add('<table class="toc" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5><tr class="odd">'); // открыть новую
таблицу
           end;

           if z = 3 then begin // запустить новую строку
              inc(page_counter,10);  // высота строки - 9 пунктов
              inc(rows_counter);
              z := 0;
              if odd_even
                 then Rubriks_Names.Add('</tr><tr class="odd">')
                 else Rubriks_Names.Add('</tr><tr class="even">');
              odd_even:=not odd_even;
           end;
           if (page_counter>220) and (rows_counter>1) then begin // высота
страницы - 210 пунктов
              Rubriks_Names.Add('</tr></table>');
              z := 0;
              rows_counter := 0;
              page_counter := 0;
              odd_even:=false;
              //Rubriks_Names.add('<div STYLE="{PAGE-BREAK-BEFORE:
always}"></div>');
```

```
              Rubriks_Names.add(format('<h2 STYLE="{PAGE-BREAK-BEFORE:
always}">%s</h2>',[upcase(sorted_rubriks.Strings[i][1])]));
              Rubriks_Names.Add('<table class="toc" style="font-size=-2"
width=100% cellspacing=0 cellpadding=5><tr class="odd">');
              inc(page_counter,30); // высота заголовка - 27 пунктов
           end;
           inc(z);

           Rubriks_Names.Add(format('<td class="toc" width="33%%"><a
href="#Rubrika_%1:d">%0:s</a></td>',[sorted_rubriks.Strings[i],
TRubr_Info(sorted_rubriks.Objects[i]).info.number]));
           local_progress.StepIt;
        end;
        for x:=z to 2 do begin // закончить последнюю строку в таблице
           Rubriks_Names.Add('<td class="toc"> </td>'); // дописать пустые
ячейки до трёх
        end;
        Rubriks_Names.Add('</tr></table>');
        Rubriks_TOC.add('</td></tr></table>');

        // построить список нижнего уровня
        local_progress.Min:=0;
        local_progress.max:=length(Rubriks)-1;
        local_progress.Position:=local_progress.Min;
        local_progress.Step:=1;
        statusbar1.simpletext:='Построение поавторского списка статей...';
        Rubriks_Articles.add('<div STYLE="{PAGE-BREAK-BEFORE:
always}"></div>');
        prev_group := 0;
        for i := 1 to length(Rubriks)-1 do begin
           local_progress.StepIt;
           if (Rubriks[i].group_number<>prev_group) then begin // сменилась
группа рубрик
              prev_group := Rubriks[i].group_number; // сменить текущую группу
              Rubriks_Articles.add(format('<h1 style="{page-break-before:
always}">%s</h1>',[Rubriks_Groups[prev_group].group]));
           end;
           if rubriks[i].common=1 then begin // регулярная рубрика
              // style="{page-break-before: always}"
              stroka := format('<h2><a
name="Rubrika_%0:d">%1:s</a></h2>',[i, Rubriks[i].name]);;
              Rubriks_Articles.add(stroka);
              show_table_rubrik(i);
           end else begin // обычная рубрика
              stroka := format('<h2><a
name="Rubrika_%0:d">%1:s</a></h2>',[i, Rubriks[i].name]);;
              Rubriks_Articles.add(stroka);
              Rubriks_Articles.add('<ul>');
              for x:=Rubriks[i].index + Rubriks[i].Count - 1 downto
Rubriks[i].index do begin // пройтись по всем записям индекса автора
                 with articles[Rubr_index[x].index] do begin
```

```
                    if (year = '1992') and  (volume = '05') then begin
                       year_vol_str := '1992_05_06';
                       str_vol := '05-06';
                    end else begin
                       year_vol_str := year + '_' + volume;
                       str_vol := volume;
                    end;

                    stroka:=format('<li><a href="%0:s/%1:s.djvu#%2:s">%0:s
№%3:s – ',[year, year_vol_str, page, str_vol]);
                    if (rubrika <> '') then stroka := stroka + '[<font
color=blue>' + rubrika + '</font>] ';
                    if (author <> '') then stroka := stroka + '(<font
color=green>' + author + '</font>) – ';
                    if (title <> '') then stroka := stroka + title + ' ';
                    stroka := stroka + '</a>';
                    Rubriks_Articles.Add(stroka);
                 end;
              end;
              Rubriks_Articles.add('</ul>');
           end;
        end;
        Rubriks_Articles.add('</ul>'); // конечный /UL
        statusbar1.simpletext:='Готово';

        Rubriks_TOC.SaveToStream(file_All_Rubriks_Index);
        Rubriks_Names.SaveToStream(file_All_Rubriks_Index);
        Rubriks_Articles.SaveToStream(file_All_Rubriks_Index);
      finally
        IRange := nil;
        ISheet := nil;
        ISheet_Rubr_index := nil;
        ISheet_Rubriks := nil;
        setlength(articles,0);
        setlength(Rubr_index,0);
        setlength(Rubriks,0);
        file_All_Rubriks_Index.Free;
        Rubriks_TOC.Free; // освободить метсто для создания главной таблицы (верхний
уровень)
        Rubriks_Names.Free; // освободить мнсто для создания списка фамилий
(средний уровень)
        Rubriks_Articles.Free; // освободить место для создания блоков ссылок на
статьи (нижний уровень)
        {
    // открыть созданный Html во встроенном редакторе
        Flags := 0;
        html_preview.Navigate(WideString(prog_path+'all_content.htm'),
Flags, Flags, Flags, Flags);
        }
      end;
```

```
    except
      raise Exception.Create('Не могу прочитать данные в массив!');
    end;
    end;
procedure make_bookmarks;
    var Bookmarks_Content: TStringList;
    x, y: integer;
    year_vol_str, str_vol: string;
Const min_vol = 1;
      max_vol = 12;
begin
    Bookmarks_Content := TStringList.Create;
    Bookmarks_Content.Add('<ul>');
    Bookmarks_Content.Add('<li><a href="all_content.djvu">Сквозное
оглавление</a></li>');
    Bookmarks_Content.Add('<li><a href="all_authors.djvu">Авторский
указатель</a></li>');
    Bookmarks_Content.Add('<li><a
href="all_rubriks.djvu">Рубрикатор</a></li>');
    Bookmarks_Content.Add('<li><a href="#">Архив</a></li>');
    Bookmarks_Content.Add('<ul>');
    for x := max_year downto min_year do begin
       Bookmarks_Content.Add(format('<li><a href="#">%d год</a></li>',[x]));
       Bookmarks_Content.Add('<ul>');
       for y := min_vol to max_vol do begin
          if y<10
             then begin
                year_vol_str := format('%d_0%d',[x,y]);
                str_vol := format('0%d',[y])
             end else begin
                year_vol_str := format('%d_%d',[x,y]);
                str_vol := format('%d',[y])
             end;
          if (year_vol_str = '1992_05') then begin // сдвоенный номер 1992 05-06
             year_vol_str := '1992_05_06';
             str_vol := '05-06';
          end;
          if (x = 1992) and (y = 6) then continue; // пропустить номер 1992 6
          Bookmarks_Content.Add(format('<li><a href="%0:d/%1:s.djvu">%0:d
№%2:s</a></li>',[x, year_vol_str, str_vol]));
       end;
       Bookmarks_Content.Add('</ul>');
    end;
    Bookmarks_Content.Add('</ul>');
    Bookmarks_Content.Add('<li><a href="search.exe">Сквозной
поиск</a></li>');
    Bookmarks_Content.Add('</ul>');
    Bookmarks_Content.SaveToFile('all_bookmarks.htm');
end;
begin
  global_progress.Min:=0;
```

```
  global_progress.max:=6;
  global_progress.Position:=global_progress.Min;
  global_progress.Step:=1;
  statusbar1.simpletext:='Создание BAT файлов...';
  CreateBatches; // создать BAT файлы для конвертирования PDF в DJVU
  Load_complex_rubriks; // загрузить список сборных рубрик
  Load_rubriks_groups; // загрузить список групп рубрик
  global_progress.StepIt;

  if chk_make_bookmarks.Checked then begin
     statusbar1.simpletext:='Создание закладок для DJVU...';
     Local_Content_Create(self);
  end;
  global_progress.StepIt;

  if chk_make_content.Checked then begin
     statusbar1.simpletext:='Создание глобального оглавления по номерам...';
     per_volumes; // создать глобальное оглавление по номерам (all_content.htm)
  end;
  global_progress.StepIt;

  if chk_make_authors.Checked then begin
     statusbar1.simpletext:='Создание авторского указателя...';
     per_authors; // создать глобальный авторский указатель (all_authors.htm)
  end;
  global_progress.StepIt;

  if chk_make_rubriks.Checked then begin
     statusbar1.simpletext:='Создание рубрикатора...';
     per_rubrics; // создать глобальный рубрикатор (all_rubrics.htm)
  end;
  global_progress.StepIt;

  statusbar1.simpletext:='Создание закладок для файлов all_?????.htm...';
  make_bookmarks; // создать закладки для all_    .htm
  global_progress.StepIt;
  statusbar1.simpletext:='Готово';
  local_progress.Position := local_progress.Max;
  global_progress.Position := global_progress.Max;
end;

procedure TForm1.btn_simple_volumeClick(Sender: TObject);
begin
   btn_simple_volume.OnClick := Local_Content_Create;
   CreateBook;
   btn_simple_volume.Caption := 'Построить оглавление заново';
   form1.Refresh;
   chk_debug_mode.Checked := true;
   Initiate_Data(Sender);
   grp_simple_volume.Caption := edt_volumes.Items[0];
   Local_Content_Create(Sender);
```

```
end;

procedure TForm1.btn_Collect_DataClick(Sender: TObject);
type temp_content_element = record
        rubrika : string;
        author : string;
        title : string;
        page : string;
        link : string;
        volume : string;
        year : string;
        responsible : string;
     end;
var FullFileName: string;
    x, n, position:integer;
    Values: OLEVariant;
    IRange: Excel8TLB.Range;
    ISheet: Excel8TLB._Worksheet;
    Content: array of temp_content_element;
    x_volume : string;
    x_year : string;
    x_responsible : string;
    use_interpolation: boolean;
procedure check_content(FullFileName:string; x:integer; var
rubrika,author,title,page,link,volume,year,responsible:string);
   var test:integer;
   begin
      if use_interpolation then begin
         if trim(rubrika+author+title+page+link)='' then exit; // пропустить
пустые вызовы
      end else begin
         if trim(rubrika+author+title+page+link+volume+year+responsible)=''
then exit; // пропустить пустые вызовы
      end;
      if (trim(page)='') or (trim(volume)='') or (trim(year)='') or
(trim(responsible)='')
         then ShowMessageInLog(year, volume, page, format('Отсутствует одно или
несколько обязательных полей (страница, год, номер, ответственный), см. файл %s, строку
%d',[FullFileName,x]));
      if (trim(rubrika + author + title) = '')
         then ShowMessageInLog(year, volume, page, format('Не указано ни
рубрики, ни автора, ни названия статьи, см. файл %s, строку %d',[FullFileName,x]));
      try
         test:=strtoint(year);
      except
         ShowMessageInLog(year, volume, page, format('Ошибка в формате года
журнала %s, см. файл %s, строку %d',[year,FullFileName,x]));
         test:=min_year; // чтобы не сработало следующее сообщение об ошибке
      end;
      if (test>max_year) or (test<min_year)
         then ShowMessageInLog(year, volume, page, format('Год журнала %d,
```

```
находится вне диапазона %d-%d, см. файл %s, строку
%d',[test,min_year,max_year,FullFileName,x]));
      try
         test:=strtoint(volume);
      except
         ShowMessageInLog(year, volume, page, format('Ошибка в формате номера
журнала %s, см. файл %s, строку %d',[volume,FullFileName,x]));
         test:=1; // чтобы не сработало следующее сообщение об ошибке
      end;
      if (test>12) or (test<1)
         then ShowMessageInLog(year, volume, page, format('Номер журнала %d,
находится вне диапазона 01-12, см. файл %s, строку %d',[test,FullFileName,x]));
      try
         test:=strtoint(page);
      except
         ShowMessageInLog(year, volume, page, format('Ошибка в формате номера
страницы %s, см. файл %s, строку %d',[page,FullFileName,x]));
         test:=1; // чтобы не сработало следующее сообщение об ошибке
      end;
      if (test<1) or (test>200)
         then ShowMessageInLog(year, volume, page, format('Номер страницы %d,
находится вне диапазона 1-200, см. файл %s, строку %d',[test,FullFileName,x]));
      if rubrika = 'Рефераты. Садоводу — на заметку'
         then rubrika := 'Рефераты | Садоводу — на заметку';
      if rubrika = 'Вкладка'
         then rubrika := 'Вкладки';
      if rubrika = 'Обложки'
         then rubrika := 'Обложка';
      if rubrika = 'Рефераты. Садоводу — на заметку'
         then rubrika := 'Рефераты | Садоводу — на заметку';
      if title = 'Маленькие хитрости' then begin
         title := '';
         if rubrika = ''
            then rubrika := 'Маленькие хитрости'
            else rubrika := rubrika + ' | Маленькие хитрости';
         end;
   end;
function cure_volume(stroka:string):string;
   begin
      if length(stroka)=1
        then result := '0' + stroka
        else result := stroka;
   end;
function cure_page(stroka:string):string;
   begin
      try
         stroka:=inttostr(strtoint(stroka));
      except
      end;
      result:=stroka;
   end;
```

```
function cure(stroka:string):string;
   var temp: string;
   begin
      temp:=StringReplace(stroka, '  ', ' ', [rfReplaceAll]);
      temp:=StringReplace(temp, '...', '…', [rfReplaceAll]);
      temp:=StringReplace(temp, '|', ' | ', [rfReplaceAll]);
      temp:=StringReplace(temp, ' ,', ',', [rfReplaceAll]);
      temp:=StringReplace(temp, ' .', '.', [rfReplaceAll]);
      temp:=StringReplace(temp, ',', ', ', [rfReplaceAll]);
      temp:=StringReplace(temp, '.', '. ', [rfReplaceAll]);
      temp:=StringReplace(temp, '. -', '.-', [rfReplaceAll]);
      temp:=StringReplace(temp, '——', '—', [rfReplaceAll]);
      temp:=StringReplace(temp, '  ', ' ', [rfReplaceAll]);
      temp:=StringReplace(temp, ', ,', ',', [rfReplaceAll]);
      temp:=StringReplace(temp, '. .', '.', [rfReplaceAll]);
      result:=trim(temp);
   end;
begin
  OpenDialog1.InitialDir:=prog_path;
  OpenDialog1.Options:=OpenDialog1.Options + [ofAllowMultiSelect];
  if not OpenDialog1.Execute then exit; // свалить, коли не открыли ничего

  if Application.MessageBox(
    'Использовать интерполяцию года-номера-ответственного по первой статье в файлах?',
    'Выбор способа обработки  XLS  файлов',
    MB_YESNO) = IDYES then use_interpolation:=true else
use_interpolation:=false;
  statusbar1.simpletext:='Объединение файлов...';
  local_progress.Min:=0;
  local_progress.max:=OpenDialog1.Files.Count;
  local_progress.Position:=local_progress.Min;
  local_progress.Step:=1;
  global_progress.Min:=0;
  global_progress.max:=3;
  global_progress.Position:=global_progress.Min;
  global_progress.Step:=1;
  for n:=0 to OpenDialog1.Files.Count-1 do begin
     FullFileName := OpenDialog1.Files[n];
     local_progress.StepIt;
     statusbar1.simpletext:='Чтение файла
'+extractfilename(fullfilename)+'...';
     if Assigned(IXLSApp) and (not Assigned(IWorkbook2))then
       try
         try
           FIWorkbook2 := IXLSApp.Workbooks.Open(FullFileName,
                EmptyParam, EmptyParam, EmptyParam, EmptyParam, EmptyParam,
EmptyParam,
                EmptyParam, EmptyParam, EmptyParam, EmptyParam, EmptyParam,
false, 0);
           ISheet := IWorkbook2.Worksheets.Item['Содержание'] as
Excel8TLB._Worksheet;
```

```
            IRange := ISheet.UsedRange[0];  // прочесть весь лист
            Values := IRange.Value;
            position := length(Content);
            setlength(Content, position + IRange.Rows.Count-1);

            // взять значение номера, года и ответственного их первой строки
            if use_interpolation then begin
               x_volume := Values[2,6];
               x_volume := cure(x_volume);
               x_volume := cure_volume(x_volume);
               x_year := Values[2,7];
               x_year := cure(x_year);
               x_responsible := Values[2,8];
               x_responsible := cure(x_responsible);
            end;
            for x:=2 to IRange.Rows.Count do begin
               with Content[position + x - 2] do begin
                  rubrika := Values[x,1];
                  rubrika := cure(rubrika);
                  author := Values[x,2];
                  author := cure(author);
                  title := Values[x,3];
                  title := cure(title);
                  page := Values[x,4];
                  page := cure(page);
                  page := cure_page(page);
                  link := Values[x,5];
                  link := cure(link);
                  if use_interpolation then begin
                     volume := x_volume;
                     year := x_year;
                     responsible := x_responsible;
                  end else begin
                     volume := Values[x,6];
                     volume := cure(volume);
                     volume := cure_volume(volume);
                     year := Values[x,7];
                     year := cure(year);
                     responsible := Values[x,8];
                     responsible := cure(responsible);
                  end;
check_content(FullFileName,x,rubrika,author,title,page,link,volume,year,resp
onsible);
               end;
            end;
          finally
            IWorkBook2.Close(EmptyParam,EmptyParam,EmptyParam,0);
            FIWorkbook2 := nil;
            ISheet := nil;
          end;
        except
```

```
         raise Exception.Create('Не могу открыть книгу!');
       end;
     end;
  try
     statusbar1.simpletext:='Выгрузка в файл "content.xls"...';
     ISheet := IWorkbook.Worksheets.Item['Содержание'] as
Excel8TLB._Worksheet;
     IDispatch(IRange) := ISheet.UsedRange[0].Rows.Item[1, EmptyParam]; //
первая строка области
     Values := IRange.Value; // сохранить первую строку
     ISheet.Cells.ClearContents; // очистить лист "Содержание"
     IRange.Value := Values; // записать назад первую строку
     //IRange := nil;
     //VarClear(Values);

     // выгрузить инфу на лист ISheet_auth_index
     global_progress.StepIt;
     Values := VarArrayCreate([1, length(Content)+1, 1, 8], varVariant);
     local_progress.Min:=0;
     local_progress.max:=length(Content);
     local_progress.Position:=local_progress.Min;
     local_progress.Step:=1;
     for x:=0 to length(Content)-1 do begin
        local_progress.StepIt;
        with Content[x] do begin
           if page = '' then continue;
           Values[x+1,1] := rubrika;
           Values[x+1,2] := author;
           Values[x+1,3] := title;
           Values[x+1,4] := page;
           Values[x+1,5] := link;
           Values[x+1,6] := volume;
           Values[x+1,7] := year;
           Values[x+1,8] := responsible;
        end;
     end;
     IRange:= ISheet.Range['A2','H'+inttostr(length(Content)+1)];
     IRange.Value := Values;
     VarClear(Values);
  finally
     setlength(Content,0);
     IRange := nil;
     ISheet := nil;
     ShowExcel;
     Application.BringToFront;
     sort_articles; // отсортировать статьи
  end;
  global_progress.stepit;
  statusbar1.simpletext:='Сохранение "content.xls"...';
  IWorkbook.Save(0);
```

```
  local_progress.Position := local_progress.Max;
  global_progress.Position := global_progress.Max;
  statusbar1.simpletext:='Ok';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
errorlog.Items.Clear;
end;

procedure TForm1.Button2Click(Sender: TObject);
var otchet : TStringList;
    x: integer;
begin
   otchet := TStringList.Create;
   SaveDialog1.InitialDir := prog_path;
   if SaveDialog1.Execute then begin
      for x:=0 to ErrorLog.Items.Count-1 do begin
         with ErrorLog.Items.Item[x] do
            otchet.Add(format('%s: %s, %s, %s',[Caption,
SubItems.Strings[0], SubItems.Strings[1], SubItems.Strings[2]]));
      end;
      otchet.SaveToFile(SaveDialog1.FileName);
   end;
   otchet.Free;
end;

end.

// Особая позиция! Только для служебного "пользования".
//
// Я люблю Microsoft. Всей своей мелкой душонкой, всем своим бренным телом и еще, чем
только могу.
// Я люблю эту фирму за то, что периодически веселит меня изменениями в логике методов
интерфейсов
// ее приложений. За то, что мне не скучно жить от все увеличивающихся сервис-паков и
сервис-релизов.
// За редактор VBA я тоже ее люблю, а особенно за ToolTips в нем и F5/F8/F9 в
отладчике. И за язык
// этот самый я ее люблю. И за "правильные" коллекции в ExcelTLB с прекрасным
механизмом поиска в них.
// И за обработку исключений, и за их периодическую генерацию. А еще люблю за то, что один
и тот
// же код может работать, а может и не работать. Все ж зависит от влажности воздуха и
атмосферного
// давления, от фазы луны и расположения звезд. Именно от этого перестает выгружаться
Outlook с
// парами десятков потоков. Он, ведь, практически не использует ресурсов? От этого также
валится
// Word, забыв сохранить последний мой файл. А Excel от этого только устойчивей. Он
под натиском
```

```
// обработчиков событий просто вешает своих клиентов, потом отрубает их и говорит, что его нельзя
// закрывать, потому что просто нельзя!
// Так я люблю Microsoft, такой сильною любовью, что если б была она женщиной...
// Но видно не выжила б эта женщина в наших широтах с нашим климатом и под НАШИМИ звездами.
//
// А еще мне нравится справка по VBA. Особенно, в Excel 2000. И на кого ее рассчитывали?
// Или они думали, что я буду набирать что-то, типа "Покажи мне, пожалуйста, как использовать
// свойство SubTotals у объекта PivotField?"
//
//                                                  Злой.
```

From:
https://kibi.ru/ - **Киби.ru**

Permanent link:
https://kibi.ru/science_and_life/isxodniki

Last update: **2017/03/13 18:20**