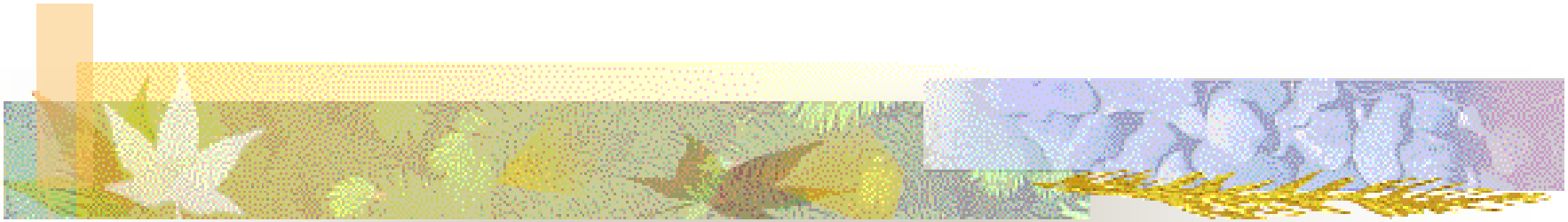
Основы программирования на языке Visual Basic в Excel



Ничего лишнего

Зачем это нужно знать

- Visual Basic — универсальный язык программирования, используемый для написания *макросов* (*минипрограмм*) в компонентах Microsoft Office:

  Excel, Access, Word

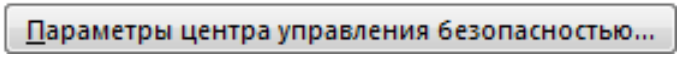
- Вы сможете создавать собственные макросы, дополняющие возможности программы Excel

- Вам станет понятен смысл программ, написанных другими людьми, и вы сможете их модифицировать

  Это не потребует от вас больших усилий и затрат времени — достаточно запомнить около 10 команд!

С чего начать

- Запустите программу Excel
- Чтобы иметь возможность выполнять макросы, необходимо установить соответствующий уровень безопасности Excel (степень защищённости от вирусов).
- Для этого выберите в меню вкладку «Файл», затем пункт «Параметры»
- В списке слева выберите пункт 

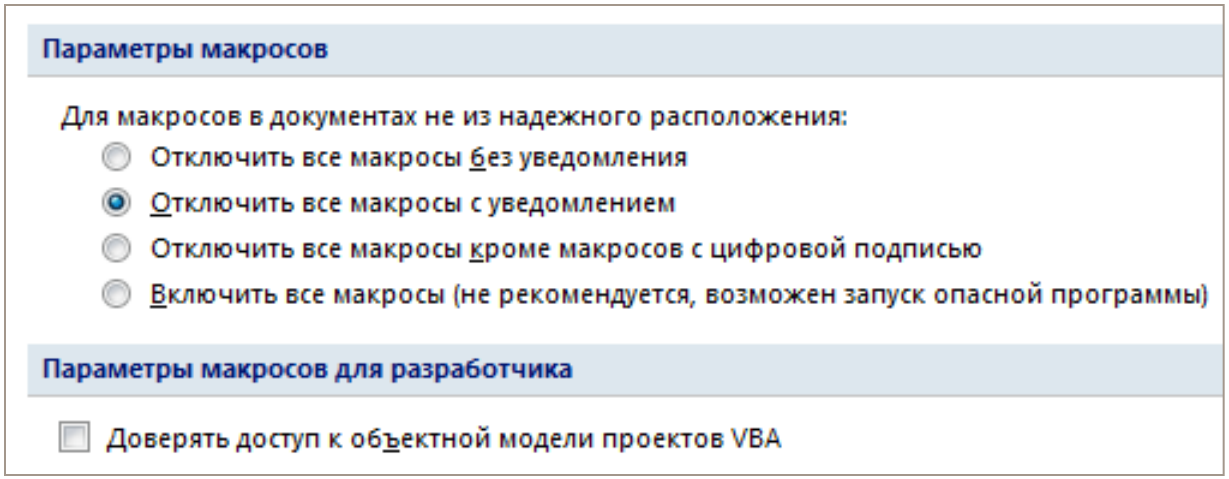
- Нажмите в окне кнопку 

- В списке слева выберите пункт 

Выбор уровня безопасности

- Если установлен такой же уровень безопасности, как в приведённом окне, то дважды нажмите кнопку Отмена

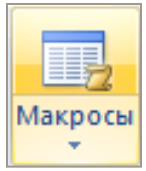
  Если установлен другой уровень, то выберите нужный и дважды нажмите кнопку OK.



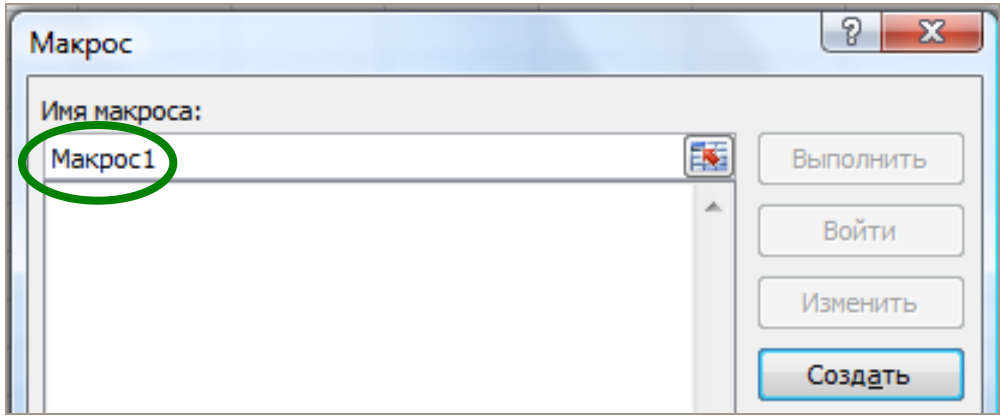
Тогда каждый раз при открытии Excel-файла, содержащего макросы, программа будет спрашивать ваше разрешение на их подключение.

Как создать макрос

- Нажмите одновременно клавиши **Alt+F8** или перейдите в меню на вкладку «**Вид**» и нажмите справа в меню на верхнюю половину кнопки



- В появившемся окне в пустое поле введите любое имя макроса (скажем, **Макрос1**) и нажмите кнопку Создать



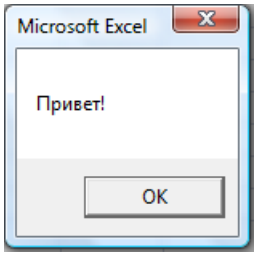
Пример простейшего макроса

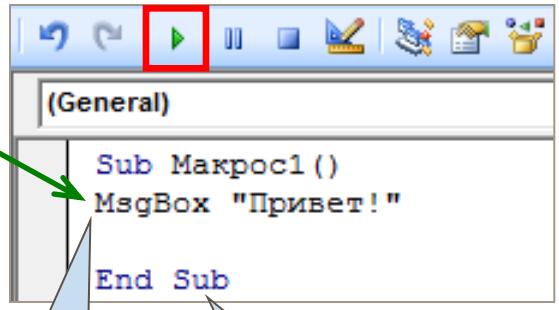
- Введите с клавиатуры команду

```
MsgBox "Привет!"
```

и нажмите клавишу **Enter**

- Для запуска макроса нажмите в меню сверху кнопку 
- В результате на экране на фоне рабочего листа появится окно





Message Box — окно сообщения

Subroutine — подпрограмма

- Нажав кнопку **OK**, вернитесь к тесту макроса, который можно редактировать и дополнять новыми командами

Возвращение из макроса в Excel

- Чтобы вернуться из редактора Microsoft Visual Basic назад на рабочий лист программы Excel, можно

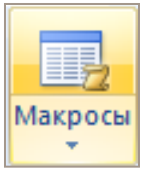
  а) нажать одновременно клавиши **Alt+F11**,

  б) щёлкнуть по кнопке программы Microsoft Excel, находящейся на панели задач внизу экрана

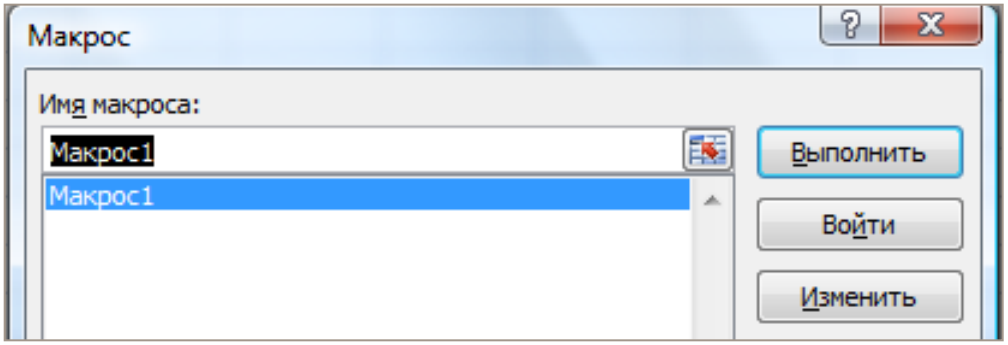
- Аналогично можно снова перейти в редактор Microsoft Visual Basic и продолжить написание макроса

Вызов макроса из Excel

- Нажмите одновременно клавиши **Alt+F8** или перейдите в меню на вкладку «**Вид**» и нажмите справа в меню на верхнюю половину кнопки

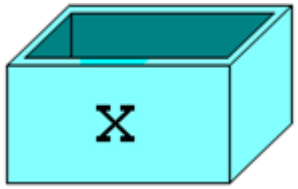


- В появившемся окне следует выбрать нужный макрос и нажать кнопку Выполнить



Представление о переменных

Программа на языке Visual Basic представляет собой набор последовательно выполняемых команд. Команды описывают некоторые действия над переменными.



Можно представлять себе переменные в виде ящиков, в которых хранятся текущие значения переменных.

В этом смысле команда **`X = 5`** помещает в ящик **`X`** константу **`5`**.
Загадочное, на первый взгляд, выражение

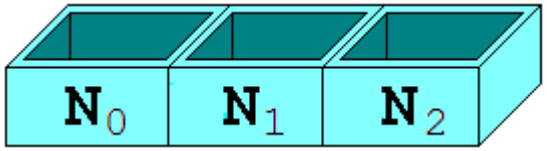
```
N = N + 1
```

означает, всего лишь, что текущее значение в ящике **`N`** увеличивается на **`1`**.

`A = 0` (значение переменной **`A`** полагается равным **`0`**)
`B = C` (значение переменной **`C`** копируется в переменную **`B`**)
`X = Y - Z` (переменной **`X`** присваивается разность значений **`Y`** и **`Z`**)

Массивы

Dimension — размерность



Часто удобно бывает дать общее имя сразу нескольким переменным, а отличие между ними обозначить, приписав им разные индексы. Такой объект называется ***массивом.***

Чтобы использовать массив, его сначала необходимо описать. Для описания массива используется команда

Dim «имя массива»(«количество элементов массива»)

Например, массив **N** длины **7** определяется командой

```
Dim N(7)
```

В свою очередь, команды (двоеточия разделяют команды в строке)

```
N(0) = 0: N(1) = 0: N(2) = 0
```

зануляют первые 3 элемента массива **N**.
(По умолчанию индексация элементов массива начинается с 0).

Команды с условием

Обычно команды программы выполняются одна за другой.
Но можно пропускать некоторые команды, если не выполняются определённые условия. Для этого служит конструкция

If «условие» **Then** «команда»

Здесь «условие» обозначает логическое выражение, содержащее один из знаков сравнения <, >, =, <=, >=, <> (*не равно*).

Например,

```
If K = 0 Then N = N + 1
If I >= J Then X = 0
```

Группы команд с условием

```
If «условие» Then
    «команды1»
Else
    «команды2»
End If
```

Как правило, требуется пропустить не одну команду, а сразу несколько. Для этого используется конструкция, приведённая слева. В ней «команды1» и «команды2» обозначают одну или несколько команд (допустимо также полное отсутствие команд).

Если «условие» верно, то выполняются «команды1», в противном случае выполняются «команды2». Например,

```
If K = 0 Then
    X = A
    N = N + 1
Else
    X = B
End If
```

Отступы делают текст программы более понятным. Чтобы их сделать, надо выделить группу нужных команд и нажать на клавишу **Tab**. Чтобы убрать отступы, следует выделить интересующие команды и нажать клавишу **Tab** и вместе с клавишей **Shift**.

Циклы

For «имя» = «нач» **To** «кон»
«команды»
Next «имя»

Цикл является, пожалуй, важнейшей конструкцией в Visual Basic, в которой «имя» обозначает имя переменной-счётчика, «нач» — начальное значение счётчика, «кон» — конечное значение счётчика.

При этом «команды» выполняются для каждого промежуточного значения счетчика: всего будет («кон» − «нач» + 1) повторов.

```
For I = 0 To 2
   N(I) = 0
Next I
```

Как показывает пример слева, циклы прекрасно взаимодействуют с массивами. В данном случае зануляются первые **3** элемента массива **N**.

Для досрочного выхода из цикла служит команда **Exit For**.

Например,

```
If I = 1 Then Exit For
```

Циклы с условием

Циклы с условием используются тогда, когда заранее не известно, сколько повторов потребуется.

```
Do
    «команды1»
    If «условие» Then Exit Do
    «команды2»
Loop
```

Как только «условие» выполнится, произойдёт выход из цикла.
Если «условие» не выполнятся никогда, то программа «зависнет».
Чтобы её остановить, необходимо будет нажать клавишу **Esc**,
а затем нажать на появившемся окне кнопку **End**
(Данный способ прерывания выполнения программы также может пригодиться, когда макрос выполняется очень долго.)

Обмен информацией с рабочим листом

Получить доступ к содержимому ячейки в **I**-й строке и **J**-ом столбце текущего рабочего листа книги Excel можно с помощью обращения

```
Cells(I,J).Value
```

Например, команда

```
Cells(1,2).Value = 5
```

записывает в 1-ю ячейку 2-го столбца рабочего листа число 5 .

Обратно, чтобы присвоить переменной **X** значение, хранящееся, например, в ячейке **(3,4)** рабочего листа, применяется команда

```
X = Cells(3,4).Value
```

Детали и дополнения

В наше время накопляется огромное количество знаний, достойных изучения. Скоро наши способности будут слишком слабы, а жизнь — слишком коротка, чтобы усвоить хотя бы одну только наиболее полезную часть этих знаний.

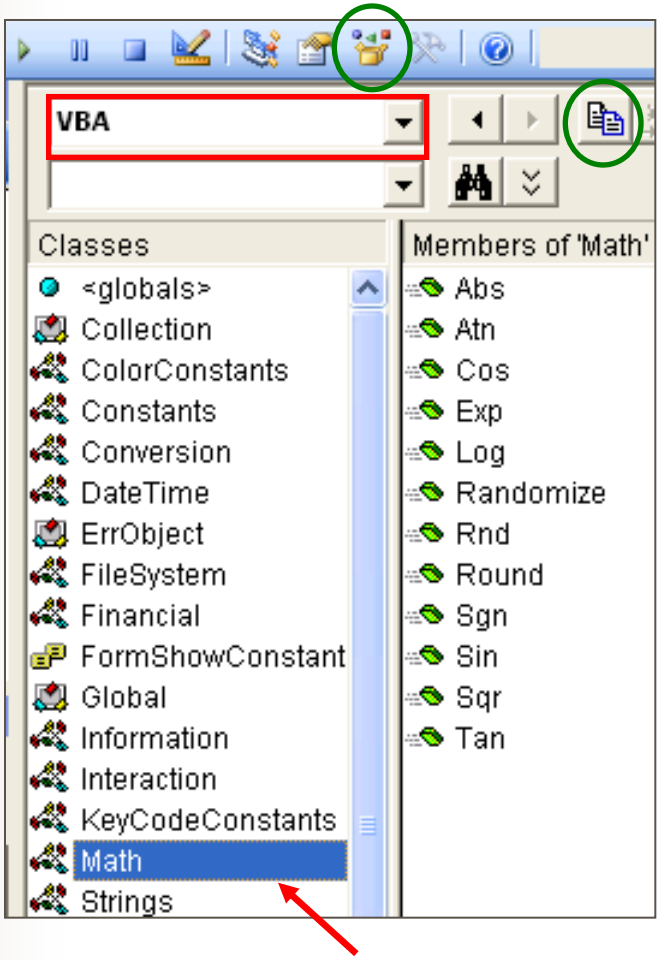
К нашим услугам полное изобилие богатств, но, восприняв их, мы должны снова отбрасывать многое, как бесполезный хлам. Было бы лучше иногда не обременять себя им.

И. Кант

Кто не понимает ничего, кроме химии, тот и её понимает недостаточно.

Г. Лихтенберг

Встроенные функции Visual Basic



- В меню нажмите кнопку
- На появившемся окне в верхнем списке выберите пункт **VBA**
- В списке в середине экрана выберите категорию функции. При этом справа появится список функций из этой категории.

  Получить описание функции можно, выделив её имя и нажав кнопку
- Выберите нужную функцию и копируйте её, нажав кнопку
- Для возвращения в макрос дважды щёлкните по значку перед **Module1**, в левом верхнем окне
- Вставьте скопированную функцию в любое место программы, нажав в меню сверху кнопку

Встроенные функции Excel

При написании макроса на Visual Basic также можно воспользоваться встроенными функциями самой программы Excel.
Они имеют вид

`Application.`«английское название функции»(…)

Например, функция среднее арифметическое (**СРЗНАЧ**) значений, содержащихся в ячейках некоторого диапазона рабочего листа (скажем, диапазона, состоящего из первых 100 ячеек столбца **A**), вызывается с помощью команды

```
Application.Average("A1:A100")
```

Подпрограммы

```
Sub Макрос
    «Подпрогр1» «пар1»
    «Подпрогр2» «пар2»
    …
End Sub
```

```
Sub «Подпрогр1» («пар1»)
    «команды1»
End Sub
```

```
Sub «Подпрогр2» («пар2»)
    «команды2»
End Sub
…
```

Большую программу следует разбить на небольшие логически связные части, называемые подпрограммами (subroutines).

Слева «Подпрог1» — имя первой подпрограммы, «пар1» — список имён переменных и массивов (через запятую), передаваемых в подпрограмму в качестве параметров.

Например, подпрограмма с именем **Step** и параметрами **X,Y,N** вызывается командой

Step X,Y,N

При запуске макроса сначала выполняются «команды1», затем выполняются «команды2» и т. д.

Новые функции

Sub Макрос
«имя» = «функ»(«парам»)
End Sub

Function «функ»(«парам»)
«команды»
«функ» = «знач»
End Function

Вы можете не только использовать функции Visual Basic, но и создать свои.

Слева «функ» — имя функции, «парам» — список имён переменных (через запятые и заключённый в скобки в отличие от вызова подпрограммы), которые передаются в функцию в качестве аргументов.

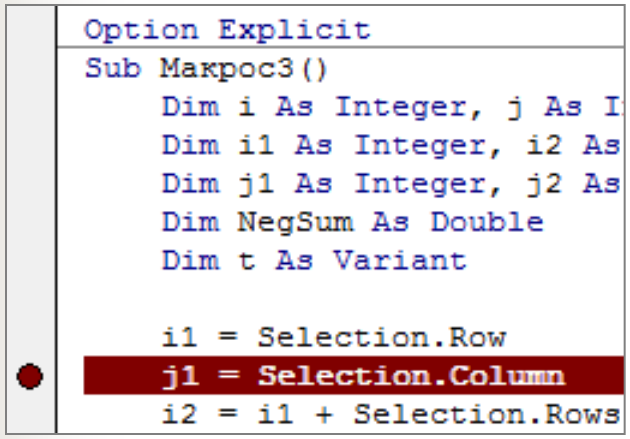
Например, функция с именем **F** и параметрами **X,Y** вызывается как

Z = F(X,Y)

При запуске функции «функ» и выполняются внутренние «команды».
Имени функции «функ» присваивается вычисленное значение «знач».
Наконец, происходит возвращение в вызывающую программу,
и переменная «имя» получает значение «знач».

Отладка программы

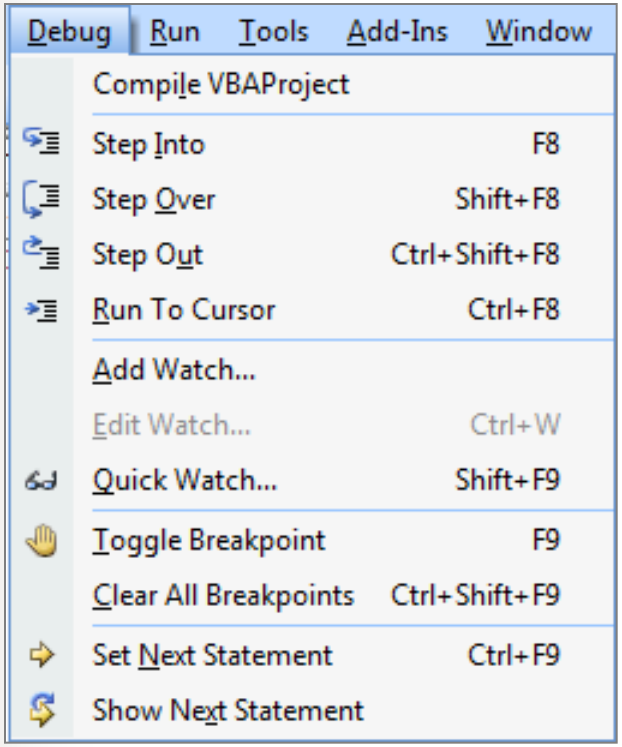
Для обнаружения ошибок в программе применяется отладчик. Чтобы им воспользоваться, необходимо поставить в тексте макроса точку прерывания (breakpoint) [одну или несколько] с помощью клавиши **F9**. Повторное нажатие **F9** убирает имеющуюся точку прерывания.



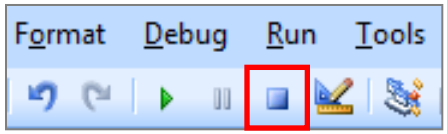
При запуске макроса его команды автоматически выполняются вплоть до точки прерывания, и там происходит остановка. Вы можете узнать текущие значения переменных и элементов массивов, подведя к ним курсор мыши.

Для более сложных объектов необходимо выделить объект в тексте, затем нажать клавиши **Shift+F9**. Тогда текущее значение объекта будет показано в окне **Quick Watch**.

Команды отладчика



- Чтобы остановить работу отладчика, надо нажать в меню на кнопку с синим квадратиком

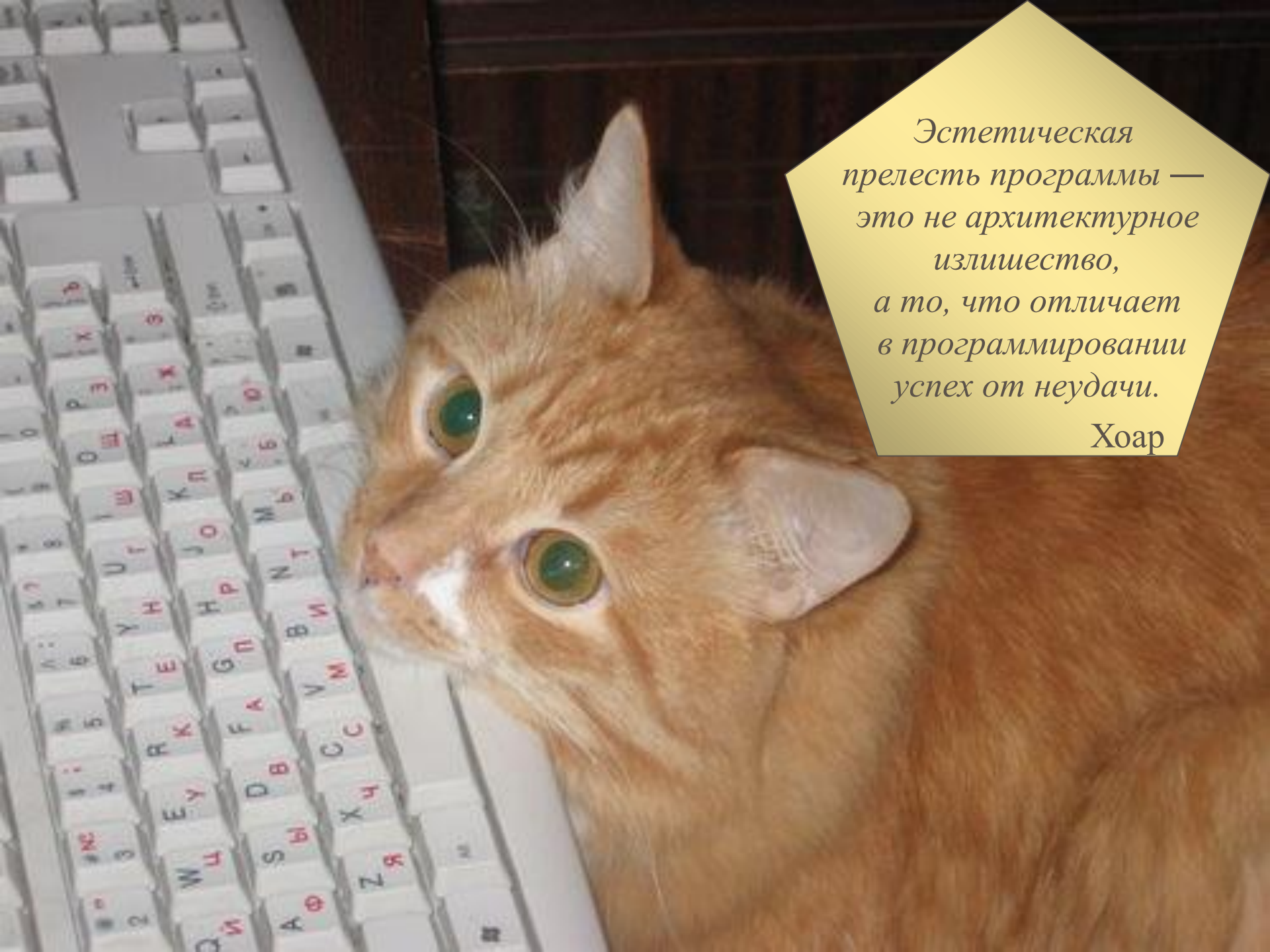


Назначение клавиш

F8 — сделать шаг, т. е. выполнить текущую команду и перейти к следующей. Если на пути встречается подпрограмма, то происходит вход в неё. Чтобы её обойти (выполнить автоматически), надо нажать **Shift+F8**.

F5 — автоматическое выполнение до следующей точки прерывания. [Отметим, что точки прерывания можно добавлять и убирать по ходу отладки.]

Другие возможности отладчика показаны в приведённом слева меню.


Эстетическая прелесть программы — это не архитектурное излишество, а то, что отличает в программировании успех от неудачи.
Хоар