##### Как установить «обратную связь» с пользователем

Для того чтобы сделать макрос более универсальным, т.е. не писать каждый раз новый макрос, если хочется окрасить не одно слово, а два, и не в красный цвет, а в зелёный или синий, надо спросить у пользователя, чего он хочет. Для этого есть простейший способ и соответственно функция, которые мы и будем использовать.

С помощью функции InputBox пользователь может ввести нужное значение в ответ на поставленный вопрос.

Например, спрашиваем: «Сколько выделить слов в начале абзаца?». Пользователь вводит то или иное число. Обязательный параметр этой функции – сообщение-вопрос, на который должен ответить пользователь. Подробнее о функции InputBox смотри в приложении в параграфе «Функции InputBox и MsgBox».

Как же воспользоваться ответом? Полученный ответ будет содержаться в переменной Otwet. Переменная – это область памяти компьютера, в которой хранятся данные, и у неё есть имя. Подробнее о переменных в VBA смотри в приложении в параграфе «Переменные, типы, области видимости». Эту переменную используем в команде:

Otwet = InputBox(“Сколько выделить слов в начале абзаца?”)

Selection.MoveRight Unit:=wdWord, Count:=Otwet, Extend:=wdExtend

К сожалению, из-за несоответствия типов переменных нельзя написать в окне функции InputBox значение wdColorRed и потом подставить переменную Otwet в выражении Selection.Font.Color = Otwet**.** Такой прием вызовет сообщение об ошибке. Для обработки введенного значения, обозначающего цвет, необходим условный оператор. Об этом пойдёт речь чуть позже.

##### Графические объекты.

Если панель инструментов Рисование не видима, включаем её:

Строка меню 🡪 Вид 🡪 Панели инструментов 🡪 Рисование

Для начала надо настроить MsWord для работы с графикой:

Строка меню 🡪 Сервис 🡪 Параметры 🡪 вкладка Общие 🡪 отключить опцию 🞎 Автоматически вставлять полотно при вставке автофигур.

Работу с графикой начнём с записи простого макроса. Запишем последовательность действий:

 (1) Включите запись макроса.

(2) Нарисуйте в документе автофигуру «Сердце».

(3) После того, как фигура была нарисована, остановите запись.

Теперь выполним команды, которые мы записали в макрос, то есть, вставим в документ автофигуру «Сердце» с помощью созданного макроса. Строка меню 🡪 Сервис 🡪 Макрос 🡪 Макросы 🡪 кнопка [Выполнить]

На экране, казалось бы, ничего не происходит. Но если мы передвинем «Сердце» мышкой чуть в сторону, то обнаружим, что под ним имеется еще одно точно такое же «Сердце». Значит, щелкнув по кнопке [Выполнить], мы воспроизвели ранее созданную автофигуру в том же месте.

Откроем для редактирования этот макрос:

Строка меню 🡪 Сервис 🡪 Макрос 🡪 Макросы 🡪 выбрать последний макрос в списке и нажать кнопку Изменить

Вот его программный код:

Sub Макрос#()

ActiveDocument.Shapes.AddShape(msoShapeHeart, \_

 200, 100, 40, 40).Select

End Sub

Замечание. Числа в скобках в команде, записанной у вас, могут отличаться от приведенного здесь образца. Это нормально! Важен сам принцип! Там могут быть дробные значения с десятичной точкой, могут быть целые значения с решеткой после числа. Удалите после чисел дробную часть и решетку, если они там есть. Пусть останется только целое число. Можете изменить цифры и вписать в свой макрос значения такие же, как в указанном примере.

Внимание! Если строчка кода длинная и не помещается в одной строке, то перенос осуществляется последовательностью знаков пробел и подчерк.

Вы можете печатать текст в окне программного кода в одной строке, не перенося его на другую строку.

В макросе имеется всего одна команда – команда, добавляющая автофигуру в документ. Поясним составляющие ее элементы:

ActiveDocument - активный документ (то есть тот, который

 открыт сейчас)

Shapes – обращение к коллекции автофигур

AddShape - добавить автофигуру

msoShapeHeart-автофигура Сердце

200, 100 – координаты верхнего левого угла (белый маркер выделения) автофигуры

40, 40 сама автофигура имеет размер 40 точек в ширину и 40 точек в высоту.

Select – выбрать, выделить (то есть автофигура после создания будет выделена – по ее контуру будут расположены 8 белых маркеров выделения).

Обратите внимание на словосочетание Add – добавить, Shape – фигура. AddShape – это метод (методы – это те действия, которые пользователь может производить с объектами). У метода AddShape есть свои параметры. Действительно, если мы добавляем фигуру, то надо указать что это за фигура (название, тип), ее положение и размеры. Если стереть скобку или любую запятую в области параметров, то внизу под текстом программного кода появится подсказка:

Она поможет понять, какой параметр должен быть введён, и какой тип он имеет.

Воспользуемся введённой в предыдущем параграфе функцией InputBox и дадим возможность пользователю вводить один из параметров фигуры. Пусть это будет параметр, отвечающий за положение фигуры. Тогда макрос будет выглядеть так:

Sub Макрос#()

X= InputBox("Введите расстояние от левого края \_

 страницы документа до автофигуры")

ActiveDocument.Shapes.AddShape(msoShapeSmileyFace, X, \_

30, 50, 50).Select

End Sub

Запускайте макрос и задавайте последовательно значения 100, 150, 200 и 250. Получатся веселые рожицы, расположенные по горизонтали.

Конечно, такой макрос не имеет особого смысла, но с его помощью можно увидеть, как работает переменная, задающая параметр, и подготовиться к изучению следующей темы.

Использование справки. Для того чтобы узнать, как в языке VBA обозначаются разные автофигуры, находясь в редакторе программного кода VBA нажмите клавишу F1. В окне поиска справочной системы напечатайте MsoAutoShapeType и запустите поиск. Затем из двух появившихся значений выберите AutoShapeType Property. (В Microsoft Office 2007 MsoAutoShape Type Enumeration ). В появившемся тексте справки нажмите на синий текст MsoAutoShapeType. Раскроется список названий автофигур и их числовых значений. На рисунке показан фрагмент справки, которая появится в результате.



Задания для самостоятельной работы.

1. Напишите макрос, в котором пользователь задаёт:
	1. ширину фигуры **W**;
	2. высоту фигуры **H**;
	3. координату **Y**, т.е. расстояние до фигуры по вертикали.
2. Напишите макрос, в котором пользователь задаёт два параметра – ширину и высоту фигуры. Для этого используйте два раза функцию InputBox с разными сообщениями, чтобы не перепутать когда какой параметр задавать.
3. Напишите макрос, в котором пользователь может задать тип фигуры числом. Поэксперементируйте с разными автофигурами и их параметрами.

## Глава 2. Операторы языка

##### Циклы. Круг, Circulus, Circle…

Пони девочек катает,

Пони мальчиков катает,

Пони бегает по кругу

И в уме круги считает.

Вспомним две наши задачи из предыдущих параграфов: изменение цвета слова в начале каждого абзаца и вставка фигуры. Мы автоматизировали повторяющиеся операции. Но можно пойти дальше в совершенствовании работы. Для многократного повторения нескольких операторов применяется алгоритмическая конструкция языка программирования – цикл.

Что делать, если мы хотим создать несколько сердец, расположенных вдоль горизонтальной линии? При этом все автофигуры должны располагаться друг от друга на одинаковом расстоянии. Если сердец 10 штук, то добиться точности копированием и перетаскиванием вручную, без использования макросов, будет довольно сложно.

А если использовать макрос… Придётся несколько раз запускать макрос, как в предыдущих примерах, с разными параметрами. Или использовать в одном макросе несколько команд для создания сердец с разными значениями? А что, если будет не десять сердец, а пятьдесят? В подобных случаях работа превращается в ночной кошмар программиста…

Мы можем избежать всех этих проблем, если при написании программного кода будем использовать специальную конструкцию языка программирования цикл. Цикл – это команда или блок команд, которые исполняются определенное количество раз.

Для начала рассмотрим пример, в котором мы будем создавать 4 сердца. Пусть размеры всех сердец будут одинаковы - 40\*40. То есть параметр ширины будет 40 для всех автофигур, точно также как и параметр высоты – тоже 40.

Так как наши автофигуры будут расположены вдоль горизонтальной линии, то параметр, отвечающий за положение по оси Y, меняться не будет. Пусть он у всех фигур будет равен 100.

Нам надо изменить только параметр, отвечающий за положение по оси X. Остальные параметры будут неизменны. Программный код с использованием цикла для рисования 4-х сердец будет выглядеть следующим образом:

Sub Макрос#()

‘вместо # должна быть цифра, которая в имени

‘макроса ранее не использовалась

 For i=100 to 250 step 50

 ActiveDocument.Shapes.AddShape(msoShapeHeart, \_

 i, 100, 40, 40).Select

 Next i

End Sub

Какой же смысл имеет эта конструкция?

Операторы, расположенные между строчками For … и Next , называем телом цикла, это тот программный код, который будет многократно повторяться.

Программный код, который вы видите, означает, что команда создания автофигуры Сердце будет повторяться 4 раза, переменная **i** последовательно принимает значения 100, 150, 200, 250.

For i=100 to 250 step 50

– начало цикла, его заголовок. Эту строку можно интерпретировать так: для переменной i изменяем значение от 100 до 250 с шагом 50

ActiveDocument.Shapes.AddShape(msoShapeHeart, i, 100, 40, 40).Select

Значение переменной **i** подставляется в команду на место параметра Х и команда создания автофигуры Сердце выполняется четыре раза с подставленным цифровым значением.

Next i – это конец цикла.

Запустите этот макрос и посмотрите, как в документе Ms Word появятся 4 сердца.

Вернёмся к макросу. Цикл написан «в лоб», т.е. мы подставили те значения параметра, которые необходимо было использовать в макросе. Преобразуем цикл таким образом, чтобы было легко задавать количество создаваемых автофигур, а значение параметра вычислялось.

Действительно, если написать For i=1 to 4

то значение параметра Х можно вычислить по формуле:

(i - 1) \* 50 + 100

В этой формуле 50 – это расстояние между автофигурами, а 100 – это позиция первой фигуры, которая будет создана, на странице документа.

For i=1 to 4 означает, что будет создано 4 автофигуры.

В готовом виде макрос будет выглядеть следующим образом:

Sub Макрос#()

 For i=1 to 4

 ActiveDocument.Shapes.AddShape(msoShapeHeart, \_

 (i - 1) \* 50 + 100, 200, 40, 40).Select

 Next i

End Sub

Можно ввести ещё одну переменную для хранения промежуточного значения. Тогда макрос запишется следующим образом:

Sub Макрос#()

 For i=1 to 4

 p=(i - 1) \* 50 + 100

 ActiveDocument.Shapes.AddShape(msoShapeHeart, \_

 p, 200, 40, 40).Select

 Next i

End Sub

То, что мы только что сделали при модификации (т.е. изменении) цикла, можно сравнить с решением арифметических примеров в школе. Ответ можно просчитать в уме сразу, а можно расписать решение по отдельным действиям – если пример сложный.

Попробуйте в уме сразу просчитать 950/19-27\*13 … Если получилось – замечательно. Но гораздо проще записать и решить его в несколько действий. Точно также и в нашем макросе с циклом.

Приведенные способы записи цикла не только более изящны с точки зрения программирования, но и необходимы, если меняться будет не один параметр, а несколько.