```
Xk = Val(InputBox("Введите значение Xk", "Xk", "1"))
     sht.Range("d3").Value = Xk
Dx = Val(InputBox("Введите значение Dx", "Dx", "0.1"))
     sht.Range("f3").Value = Dx
a = Val(InputBox("Введите значение a", "a", "1"))
    sht.Range("b2").Value = a
b = Val(InputBox("Введите значение b", "b", "1"))
   sht.Range("d2").Value = b
```

- **ВНИМАНИЕ.** Следует обратить внимание на то, что параметры, значения которых представлены в виде чисел, имеющих целую и дробную части. При вводе числа в качестве разделителя между целой и дробной частями числа необходимо использовать точку.

  После фиксации введённого значения в ячейке таблицы происходит автоматическое преобразование разделительной точки в запятую.
- После ввода исходных данных следует проверить их совместимость (защита от дурака). Для этого должны быть выполнены следующие условия:
  - значение ***Dx*** не может быть равным 0;
  - если значение ***Dx > 0***, то ***Xk*** должно быть не меньше, чем ***X0***,
  - а если ***Dx < 0***, то ***Xk*** должно быть не больше, чем ***X0***.

Эта проверка реализуется при записи условного оператора

```
' Проверка совместности введённых данных
If (Dx=0) Or (X0<Xk AND Dx<0) Or (X0>Xk AND Dx>0) Then
MsgBox "Введённые значения X0, Xk и Dx несовместны"
Exit Sub ' выход из процедуры
End If
```

Формат условного оператора If...Then...Else

Синтаксис (Syntax)

If condition **Then** [statements] [**Else** elsestatements]

Или блочная синтаксическая форма оператора:

If *condition* **Then**
[*statements*]
[**ElseIf** *condition-n* **Then**
[*elseifstatements*] . . .
[**Else**
[*elsestatements*]]
End If

Здесь *condition (условие)* – обязательный параметр, определяемый выражением одного из двух типов:

1) Числовым (*numeric)* или ***строковым*** *(string)*. В результате вычисления значения этого параметра определяется либо как ***True*** (*Истина*), либо как ***False*** (*Ложно*). Значение условия ***Null (нуль)*** считается равным ***False*** (*Ложно*).

> ***Числовым выражение*** *(numeric expression)*, может быть любое выражение, принимающее в результате вычислений числовое значение. Компоненты, входящие в выражение, могут включать в себя любое сочетание ключевых слов, переменных, констант и операторов, имеющих числовое значение.
>
> ***Строковым выражением*** (*string expression*) является любое выражение, значением которого, получаемым в результате вычисления, является последовательность символов. Элементами строкового выражения могут быть функции, возвращающие строки, строковые литералы, строковые константы, строковые переменные, а также строки типа **Variant** или функции, возвращающие строку **Variant** (**VarType 8**).

2) Выражением вида **TypeOf** *objectname* **Is** *objecttype*.

Здесь *objectname* – это ссылка на любой объект, а *objecttype* любой допустимый тип объекта. Выражение считается истинным (**True),** если имя объекта имеет тип, соответствующий типу объекта, определяемому параметром *objecttype.*

- Определив все, необходимые для решения задачи значения параметров, организуем цикл вычислений значений функции.

В языке VBA имеется несколько видов операторов цикла. Это операторы

Do … Loop
While...Wend
For Each...Next
For...Next

Циклы **Do...Loop** имеют два варианта синтаксической записи. В зависимости от выбранного варианта в записи цикла **Do...Loop** блок операторов либо повторяется до тех пор, пока записанное условие либо имеет значение **Истина** (**True**), либо становится истинным (becomes **True).**

Синтаксис

Do [{**While** | **Until**} *condition*]
[*statements*]
[**Exit Do**]
[*statements*]
Loop

Другой вариант синтаксиса записи этого оператора

Do
[*statements*]
[**Exit Do**]

[*statements*]
Loop [{**While** | **Until**} *condition*]

Для реализации цикла вычисления значений табулируемой функции используем форму цикла ***Do While … Loop*** *(Делай пока.).* Вычисления необходимо выполнить для значений аргумента ***X***, изменяющихся от начального значения ***X0*** с шагом ***Dx*** до тех пор, пока текущее значение ***X*** не станет больше, чем заданное конечное значение ***Xk***.

- Определим начальное значение переменной ***X***, объявленной в программе и используемой как текущее значение аргумента функции её, записав инструкцию:

X = X0 ' Присваиваем аргументу X начальное значение X0

- Определим начальное значение объявленной в программе индексной переменной ***i0*** равным индексу строки, в которой находится первая строка создаваемой таблицы и .

i0 = 5 ' начальное значение индекса, определяющего первую строку таблицыI
I = i0 ' текущее значение индекса - номер заполняемой строки на листе

- Запишем операторы, цикла вычислений:

```
Do While X<=Xk ' Цикл, выполняемый до тех пор, пока X<=Xk
        < операторы  тела цикла>
Loop ' конец цикла заполнения таблицы
```

- В этом цикле выполняются операторы вычисления индекса заполняемой строки таблицы, значения аргумента ***X*** и значения табулируемой функции, а также оператор, изменяющий значение аргумента ***X***.

```
' операторы тела цикла
sht.Cells(i,1) = i – i0 + 1' запись номера строки таблицы
sht.Cells(i,2) = X     ' запись аргумента в строку таблицы
sht.Cells(i,3) = (a*X+b)*X+sin(X)
X=X+Dx : I = I+1
```

- В конце формируемого документа можно вывести сообщение об окончании обработки и указать число строк в сформированной таблице. Запишем оператор

```
MsgBox "В сформированной таблице "+ CStr(I-i0) + "строк",
        ,0,"Решение завершено успешно"
```

Для создания более удобной интерактивной среды на лист книги, используемый для автоматизированного заполнения таблицы, можно поместить на лист книги ещё одну кнопку, обеспечивающую очистку ячеек листа книги от записанной в них информации. Фактически в тексте этой процедуры используются операторы, имеющиеся в процедуре CmdFunTab_Click().

Ниже приводится текст этой процедуры

```
' Процедура, обеспечивающая очистку столбцов таблицы
Private Sub CmdClear_Click()
Dim sht As Worksheet
' Определяем программное имя листа, на котором будут выполняться
' вычисления
 Set sht = Worksheets("Lection6")
' используя метод Clear, очищаем столбцы листа sht
 sht.Range("A:F").Clear
End Sub
```

Создание пользовательских форм средствами VBA

Пользовательские формы ***UserForm*** это один из наиболее интересных инструментов языка ***VB***, используемый при разработке интерактивных приложений.

Пользовательская форма создаётся как оконное приложение, обладающее всеми возможностями таких приложений, разрабатываемых в среде ***ОС*** семейства ***Windows***. В окне формы размещаются элементы управления, обеспечивающие реализацию функций создаваемого приложения.

Для создания пользовательской формы в окне редактора ***VB*** в меню ***Insert*** (*Вставка*) выбираем команду ***UserForm.*** После ***ЩЛК*** по этой строке в окне редактора ***VB*** открывается ещё одно подчинённое окно – окно редактора формы пользователя ***UserForm.***

В поле этого окна находится заготовка окна формы пользователя:

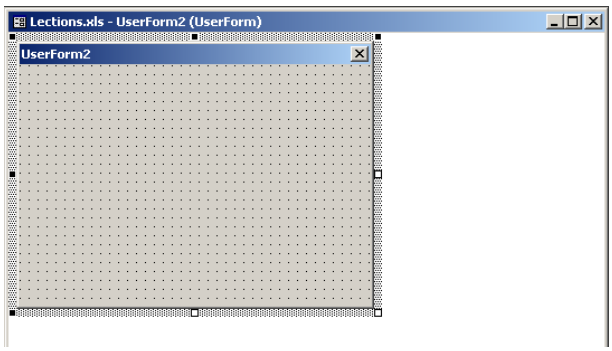


Рис. 1. Окно редактора пользовательских форм

Находящаяся в этом окне заготовка формы содержит два базовых элемента: заголовок окна и рабочее поле окна.

В заголовке записано стандартное название создаваемого окна "***UserFormN***", где ***N*** – число, соответствующее порядковому номеру создаваемой формы. Кроме названия формы в поле заголовка имеется кнопка "***Закрыть***", позволяющая закрыть приложение ***UserFormN***.

На рабочем поле окна ***UserFormN*** может быть нанесена разметочная точечная сетка. (Режим включения этой сетки определяется состоянием

флага ***Show Grid***, находящегося в поле ***Form Grid Settings*** на вкладке "***General***" окна ***Options***.)

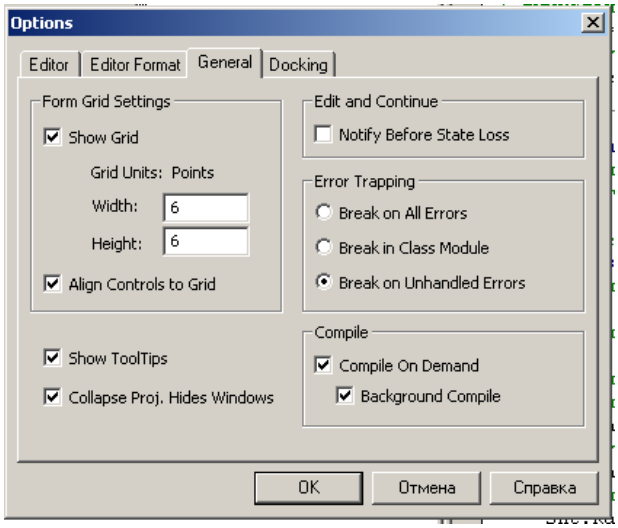


Рис.2. Окно ***Options*** меню "***Tools***" программы ***VB***

В поле окна проекта (на дереве структуры проекта) в папке ***Forms*** (*Формы*) появится объект ***UserFormN.***

Вообще говоря, мы уже создали новое приложение ***UserFormN.*** Для того, чтобы убедиться в этом на ПИ "***Standard***" окна программы ***"Microsoft Visual Basic"*** достаточно нажать кнопку ***"Run Macro***" .

В окне программы Excel откроется окно ***UserFormN,*** которое ничего не может делать. Закроем его, нажав кнопку "***Закрыть***" в заголовке окна.

Объект ***UserForm*** как и любой объект VBA имеет свои *свойства, методы и события*.

Свойства объекта

Свойства объекта – это атрибуты, определяющие его характеристики. Так для пользовательской формы определены такие группы свойств, как ***Appearance*** (*Вид*), ***Behavior*** (*Поведение*), ***Font*** (*Шрифт*), ***Misc*** (*Разное*), ***Picture*** (*Изображение*), ***Position*** (*Положение*), ***Scrolling*** (*Перемещение, прокрутка информации на экране*).

В каждую группу включены один или несколько атрибутов, просмотр и изменение установленных значений которых возможен в окне ***Properties***.

Так, например, в группу ***Misc*** (*Разное*) включен атрибут ***Name*** *(Имя)*, определяющий имя объекта, отображаемое в заголовке окна ***Properties,*** а также на дереве структуры проекта в окне проекта. Так для проверки заменим стандартное имя формы ***UserFormN***, задаваемое при её создании на новое, например ***FrmCalculater***.

В группу ***Appearance*** (*Вид*) включен атрибут ***Caption*** (*Заголовок*), значение которого отображается в строке заголовка окна формы. Заменим стандартный заголовок ***UserFormN*** на название создаваемого приложения – ***Миникалькулятор.***

Методы

Методы – это действия, которые могут быть выполнены над объектом.

Так для объектов ***UserForm*** наиболее часто используются методы ***Show*** *(Показать)*, ***Hide*** *(Скрыть)*, ***Move*** *(Переместить)*, ***Load*** *(Загрузить)*, ***UnLoad*** *(Выгрузить)*.

Рассмотрим пример использования метода ***Show.*** Создадим в рабочей книге лист "***Формы***". На этом листе создадим командную кнопку. В строку ***Caption*** (*Заголовок*) окна ***Properties*** созданной кнопки введём текст ***"Показать форму"***, а в строку ***Name*** *(Имя)* введём новое имя кнопки, например, ***CmdFrmShow***.

Двойной ***ЩЛК*** по кнопке ***CmdFrmShow*** обеспечивает переход в окно редактора ***VB*** и вывод в нём текста заготовки процедуры:

```
Private Sub CmdFrmShow_Click()
    . . .
End Sub
```

Между строками шаблона процедуры запишем инструкцию

```
FrmCalculater.Show
```

Выйдем из режима конструктора и нажмём кнопку ***"Показать форму"*** на листе ***Формы.***

События формы

События – определяют в ***VB*** *действия, распознаваемые объектом*. Для события, опознанного объектом можно запрограммировать отклик. События происходят в результате действия пользователя, в результате выполнения какой-либо программной операции или в результате каких-то действий, выполняемых системой. Например, событием является ***ЩЛК*** по форме, кнопке или какому-либо иному объекту в окне формы.

Отклики на события реализуются в ***VB*** как процедуры. Для некоторых событий заготовки текстов процедур создаются на листе модуля формы автоматически, для других событий процедуры их обработки должны быть полностью записаны программистом.

Создание приложений в окнах UserForm

Приложения, создаваемые в окнах пользовательских форм реализуются путём включения в формы различных элементов управления (***ЭУ***) и записи для этих ***ЭУ*** программных кодов.

В рассматриваемом нами примере создания простого приложения "***Миникалькулятор***" мы используем ***ЭУ*** типа ***Label*** (*Метка*), ***TextBox*** (*Поле ввода*), ***Frame*** (*Рамка*), ***OptionButton*** (*Переключатель*) и ***CommandButton*** (*Кнопка*).

До создания приложения на базе созданной нами формы выполним на ней некоторые предварительные операции.

Во-первых, изменим заголовок окна ***UserFormN***. Для этого в окне ***Properties VBA*** выберем строку ***Caption*** и вместо стоявшего там заголовка ***UserFormN*** введём новое название "***Миникалькулятор***". В строку ***Name*** вместо ***UserFormN*** введём новое имя ***FrmCalculater***.

Следующим шагом нарисуем проект окна приложения, в котором определим набор ***ЭУ***, необходимый для реализации функций приложения.

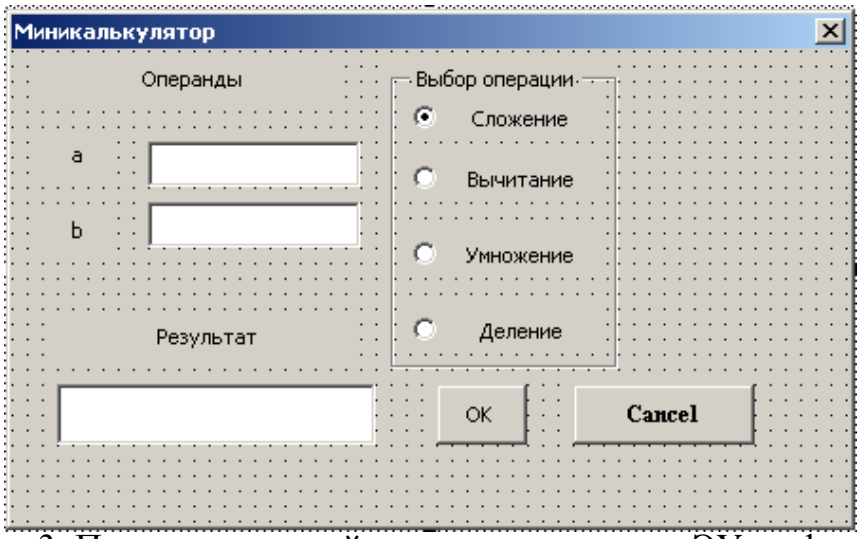


Рис. 3. Предварительный проект размещения ЭУ на форме

При занесении ***ЭУ*** в окно этого проекта, используя окно ***Properties***, определим следующие значения свойств ***ЭУ***.

Тип элемента управления	**Изменяемое свойство**	**Новое значение**	Комментарий
Метка (Label)	Caption	Операнды	Для всех меток устанав-ливаем выравнивание текста по центру
Метка (Label)	Caption	a	
Метка (Label)	Caption	b	
Метка (Label)	TextAlign	**2 -fmTextAlignCenter**	
Поле ввода	Name	TxtA	первый операнд
Поле ввода	Value или Text	0	значение по умолчанию
Поле ввода	Name	TxtB	второй операнд
Поле ввода	Value или Text	1	значение по умолчанию
Поле ввода	Name	TxtResult	результат
Поле ввода	TextAlign	**3-fmTextAlignRight**	для всех полей ввода
Поле ввода TxtResult	Enabled	False	запрещает пользователю ввод данных в поле
Рамка (Frame)	Caption	Выбор операции	
Тип элемента управления	**Изменяемое свойство**	**Новое значение**	Комментарий
Переключатель	Caption	Сложение	
Переключатель	Value	True	активен по умолчанию

Переключатель	Caption	Вычитание	
Переключатель	Caption	Умножение	
Переключатель	Caption	Деление	
Кнопка	Caption	CmdOK	
Кнопка	Caption	Cancel	

ЭУ ***Frame*** (*Рамка*) используется для визуальной группировки таких ЭУ как ***Переключатели*** (*Option Button*) и ***Флажки*** (*Check Box*).

Теперь можно записать программный код для созданной формы.

Начнём с записи программного кода, выполняемого при нажатии кнопки "***Cancel***". В режиме конструктора сделаем ***2ЩЛК*** по кнопке ***Cancel***. В открывающемся при этом окне редактора ***VB*** будет представлена стандартная заготовка программного кода

```
Private Sub CmdCancel_Click()
        . . .
End Sub
```

Между строками заготовки введём код инструкции:

```
Unload Me
```

Объект, определяемый ключевым словом ***Me,*** использованным в записи этого оператора, определяет в коде модуля формы ссылку на саму форму. Оператор ***Unload*** (*Выгрузить*) выгружает форму с экрана и из памяти.

Записав эту процедуру, проверим правильность её выполнения, нажав кнопку ***Cancel*** в окне работающего приложения "Миникалькулятор".

Реализация процедуры, выполняемой при нажатии кнопки ОК

В режиме конструктора сделаем ***2ЩЛК*** по кнопке ***ОК*** и получим заготовку кода процедуры

```
Private Sub CmdOK_Click()


End Sub
```

В первую строку кода модуля формы введём инструкцию

```
Option Explicit
```

Между строками заготовки процедуры ***CmdOK_Click*** введём следующие операторы:

```
' Объявление переменных
Dim a As Double, b As Double, res As Double
' переменной a присваивается значение, введённое в поле ввода TxtA
    a = CDbl(TxtA.Text)
' переменной b присваивается значение, введённое в поле ввода TxtB
    b = CDbl(TxtB.Text)
' сложение
If OptionButton1.Value Then TxtResult.Text = a + b
```

Ниже приводится полный текст процедуры ***CmdOK_Click***

```
Private Sub CmdOK_Click()
' Объявление переменных
   Dim a As Double, b As Double, res As Double
' переменной a присваивается значение, введённое в поле ввода TxtA
   a = CDbl(TxtA.Text)
' переменной b присваивается значение, введённое в поле ввода TxtB
   b = CDbl(TxtB.Text)
' сложение
   If OptionButton1.Value Then TxtResult.Text = a + b
' вычетание
   If OptionButton2.Value Then TxtResult.Text = a - b
' умножение
   If OptionButton3.Value Then TxtResult.Text = a * b
' деление
   If OptionButton4.Value And b <> 0 Then TxtResult.Text = a / b
End Sub
```

Сделаем небольшое дополнение к коду, позволяющее ввести в приложение ещё один уровень защиты "от дурака". Запишем операторы ввода значений параметров a и b в следующем виде:

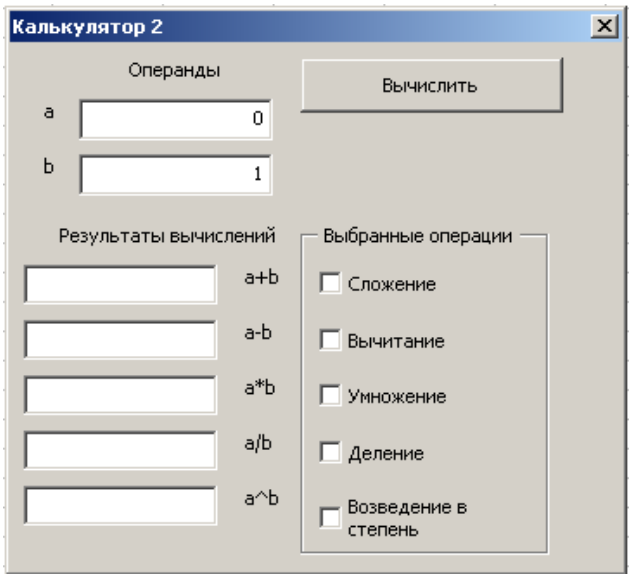
```
If IsNumeric(TxtA.Text) Then a = CDbl(TxtA.Text) _
Else MsgBox ("Введённое значение A не число"): Exit Sub
If IsNumeric(TxtB.Text) Then b = CDbl(TxtB.Text) _
Else MsgBox ("Введённое значение B не число"): Exit Sub
```

Для того, чтобы при смене режима вычислений выполнялась очистка поля вывода результата, включим в модулm формы процедуры, выполняющиеся при ***ЩЛК*** по кнопкам переключателей.

```
Private Sub OptionButton1_Click()
   TxtResult.Text = " "
End Sub
```

Аналогичные процедуры создаются для остальных переключателей.

Вторая версия калькулятора




```
Option Explicit
Private Sub CommandButton1_Click() ' ЩЛК по кнопке "Вычислить"
   Dim a As Double, b As Double
If IsNumeric(TxtA.Text ) Then a = CDbl(TxtA.Text) _
                  Else MsgBox("Введённое А не число!!!"): Exit Sub
If IsNumeric(TxtB.Text ) Then b = CDbl(TxtB. Text) _
                  Else MsgBox("Введённое В не число!!!"): Exit Sub
If CheckBox1 Then TxtResSum = a + b Else TxtResSum = " "
If CheckBox3 Then TextBox2 = a * b Else: TextBox2 = " "
If CheckBox2 Then TextBox1 = a – b Else: TextBox1 = " "
If CheckBox4 Then _
        If b > 0 Or b < 0 Then TextBox3 = a / b _
                          Else TextBox3 = "Деление на 0!!!" _
Else TextBox3 = ""
If CheckBox5 Then TextBox4 = a ^ b Else TextBox4 = ""
End Sub
```

Элемент управления Список

```
Private Sub CmdAdd_Click()
   Static iCount As Long
   Dim NextItem As String
   iCount = iCount + 1
   ' LstItems.AddItem "Item  " & CStr(iCount)
   NextItem = TxtNextItem.Text
   LstItems.AddItem NextItem & " " & CStr(iCount)
End Sub

Private Sub CmdClear_Click()
   LstItems.Clear
End Sub

Private Sub CmdDelete_Click()
      If .ListCount >= 1 Then _
      ' Убеждаемся, что элемент выбран
         If .ListIndex = -1 Then Exit Sub
        .RemoveItem .ListIndex
       Else: Exit Sub
      End If
   End With
End Sub
```