Уральский федеральный университет
имени первого Президента России Б.Н.Ельцина
Высшая школа экономики и менеджмента

В. А. ПЕТРОВА

ПРОГРАММИРОВАНИЕ И РЕШЕНИЕ СЛОЖНЫХ ЗАДАЧ В EXCEL

Учебное пособие

Министерство образования и науки Российской Федерации
Уральский федеральный университет
имени первого Президента России Б. Н. Ельцина

В. А. Петрова

ПРОГРАММИРОВАНИЕ И РЕШЕНИЕ СЛОЖНЫХ ЗАДАЧ В EXCEL

Рекомендовано методическим советом
Уральского федерального университета
в качестве **учебного пособия** для студентов,
обучающихся по направлениям подготовки
230700 — Прикладная информатика,
080500 — Бизнес-информатика

Екатеринбург
Издательство Уральского университета
2016

УДК 004.91(075.8)
ББК 32.973.26-018.2я73
П30

Рецензенты:
кафедра высшей математики Уральского государственного университета путей сообщения (зав. кафедрой, проф., д-р физ.-мат. наук Г. А. Тимофеева);
канд. физ.-мат. наук Д. Г. Ермаков (Институт математики и механики УрО РАН)

Петрова, В. А.

П30 Программирование и решение сложных задач в Excel : учеб. пособие / В. А. Петрова. — Екатеринбург : Изд-во Урал. ун-та, 2016. — 88 с.

ISBN 978-5-7996-1949-7

Раздел 1 содержит сведения о способах обработки списков средствами программы MS Excel 2013 и задания для выполнения лабораторной работы по данной теме.

Раздел 2 содержит основные сведения о решении оптимизационных задач в среде программы MS Excel 2013 и задания для выполнения лабораторной работы по данной теме.

Раздел 3 содержит сведения о решении задач с использованием финансовых функций MS Excel и задания для выполнения лабораторной работы по данной теме.

Раздел 4 содержит основные сведения о возможностях языка программирования Visual Basic for Application и задания для выполнения лабораторных работ по программированию.

Раздел 5 содержит основные сведения о работе с формами в среде программы MS Excel 2013 и задания для выполнения лабораторной работы по разработке формы.

Пособие предназначено для студентов четвертого курса Высшей школы экономики и менеджмента. Составлено в соответствии с программой курса «Программирование и решение сложных задач в Excel» и может быть использовано для самостоятельного изучения данного курса.

Библиогр.: 7 назв. Табл. 5. Рис. 27.

УДК 004.91(075.8)
ББК 32.973.26-018.2я73

Учебное электронное сетевое издание

Петрова Вера Александровна

ПРОГРАММИРОВАНИЕ И РЕШЕНИЕ СЛОЖНЫХ ЗАДАЧ В EXCEL

Корректор *Е. Е. Афанасьева*
Верстка *О. П. Игнатьевой*

Подписано в печать 25.12.2016. Формат 70×100/16. Гарнитура Century Schoolbook.
Уч.-изд. л. 5,0.

Издательство Уральского университета. Редакционно-издательский отдел ИПЦ УрФУ
620049, Екатеринбург, ул. С. Ковалевской, 5. Тел.: 8(343)375-48-25, 375-46-85, 374-19-41. E-mail: rio@urfu.ru

ISBN 978-5-7996-1949-7

Оглавление

Введение

В работе современных предприятий все большее значение приобретает возможность использования существующей информации и возможность получения качественно новой информации. Способность работников предприятия извлечь нужные данные и умение представить их в виде различных отчетов напрямую связано с результатами, которые могут быть использованы руководством предприятия для текущего анализа деятельности и дальнейшего принятия решений по совершенствованию работы предприятия.

Табличный процессор MS Excel является мощным инструментом для получения рациональных решений в тех случаях, когда требуется обработка больших объемов информации, связанная с поиском, фильтрацией, сортировкой и получением итоговых значений с помощью различных функций.

Мощный математический аппарат встроенных функций (математических, статистических, финансовых и пр.) предоставляет огромные возможности по обработке данных, помещенных в ячейки листов с помощью непосредственного ввода или путем вывода из других офисных приложений и приложений, поддерживающих технологию *COM*.

Встроенный в MS Excel язык программирования *VBA* позволяет существенно улучшить качество создаваемых приложений. Легко выбираемые из списка отдельные макросы позволяют автоматизировать выполнение часто повторяющихся операций, а наличие пользовательской формы с определенным набором элементов управления предоставляет дополнительные возможности, такие как проверка вводимых значений в табличную базу данных, поиск нужных значений, быстрое удаление и обновление записей табличной базы.

Раздел 1.
Обработка табличных баз данных

Программа MS Excel представляет собой не просто удобное средство для выполнения математических и логических операций, а мощный и универсальный инструмент по решению задач, возникающих в сфере экономики и финансов. Создавая отчетную, финансовую и экономическую документацию на рабочем листе MS Excel, важно уметь анализировать информацию и выбирать оптимальное решение на основе имеющихся данных. Для представления экономической информации на листе рабочей книги очень часто используются списки, так как это удобный способ представления данных и возможность использования целого ряда мощных инструментов MS Excel по обработке и анализу данных.

Список или база данных рабочего листа — это упорядоченный набор данных, обладающий следующими свойствами:

— данные располагаются в столбцах;

— каждый столбец имеет однородный тип данных;

— каждый столбец имеет уникальное имя;

— первая строка списка — строка заголовков столбцов списка.

Столбцы списка называют полями, а строки — записями. Размер списка ограничен размерами рабочего листа. К списку применимы следующие операции: сортировка, фильтрация, подведение итогов, построение сводных таблиц.

Для автоматического обнаружения списка на листе *Excel* при выполнении перечисленных операций необходимо отделить список от остальных данных листа, оставив как минимум одну пустую строку над списком и одну пустую строку под списком. Аналогично слева и справа от списка оставляют пустыми как минимум по одному столбцу.

1.1. Сортировка списка

Сортировка позволяет выстраивать данные списка в алфавитном, цифровом и хронологическом порядках. При этом можно задать возрастающий, убывающий и пользовательский порядки сортировки.

Если задан возрастающий порядок сортировки, то данные списка упорядочиваются следующим образом:

— числа выстраиваются от наименьшего отрицательного к наибольшему положительному;
— значения даты и времени сортируются в хронологическом порядке от самого раннего к самому позднему;
— текстовые данные сортируются по алфавиту;
— столбец логических значений будет начинаться со значения *ЛОЖЬ* и заканчиваться значением *ИСТИНА*;
— пустые ячейки располагаются в конце списка.

Если задан убывающий порядок сортировки, то пустые ячейки также располагаются в конце списка, а данные всех остальных типов сортируются в обратном описанному ранее порядку.

Пользовательский порядок сортировки заключается в сортировке по дням недели либо по названиям месяцев года. Можно создать свой собственный пользовательский порядок сортировки.

Поле, по которому выполняется сортировка, называется ключом сортировки. Для упорядочивания списка по одному ключу можно воспользоваться кнопками на панели инструментов: *Сортировка по возрастанию*  , *Сортировка по убыванию*  . Предварительно выделяется любая ячейка того поля, по которому необходимо отсортировать список.

Сортировать список можно с помощью команды *Сортировка* в меню *Данные*. Команда *Сортировка* позволяет сортировать максимум по трем ключам, т. е. список сортируется сначала по одному полю, затем внутри отсортированных значений этого поля выполняется сортировка по второму полю, и аналогично внутри отсортированным значениям второго поля можно отсортировать по третьему ключу. Сортировка по нескольким ключам возможна в том случае, если первый и второй ключи сортировки имеют повторяющиеся значения. Если ключей сортировки больше трех, команда *Сортировка*

выполняется еще раз. Причем четвертый ключ имеет приоритет над первыми тремя, т. е. список сортируется по четвертому ключу в первую очередь.

Если в дальнейшем необходимо вернуться к первоначальному порядку сортировки, следует воспользоваться индексом. Индекс — это поле, содержащее уникальное значение для каждой записи, например порядковый номер.

Порядок действий при выполнении сортировки

1. Выделите любую ячейку внутри сортируемого списка.
2. Выберите команду *Данные — Сортировка*. На экране отобразится диалоговое окно *Сортировка* (рис. 1.1).

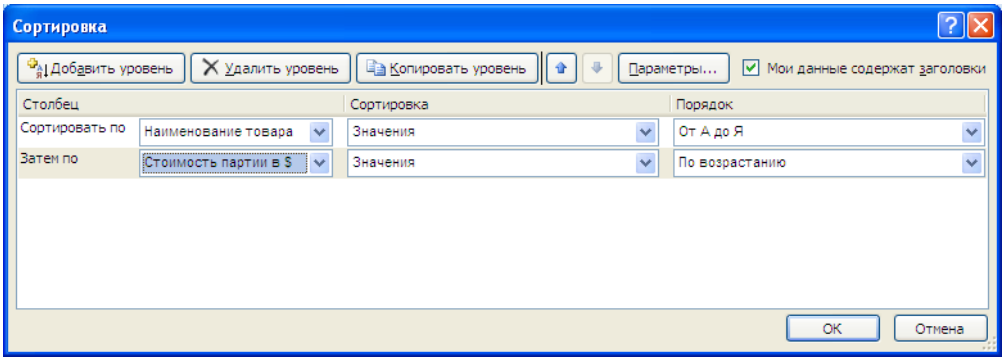


Рис. 1.1. Сортировка списка по двум полям

3. В окне диалога команды заполните все области строки *Сортировать по*.
4. В области *Столбец* нажмите кнопку раскрывающегося списка и выберите имя поля, по которому необходимо отсортировать список.
5. В области *Сортировка* по умолчанию устанавливается вариант *Значения* (при необходимости выберите вариант *Цвет ячейки* или *Цвет шрифта*).
6. В области *Порядок* устанавливается тип сортировки: *По возрастанию*, *По убыванию* (для числового типа данных); *От А до Я*, *От Я до А* (для текстового типа данных).

Примечание. Вариант *Настраиваемый список*, выбранный в области *Порядок*, позволяет задать пользовательский порядок сортировки, например по дням недели.

7. Для сортировки списка по второму (третьему) полю в диалоге команды нажмите кнопку *Добавить уровень* и заполните все области строки *Затем по*.
8. Кнопка *Копировать уровень* в диалоге команды позволяет добавить строку *Затем по* с параметрами, аналогичными установленным в предыдущем уровне.
9. Нажмите кнопку *ОК*.

Примечание. В диалоговом окне *Параметры сортировки*, отображаемом на экране нажатием кнопки *Параметры* диалогового окна *Сортировка*, можно потребовать учета регистра символов или задать сортировку данных по столбцам. Для сортировки списка по столбцам выберите переключатель *Столбцы диапазона*.

1.2. Фильтрация списка

Фильтрация — это выбор из списка записей, удовлетворяющих какому-либо критерию отбора. Фильтрация выполняется на вкладке *Данные* с помощью команд *Фильтр* и *Дополнительно* (*Расширенный Фильтр*). Команды *Фильтр* и *Дополнительно* фильтруют список, а команда *Очистить* отменяет фильтрацию.

Использование команды *Фильтр*

Команда *Фильтр* (автофильтр) позволяет очень быстро, с помощью элементарных действий мыши, отфильтровать список. *Фильтр* работает по принципу: «включил — выключил». После выбора команды все заголовки полей снабжаются кнопками раскрывающихся списков, используемых для фильтрации (фильтр — «включен»). Повторный выбор команды убирает кнопки раскрывающихся списков (фильтр — «выключен»).

Порядок действий при выполнении команды *Фильтр*

1. Выделите любую ячейку внутри фильтруемого списка.
2. Выберите команду *Данные — Фильтр*.
3. Нажмите кнопку раскрывающегося списка, соответствующую полю, по которому выполняется отбор записей. Выполните следующие действия:

— снимите флажок *Выделить все*;
— установите флажок напротив конкретного значения поля для вывода на экран записей, содержащих выбранное значение данного поля;
— нажмите кнопку *ОК*.

4. Для отбора записей внутри диапазона значений числового поля выберите команду *Числовые фильтры* (рис. 1.2), затем одну из подкоманд: *равно*, *не равно*, *больше*, *больше или равно*, *меньше*, *меньше или равно*, *между*, *Настраиваемый фильтр*.

Примечание. Все перечисленные подкоманды выводят на экран диалог *Пользовательский автофильтр*, в котором необходимо сформировать условие отбора записей по активному полю.

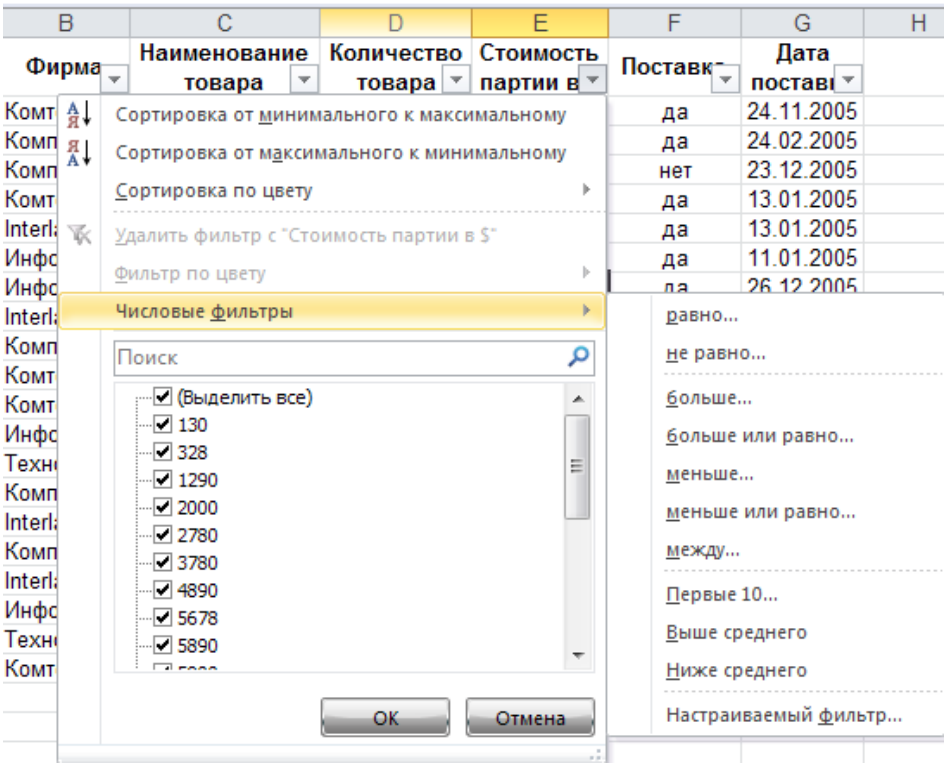


Рис. 1.2. Работа с командой *Числовые фильтры*

5. Если активное поле является текстовым, то в меню будет представлена команда *Текстовые фильтры* и подкоманды: *начинается с*, *заканчивается на*, *содержит*, *не содержит*.

6. Подкоманда *Первые 10* команды *Числовые фильтры* выводит на экран заданное число записей с наибольшими (наименьшими) по данному полю значениями.

Примечание. После выполнения фильтрации кнопка раскрывающегося списка того поля, которое использовалось в качестве критерия отбора, будет снабжена знаком фильтрации.

Использование команды *Дополнительно (Расширенный фильтр)*

Расширенный фильтр позволяет задавать более сложные условия фильтрации и выполнять копирование отфильтрованного списка. Критерии отбора задаются в любом месте рабочего листа и оформляются в виде отдельной таблицы. Расширенный фильтр в отличие от автофильтра позволяет отбирать не только записи, но и поля из исходного списка.

Перед выполнением команды *Расширенный фильтр* необходимо:

— составить таблицу критериев отбора;

— скопировать в любое свободное место листа заголовки тех полей списка, которые требуется отобрать в процессе фильтрации (для вывода отфильтрованной таблицы в ячейки листа, расположенные вне списка).

Создание таблицы критериев

1. Поместите таблицу критериев отбора над списком в любое свободное место листа (между таблицей критериев и списком должна остаться как минимум одна пустая строка).
2. В первую строку таблицы критериев скопируйте заголовки тех полей списка, по которым будет выполняться отбор данных.
3. В ячейки, расположенные под заголовками, введите сами критерии (условия отбора).

Критерии отбора

Критерии отбора могут быть как простые, так и вычисляемые. Простые критерии отбора — это либо конкретные значения полей, по которым ведется отбор, либо логические выражения, использующие знаки сравнения: >, ≥, <, ≤, <>. При задании критериев можно воспользоваться следующими символами шаблона: символ «*» обозначает произвольную последовательность любых символов, символ «?»

обозначает любой единичный символ. Все критерии (как простые, так и вычисляемые), которые заданы в одной строке, выполняются одновременно — аналог логического оператора *И*. Для объединения критериев с помощью условного оператора *ИЛИ* критерии задаются в разных строках таблицы критериев.

На рис. 1.3 представлен результат фильтрации списка по условию, состоящему из трех простых критериев. Отобраны записи с данными, относящимися к группе ИМ-14011, в которых значения поля *Сумма МЛ* находятся в диапазоне от 100 до 150. В критерии, используемом для отбора значений по полю *Группа*, используется символ шаблона «*» (рис. 1.3).

	A	B	C	D	E	F	O
1		**Таблица критериев**					
2		Группа	Сумма МЛ	Сумма МЛ			
3		*11	>100	<150			
4							
5	**№**	**ФИО**	**Группа**	**МЛ1**	**МЛ2**	**МЛ3**	**Сумма МЛ**
8	3	Вахмистров И.С.	ИМ-14011	28	10	4	106
10	5	Выскребец Е.Ю.	ИМ-14011	19	12	15	121
11	6	Гордюшев И.А.	ИМ-14011	18	17	17	133
12	7	Гуров И.И.	ИМ-14011	16	4	18	102
15	10	Папаков В.И.	ИМ-14011	23	12	10	144

Рис. 1.3. Задание простых критериев

При создании вычисляемого критерия учитывают следующие правила.

1. Заголовок вычисляемого критерия не должен совпадать ни с одним заголовком поля фильтруемого списка. Можно ввести новый заголовок или оставить ячейку заголовка пустой.
2. Формула критерия должна ссылаться хотя бы на одно поле списка.
3. Ссылки на ячейки, с которыми будет выполняться сравнение, должны быть абсолютными.
4. Ссылка на ячейку поля, значения которого будут сравниваться с эталонными, должна быть относительной.
5. Формула критерия является логической формулой, поэтому возвращает значение *ИСТИНА* или *ЛОЖЬ* (рис. 1.4).

C3 =О6>СРЗНАЧ($O$6:$O$72)

	A	B	C	D	E	F	O	Z
2		Группа						
3		ИМ-14012	ИСТИНА					
4								
5	**№**	**ФИО**	**Группа**	**МЛ1**	**МЛ2**	**МЛ3**	**Сумма МЛ**	**Сумма ИНФ**
29	24	Агафонов В.А.	ИМ-14012	27	10	5	131	92
32	27	Борисова А.А.	ИМ-14012	25	11	0	103	70
34	29	Высоких С.О.	ИМ-14012	22	4	13	106	84

Рис. 1.4. Задание вычисляемого критерия

Порядок действий при выполнении команды *Расширенный фильтр*

1. Выделите любую ячейку списка.
2. Выберите команду *Данные — Дополнительно*. На экране отобразится диалоговое окно *Расширенный фильтр*.

В диалоге команды:

— область *Исходный диапазон* заполняется автоматически (проверьте, правильно ли был определен диапазон для фильтрации);

— в область *Диапазон условий* введите диапазон ячеек, который содержит таблицу критериев;

— установите в группе *Обработка* переключатель *Скопировать результат в другое место* для вывода отфильтрованной таблицы в ячейки листа, расположенные вне списка. После этого в области *Поместить результат в диапазон* укажите диапазон, в который будет помещена отфильтрованная таблица;

— нажмите кнопку *ОК*.

Примечание. Для вывода всех полей отфильтрованного списка в области *Поместить результат в диапазон* достаточно указать одну любую ячейку на свободном месте текущего листа, которая станет левым верхним углом диапазона копирования. Для вывода определенных полей списка указывают диапазон тех ячеек, в которые были скопированы заголовки полей списка до выполнения команды.

1.3. Подведение промежуточных итогов

Подведение промежуточных итогов — это получение результатов вычислений по группам записей списка. Подведение итогов выполняется с помощью команды *Промежуточные итоги* на вкладке *Данные* (группа *Структура*). Команда *Промежуточные итоги* позволяет пользователю, не составляя формул и не преобразуя список, получить промежуточные и общие итоги в виде дополнительных строк списка, а также структурировать данный список.

Перед выполнением команды *Промежуточные итоги* необходимо определить «промежутки», по которым будут подводиться итоги. Имеющийся список должен быть разбит на отдельные группы записей. На группы список можно разбить по любому полю, в котором есть повторяющиеся значения. Для этого выполняется сортировка списка по данному полю. В результате сортировки записи с одинаковыми значениями поля собираются в одну группу. Сколько разных значений присутствует в данном поле, столько групп будет обработано при выполнении команды *Промежуточные итоги*.

Список функций для подведения промежуточных и общих итогов приведен в табл. 1.1.

Таблица 1.1

Функции для подведения итогов

Функция	Итог по группе и по полю в целом
Сумма	Сумма всех значений
Количество значений	Количество элементов
Среднее	Среднеарифметическое значение элементов
Максимум	Наибольшее значение
Минимум	Наименьшее значение
Произведение	Произведение всех значений
Количество чисел	Количество ячеек, содержащих числовые значения
Смещенное отклонение	Значение стандартного отклонения по выборке
Несмещенное отклонение	Значение стандартного отклонения по совокупности
Смещенная дисперсия	Значение дисперсии по выборке
Несмещенная дисперсия	Значение дисперсии по совокупности

При создании промежуточных итогов обрабатываемый список автоматически структурируется. Слева от списка появляются элементы структуры: точки, линии и кнопки.

Точки обозначают детальные данные, линии охватывают группы записей или весь список. Кнопка – замыкает линию структуры группы или линию структуры всего списка и используется для скрытия отдельной группы записей или всего списка. Кнопка 1 позволяет скрыть список и промежуточные итоги, оставив строку заголовков полей и строку с общим итогом. После нажатия кнопки 2 на листе остается строка заголовков полей, строки с промежуточными итогами (по группам записей) и строка с общим итогом. Нажатие кнопки 3 возвращает на экран все записи списка.

Порядок действий при выполнении команды *Промежуточные итоги*

1. Отсортируйте список по полю, которое будет делить список на группы записей.
2. Выделите любую ячейку списка.
3. Выберите команду *Данные — Промежуточные итоги*. На экране отобразится диалоговое окно *Промежуточные итоги*.

В диалоге команды:

— в области *При каждом изменении в:* выберите из списка имя поля, по которому выполнялась сортировка для деления списка на группы записей;

— в области *Операция* выберите функцию, которая будет использоваться для подведения итогов;

— в области *Добавить итоги по:* установите флажки против имен тех полей, по которым необходимо подвести итоги;

— установите флажок *Заменить текущие итоги*, если команда выполняется повторно и необходимо заменить старые итоги новыми. Снимите флажок, если хотите сохранить старые итоги;

— нажмите кнопку *ОК*.

Чтобы вернуться к исходному виду списка (без промежуточных итогов), нужно нажать кнопку *Убрать все* в диалоге команды.

1.4. Создание сводной таблицы Excel

Сводные таблицы являются очень удобным инструментом для просмотра и анализа больших объемов информации, представленных в виде списка (базы данных) *Excel*. Предположим, у вас есть таблица, состоящая из двадцати полей, причем пять из них содержат числовую информацию, а остальные поля содержат текстовые данные. Если вам требуется посмотреть результаты только по одному или двум числовым полям, причем результаты должны быть представлены в виде сумм исходных значений и показывать зависимость от конкретных значений двух текстовых полей вашей таблицы, то, конечно, необходимо построить сводную таблицу. Такую сводную таблицу легко можно будет модифицировать, например, поменять функцию *Сумма* на функцию *Количество* или использовать значения других полей исходной таблицы.

Функции, которые можно использовать в сводной таблице, представлены в таблице 1.2.

Таблица 1.2

Функции для вычислений в сводной таблице

Функция	Итог
Сумма	Сумма значений
Количество	Количество значений
Среднее	Среднеарифметическое значение элементов
Максимум	Наибольшее значение
Минимум	Наименьшее значение
Произведение	Произведение всех значений
Количество чисел	Количество ячеек, содержащих числовые значения
Смещенное отклонение	Значение стандартного отклонения по выборке
Несмещенное отклонение	Значение стандартного отклонения по совокупности
Смещенная дисперсия	Значение дисперсии по выборке
Несмещенная дисперсия	Значение дисперсии по совокупности

Порядок действий при построении сводной таблицы

1. Выделите любую ячейку исходной таблицы.
2. На вкладке *Вставка* выберите команду *Сводная таблица*.
3. В появившемся на экране окне диалога *Создание сводной таблицы* выполните нижеперечисленные действия:

1) проверьте, правильно ли был определен диапазон для построения сводной таблицы (в окне *Создание сводной таблицы* область *Таблица или диапазон* заполняется автоматически);
2) выберите один из переключателей:
   — *на новый лист* — сводная таблица будет помещена на новом рабочем листе, начиная с ячейки *A*1 (новый лист программа добавит к существующим листам автоматически);
   — *на существующий лист* — после выбора данного переключателя в поле *Диапазон* необходимо ввести адрес левой верхней ячейки диапазона, в который будет помещена сводная таблица.

4. Заполните макет сводной таблицы на новом листе, выполнив следующие действия (рис. 1.5):
   1) из области *Список полей сводной таблицы*, расположенной справа, переместите поля, которые будут представлять заголовки строк и столбцов сводной таблицы, в соответствующие области макета *Названия строк* и *Названия столбцов*;
   2) поля, содержащие данные, которые требуется обработать с помощью вычислительных функций, поместите в область *Значения*;
   3) поле, которое позволит менять вид сводной таблицы в зависимости от выбранного значения данного поля, поместите в область *Фильтр отчета*;
   4) для изменения расположения полей перетащите их из одной области в другую;
   5) чтобы удалить поле из сводной таблицы, перетащите его обратно в область *Список полей*.

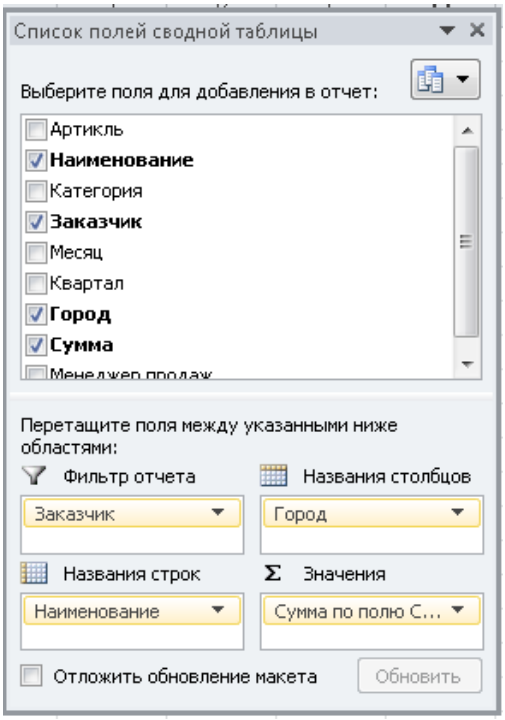


Рис. 1.5. Процесс создания сводной таблицы

Примечание. Числовое поле, помещенное в область *Перетащите сюда поля значений*, автоматически обрабатывается функцией *Сумма*, текстовое поле обрабатывается функцией *Количество*. Чтобы назначить другую итоговую функцию, откройте контекстное меню на любой ячейке в области обрабатываемых данных и выберите команду *Параметры полей значений*. В диалоговом окне *Параметры поля значений* в списке *Операция* выберите нужную функцию для обработки данных (рис. 1.6).

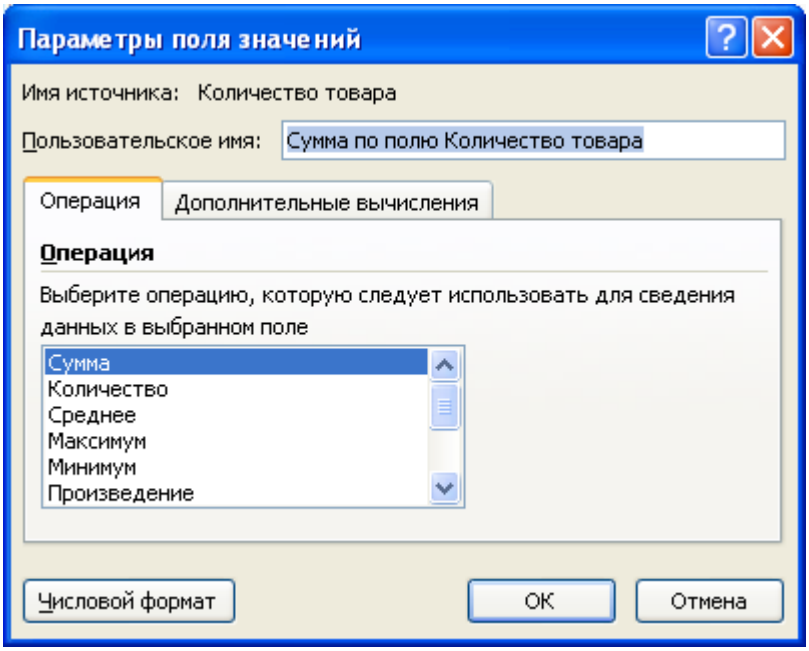


Рис. 1.6. Выбор функции для обработки табличных данных при построении сводной таблицы

Создание вычисляемого поля в сводной таблице

1. Перейдите на лист с готовой сводной таблицей.
2. Выделите любую ячейку в области данных сводной таблицы и выберите команду *Поля, элементы и наборы — Вычисляемое поле* на вкладке *Параметры* (*Работа со сводными таблицами).*
3. В диалоге *Вставка вычисляемого поля* в области *Имя* введите имя будущего поля сводной таблицы.
4. В области *Формула* введите формулу, использую имена полей из области *Поля*, например: =СУММА*100/118.

Примечание. Поместить поле в область *Формула* можно, выполнив двойной щелчок по имени поля в области *Поля*.

Редактирование сводной таблицы

При редактировании готовой сводной таблицы выполните следующие действия.

1. Щелкните правой кнопкой мыши на сводной таблице и выберите команду *Показать список полей* из контекстного меню. На экране появится окно *Список полей сводной таблицы* (рис. 1.7). С помощью мыши перетащите нужное поле в нужную область таблицы.
2. Для выбора функции используйте команду *Параметры поля значений* из контекстного меню.
3. При необходимости в готовой сводной таблице переместите поле данных из одной области в другую, перетаскивая его мышью.

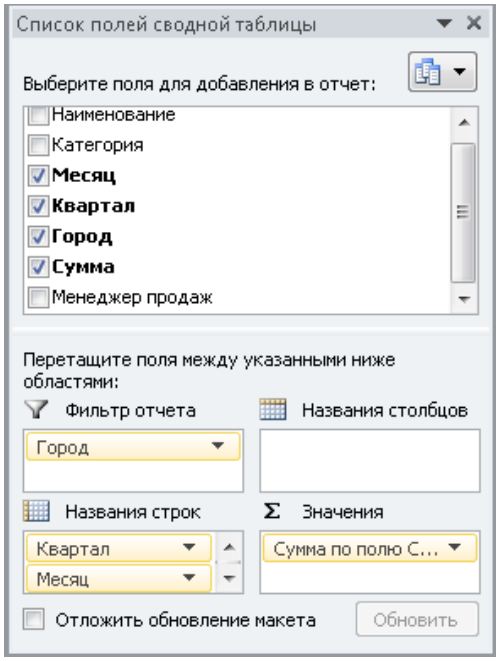


Рис. 1.7. Распределение полей исходного списка по областям сводной таблицы

4. Для удаления поля из сводной таблицы переместите поле из области представления данных сводной таблицы в область списка полей или выберите команду *Удалить* из контекстного меню, открытого на имени удаляемого поля.
5. Для обновления сводной таблицы после изменений, внесенных в окне *Список полей сводной таблицы*, используйте команду *Обновить* из контекстного меню, открытого на любом поле сводной таблицы.
6. Если сводная таблица содержит поле с данными типа *Дата*, то его значения группируются в готовой сводной таблице следующим образом:
   — выделите любую ячейку с датой в готовой сводной таблице;

— выберите команду *Группировать* из контекстного меню;
— в диалоговом окне *Группирование* выберите шаг группировки: *Дни, Месяцы, Кварталы, Годы*;
— выбрав шаг группировки — *Дни*, введите число в поле *Количество дней*;
— нажмите кнопку *ОК*.

1.5. Практические задания «Фильтрация списков. Определение промежуточных итогов. Построение сводных таблиц»

Задание 1. Построить сводную таблицу «Годовой итоговый отчет».

1. Создайте новую книгу в программе MS Excel.
2. Скопируйте таблицу *Список*.
3. В диалоге *Создание сводной таблицы* проверьте диапазон и выберите опцию *На новый лист*.
4. Перетащите с помощью мыши поля списка в следующие области сводной таблицы:
   1) *Наименование* в область строк;
   2) *Город* в область столбцов;
   3) *Сумма* в область значений;
   4) *Заказчик* в область *Фильтр;*
5. Выделите любую числовую ячейку в сводной таблице и выполните команду *Числовой формат* из *Контекстного меню*. Обнулите десятичную часть и установите *Разделитель разрядов*.
6. Отредактируйте сводную таблицу:
   1) добавьте поле *Категория* в область строк, расположив его над полем *Наименование*;
   2) поле *Город* перенесите в область *Фильтр;*
   3) перенесите поля *Месяц* и *Квартал* в область столбцов, расположив *Квартал* над полем *Месяц*.
7. Выполните форматирование сводной таблицы, выбрав любой формат на вкладке *Конструктор*.
8. Для отображения чисел используйте гистограммы (степень заливки будет зависеть от величины числа). Выделите несколько любых ячеек сводной таблицы с числами и выберите команду *Условное форматирование — Гистограммы — Градиентная заливка*.

9. Примените форматирование ко всем числовым ячейкам таблицы (кроме итоговых), щелкнув по значку *Параметры форматирования*, расположенному справа от ячеек с гистограммами.
10. Выберите команду *Вставить срез* на вкладке *Параметры* (*Работа со сводными таблицами*).
11. Установите флажок *Менеджер продаж* и поработайте со срезом.
12. Добавьте срез *Город*, предварительно удалив его из фильтра.
13. Сделайте копию листа со сводной таблицей. На листе-копии удалите фильтры, срезы и скройте детальные данные, используя элементы структуры в строке заголовков сводной таблицы.
14. Выберите команду *Список полей* сводной таблицы из контекстного меню.
15. Добавьте поле *Квартал* в область значений.
16. Поменяйте имена заголовков в сводной таблице (рис. 1.8).

	Названия столбцов				
	I		II		III
Названия строк	Стоимость заказов	Количество заказов	Стоимость заказов	Количество заказов	Стоимость заказов
Кондитерские изделия	**550 214**	**15**	**412 870**	**8**	**766 402**
Конфеты	110 790	3	175 504	2	192 131
Печенье	157 377	6	158 544	3	208 091
Торты	282 047	6	78 822	3	366 180
Консервы	**598 493**	**14**	**290 828**	**6**	**1 295 871**
Зеленый горошек	372 937	8	96 213	1	518 091
Икра кабачковая	125 913	3	90 592	2	460 646
Икра лососевых рыб	99 643	3	104 023	3	317 134
Крупы	**538 673**	**11**	**597 737**	**13**	**1 244 463**

Рис. 1.8. Сводная таблица без детальных данных

Задание 2. Построить сводную таблицу с вычисляемым полем.

1. Создайте новую сводную таблицу (суммы продаж по категориям).
2. Добавьте вычисляемое поле с помощью команды *Поля, элементы и наборы* (*Работа со сводными таблицами — Параметры*).
3. В диалоге *Вставка вычисляемого поля* заполните поля *Имя* и *Формула* (имя — *Продажи без НДС*. Формула: =СУММА*100/118) (рис. 1.9).

Названия строк	Продажи	Продажи без НДС
Кондитерские изделия	2 586 855р.	2 192 250р.
Консервы	2 635 201р.	2 233 221р.
Крупы	2 860 228р.	2 423 922р.
Молочные продукты	5 914 641р.	5 012 407р.
Морепродукты	3 306 544р.	2 802 156р.
Мясные продукты	9 847 888р.	8 345 668р.
Напитки	5 855 912р.	4 962 637р.
Овощи	2 574 734р.	2 181 978р.
Орехи	1 166 085р.	988 208р.
Сухофрукты	1 866 097р.	1 581 438р.
Фрукты	3 470 190р.	2 940 839р.
Общий итог	**42 084 374**	**35 664 724**

Рис. 1.9. Сводная таблица с вычисляемым полем

Задание 3. Работа с командой *Форматировать как таблицу.*

Команда *Форматировать как таблицу* позволяет выбрать любой стиль из коллекции стилей таблиц, а также выполнять следующие операции:

1) фильтрация списка;
2) автоматизация ввода и форматирования вновь добавляемых строк и столбцов;
3) автоматическое заполнение всего поля формулой, введенной в первую ячейку данного поля;
4) добавление строки итогов.

В задании используются некоторые функции.

Функция ОКРУГЛТ округляет число до ближайшего кратного числу, заданного вторым аргументом.

Синтаксис: *=ОКРУГЛТ (число; точность).*

Число (обязательный аргумент) — число либо ссылка на ячейку, содержащую число.

Точность (обязательный аргумент) — число, для которого необходимо найти ближайшее кратное к первому аргументу. В случае задания нулевого значения функция всегда будет возвращать 0.

Знаки двух аргументов должны совпадать, иначе функция вернет ошибку.

=ОКРУГЛТ (5,45;0) — формула возвращает значение 0.

=ОКРУГЛТ (5,45;3) — формула возвращает значение 6, т. к. 6/3 = 2 ближе, чем 3/3 = 1.

Функция ПРОСМОТР (векторная форма) вернет наибольшее значение из имеющихся, которое меньше искомого.

Синтаксис: *=ПРОСМОТР (искомое_значение; вектор_поиска; [вектор результата]).*

1. Перейдите на лист с исходным списком и выделите любую ячейку в таблице. На вкладке *Главная* в группе *Стили* нажмите *Форматировать как таблицу*.
2. Нажмите *CTRL + клавиша «стрелка вниз»* и добавьте новую строку с любыми данными из существующих в таблице для проверки автоматизации ввода и форматирования (для ввода можно использовать автоматизацию или команду контекстного меню *Выбрать из раскрывающегося списка*).
3. Добавьте новый столбец в конец таблицы с именем *СуммаСНаценкой* и введите формулу для вычисления значений поля, используя данные табл. 1.3.

Таблица 1.3

Коэффициенты наценки в зависимости от величины суммы

Сумма	Коэффициент наценки
< 10000	1,4
< 20000	1,3
< 50000	1,2
< 100000	1,2
> 100000	1,1

=ОКРУГЛТ (G2*ПРОСМОТР (G2;{0;10000;20000;50000;100000}; {1,4;1,3;1,2; 1,2;1,1});50)

4. Добавьте строку итогов, установив соответствующий флажок на вкладке *Конструктор (работа с таблицами)* — итоги динамические, т. е. меняются в зависимости от результата фильтрации. Посчитайте среднее значение суммы продаж, максимальное значение суммы с наценкой, количество по полю *Менеджер* (при фильтрации это будет число заказов каждого менеджера).
5. Установите для поля *Сумма* числовой формат с нулевой десятичной частью и разделителем разрядов. Для выделения поля используйте *SHIFT + CTRL + клавиша «стрелка вниз»*.

6. Скопируйте формат на поле *СуммаСНаценкой* (выберите команду *Формат по образцу* и щелкните по любой ячейке поля).
7. Выполните команду *Рецензирование — Защитить Лист*. Установите все флажки в диалоге команды. Снять защиту можно в диалоге *Формат ячеек*.
8. Отфильтруйте список, выбрав город Екатеринбург и суммы продаж от 50 до 100 тыс. руб.

Самостоятельная работа

1. Создайте сводную таблицу с суммами продаж по каждому продукту при наличии срезов по менеджеру и городу.
2. Добавьте поле *СуммаСНаценкой* и вычисляемое поле *Наценка*.
3. Удалите срез по городу и добавьте город в область столбцов.
4. Выполните в сводной таблице числовое и условное форматирование (покажите первые 10 сумм по Екатеринбургу).
5. Определите, кто из менеджеров оформил максимальное число заказов (используйте исходную таблицу с данными).

Раздел 2.
Решение оптимизационных задач с помощью программы «Поиск решения»

В программе MS Excel есть инструмент, который называется *Поиск решения*. Это встроенная программа, относящаяся к классу надстроек, позволяющая расширить возможности программы MS Excel. *Поиск решения* используют для подбора аргументов так называемой целевой функции. Особенно часто применяют данный инструмент для решения оптимизационных задач, базирующихся на линейном программировании.

Линейное программирование — область математики, разрабатывающая теорию и численные методы решения задач нахождения экстремума (максимума или минимума) линейной функции многих переменных при наличии линейных ограничений, т. е. линейных равенств или неравенств, связывающих эти переменные.

Линейное программирование используют при поиске оптимального решения в экономических задачах, связанных с управлением запасами, распределением ресурсов, определением максимальной прибыли и пр. Ниже приведен список подобных задач:

1) задача оптимального распределения ресурсов при планировании выпуска продукции на предприятии (задача об ассортименте);
2) задача максимального выпуска продукции при заданном ассортименте;
3) задача о смесях (рационе, диете);
4) транспортная задача;
5) задача о рациональном использовании имеющихся мощностей;
6) задача о назначениях.

Перед использованием инструмента *Поиск решения* необходимо в соответствии с теорией линейного программирования создать ма-

тематическую модель задачи. Рассмотрим пример разработки оптимального плана ремонта легковых автомобилей.

Исходные данные:

— потребности в ремонте;
— производственные мощности автомастерской;
— наличие людских ресурсов;
— *xj* — объем ремонта автомобиля *j*-го типа (*j* = 1, 2, … *n*);
— *bi* — объем имеющихся в наличии производственных ресурсов *i*-го вида; *i* = 1, 2, … *m*;
— *aij* — расход *i*-го вида ресурсов на ремонт одного автомобиля *j*-го типа;
— *Cj* — прибыль, получаемая предприятием за один отремонтированный автомобиль *j*-го типа.

Постановка задачи

1. Определить значения переменных *xj*, при которых линейная функция *F*(*x*) имеет оптимальное значение, учитывая существующие ограничения.

2. Записать ограничения в виде математических неравенств — система *m* неравенств с *n* переменными (1):

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1j}x_j + \ldots + a_{1n}x_n \le b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2j}x_j + \ldots + a_{2n}x_n \le b_2 \\ a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ij}x_j + \ldots + a_{in}x_n \le b_i \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mj}x_j + \ldots + a_{mn}x_n \le b_m. \end{cases} \quad (1)$$

Линейную функцию *F*(*x*), для которой ищется экстремальное значение, называют *целевой функцией*.

Решение задач линейного программирования в среде MS Excel предполагает следующий порядок действий:

1) составление таблицы исходных данных;
2) ввод данных в ячейки листа рабочей книги *Excel*;
3) ввод в ячейки листа формулы целевой функции и формул, определяющих ограничения задачи;
4) выполнение программы *Поиск решения*.

Практические задания «Поиск решения. Разработка сценариев»

Часть 1. Оптимизация рациона питания

Постановка задачи

Определить количество каждого продукта, входящего в рацион питания с учетом следующих показателей рациона:

— рацион должен быть сбалансирован по количеству белков, жиров и углеводов;

— стоимость дневного рациона должна быть минимальной.

Для решения задачи в среде программы MS Excel необходимо использовать инструмент *Поиск решения*.

Исходные данные

1. Наименования продуктов: мясо, рыба, масло сливочное, крупа гречневая, картофель.
2. Количество белков, жиров и углеводов в единице (100 г) каждого продукта.
3. Общее количество белков, жиров и углеводов должно быть 82, 65, 128 соответственно.
4. Стоимость единицы каждого продукта.

	A	B	C	D	E	F	G	H	
1	**Продукты**	Мясо	Рыба	Масло слив.	Крупа гречневая	Картофель	**План**	**Ограничения**	
2	Белки	18,6	20	0,5	10,8	2	82	101,9	
3	Жиры	16	8,1	82,5	3,2	0,4	65	65,0	
4	Углеводы	0	0	0,8	56	16,3	128	128,0	
5	Количество продукта (ед. изм. 100 г)	2	2	0,1150421	2,284070827	0			
6	Стоимость единицы каждого продукта	30	25	40	5	2	**Стоимость**	126,0	min

Рис. 2.1. Пример заполнения ячеек листа *Рацион питания*

Порядок выполнения работы

1. Создайте новую книгу в программе MS Excel.
2. Переименуйте Лист 1, используя имя *Рацион питания*.
3. Заполните ячейки Листа 1 в соответствии с рис. 2.1. Ячейки строки *Количество продукта* будут содержать подбираемые значения количества каждого продукта.
4. Отформатируйте диапазон *B*5:*F*5, используя числовой формат с двумя знаками после запятой.

5. В ячейку *H*6 введите формулу (целевую функцию) для вычисления количества денег, потраченных на данный рацион питания.
   — в формуле используется функция *СУММПРОИЗВ* (сумма произведений стоимости на количества каждого продукта);
   — аргументы функции: диапазон ячеек со стоимостями продуктов, диапазон ячеек с количествами продуктов.
6. Оформите блок ячеек с заголовком *Ограничения*, введя формулы, вычисляющие количества белков, жиров и углеводов рациона (используйте функцию *СУММПРОИЗВ).*
7. Выполните команду *Поиск решения* меню *Данные*. Если команда в меню отсутствует, выполните команду *Файл — Параметры — Надстройки* и активизируйте *Поиск решения.*
8. В диалоге команды выполните следующие действия:
   — в поле *Оптимизировать целевую функцию* введите адрес ячейки с целевой функцией;
   — установите опцию *Минимум*;
   — в поле *Изменяя ячейки переменных* введите адрес диапазона ячеек для количественных значений продуктов;
   — в области *В соответствии с ограничениями* введите три условия: количество мяса и рыбы — не менее 2 единиц для каждого продукта, количество белков, жиров, углеводов — не менее плановых значений;
   — установите флажок *Сделать переменные без ограничений неотрицательными*;
   — в списке *Выберите метод решения — Поиск решения линейных задач симплекс-методом*;
   — нажмите кнопку *Найти решение*.
9. В диалоге *Результаты поиска решения* нажмите кнопку *Сохранить сценарий*.
10. В окне *Сохранить сценарий* введите имя сценария *Рацион1.*
11. Внесите изменения в стоимость продуктов и выполните поиск решения.
12. Сохраните сценарий с именем *Рацион2.*
13. Выполните команду *Анализ Что-если — Диспетчер сценариев*.
14. В диалоге команды удостоверьтесь, что оба сценария сохранены и нажмите кнопку *Отчет*. Выберите тип отчета — *Структура.*
15. Проанализируйте отчет.

Часть 2. Оптимизация плана производства

Постановка задачи

Найти оптимальный план производства, при котором доход от реализации произведенной продукции должен быть максимальным.

Исходные данные

Предприятие производит два вида хлебобулочных изделий, используя запасы четырех видов ресурсов. Известны цена продукции и количество ресурсов (рис. 2.2).

Ресурсы	Продукт 1	Продукт 2	К-во ресурсов
Мука	0,6	0,5	120
Жиры	0,05	0,08	70
Сахар	0,2	0,6	66
Финансы	0,2	0,24	50
Цена	0,99	1,21	

Рис. 2.2. Нормы расходов ресурсов на производство единицы продукции

Экономико-математическая модель

X — количество изделий одного вида продукции (хлеб).

Y — количество изделий другого вида продукции (батон).

При существующих исходных данных прибыль должна быть максимальной. Целевая функция имеет вид: $F(x) = 0{,}99 \cdot X + 1{,}21 \cdot Y$.

Ограничения по запасам представлены следующими неравенствами:

$0{,}6 \cdot X + 0{,}5 \cdot Y \le 120$;

$0{,}05 \cdot X + 0{,}08 \cdot Y \le 70$;

$0{,}2 \cdot X + 0{,}6 \cdot Y \le 65$;

$0{,}2 \cdot X + 0{,}24 \cdot Y \le 50$;

$120 \le X \le 150$ (спрос);

$Y \ge 0$.

Порядок выполнения задания

1. Создайте на новом листе таблицу с исходными данными (рис. 2.2).
2. Добавьте колонку *Использовано*, в которую должны быть введены формулы для записи ограничений, и строку *План*, в ячей-

ках которой появятся искомые значения количества продукции (после выполнения программы *Поиск решения*).

3. Введите целевую функцию (прибыль).

Замечание. Во всех формулах используйте функцию *СУММПРОИЗВ*.

4. Запустите программу *Поиск решения*. Сохраните сценарий с именем *Запасы1*.
5. В диалоге *Результаты поиска решения* в списке *Тип отчета* выберите вариант *Устойчивость* и нажмите кнопку *ОК*.
6. Проанализируйте отчет.

Чтобы получить максимальный доход в размере 219,1 денежных единиц, нам нужно производить 150 единиц продукта 1 и 58,3 единиц продукта 2.

Отчет по устойчивости содержит такие параметры, как «Приведенная стоимость» и «Теневая цена». Приведенная стоимость показывает, насколько изменится целевая функция в случае принудительного включения единицы соответствующей продукции в оптимальный план выпуска. Теневая цена показывает, насколько изменится целевая функция при увеличении соответствующего ресурса (правой части ограничения) на единицу.

Отчет по устойчивости показывает, что ресурс *Сахар* находится в дефиците и его «Теневая цена» составляет 2 денежные единицы. Это означает, что при увеличении данного ресурса на 1 единицу, доход увеличится на 2 единицы и будет равен 221,1 денежных единиц.

Приведенная стоимость продукта 1 означает, что увеличение выпуска этого продукта увеличит доход на 0,6 денежных единиц за каждую дополнительную единицу продукции.

7. Увеличьте значение запаса сахара на 1 единицу и добавьте ограничение — количество изделий должно быть целым числом. Выполните поиск решения.
8. Сохраните сценарий с именем *Запасы2*.
9. Выполните команду *Анализ Что-если — Диспетчер сценариев*.
10. В диалоге команды удостоверьтесь, что оба сценария сохранены и нажмите кнопку *Отчет*. Выберите тип отчета — *Структура*.
11. Проанализируйте полученный отчет.

Раздел 3.
Работа с финансовыми функциями и построение таблицы данных

В программе MS Excel существует большое количество встроенных функций, позволяющих выполнять вычисления в разных областях финансовой деятельности. Рассмотрим некоторые функции из категории *Финансовые*.

3.1. Функция ЧПС (чистая приведенная стоимость инвестиции)

Синтаксис функции: *Функция ЧПС (ставка, значение1, [значение2], ...)*.

Функция возвращает величину чистой приведенной стоимости инвестиции, используя ставку дисконтирования, а также последовательность будущих выплат (отрицательные значения) и поступлений (положительные значения).

Аргументы функции *ЧПС:*

— *ставка* является обязательным аргументом и определяет ставку дисконтирования за один период.

— *значение 1, [значение 2]* … может быть от 1 до 254 аргументов, представляющих выплаты и поступления.

Значения выплат вводятся как отрицательные числа (со знаком минус), а значения поступлений — как положительные числа. Выплаты должны осуществляться регулярно, в конце каждого периода.

Данная функция, например, позволяет рассчитать текущую стоимость проекта, если известны капвложения в данный проект и суммы прибыли, которые приносит проект за определенные периоды, например, за каждый год.

Если аргумент идет с ссылкой на диапазон ячеек, то учитываются только числа в данном диапазоне. Пустые ячейки, логические и текстовые значения игнорируются.

При использовании данной функции необходимо учитывать, что временной срок инвестиции, значение которой рассчитывается, начинается за один период до даты платежа и заканчивается в день последней выплаты.

Примечание. Если первый денежный взнос приходится не на конец, а на начало первого периода, тогда первое значение добавляют к результату функции *ЧПС*, не включая его в список аргументов.

3.2. Функция АПЛ (линейный метод)

Функция *АПЛ* возвращает величину амортизации за текущий период с использованием линейного метода. Под амортизацией понимают уменьшение за единицу времени стоимости имущества (актива) в процессе эксплуатации.

Расчет амортизационных отчислений на предприятии выполняется для возможности определения налога на прибыль, для расчета средств, необходимых при проведении модернизации или расширении производства, для определения балансовой стоимости имущества предприятия.

Расчет равномерной амортизации с помощью функции *АПЛ* является самым простым, т. к. за каждый период списывается одно и то же значение денежных сумм. Функция возвращает величину амортизации за один период, рассчитанную линейным методом.

Синтаксис функции:

АПЛ (нач_стоимость, ост_стоимость, время_эксплуатации).

Аргументы функции *АПЛ:*

— *нач_стоимость* является обязательным аргументом и определяет начальную стоимость актива;

— *ост_стоимость* является обязательным аргументом и определяет стоимость в конце периода амортизации (ликвидационную стоимостью актива);

— *время_эксплуатации* является обязательным аргументом и определяет число периодов амортизации актива (срок полезного использования актива).

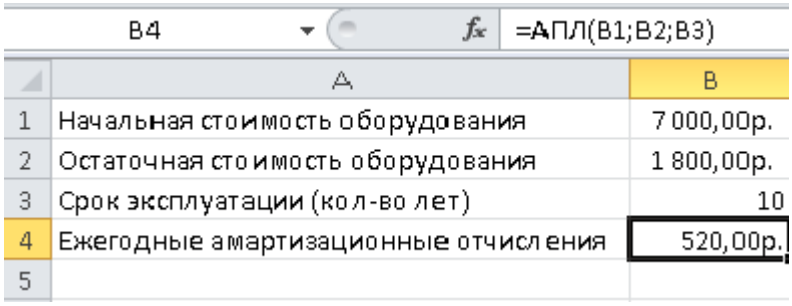


Рис. 3.1. Расчет амортизации с помощью функции *АПЛ*

Равномерная амортизация в условиях инфляции приводит к занижению себестоимости продукции, что в свою очередь приводит к повышению суммы налога с предприятия. Поэтому на предприятиях используют ускоренную амортизацию. В этом случае самые большие амортизационные отчисления — в первый год, с каждым последующим периодом они становятся меньше.

Для расчета ускоренной амортизации используются функция *АСЧ*, функция *ДДОБ* и другие.

3.3. Функция АСЧ (метод суммы чисел)

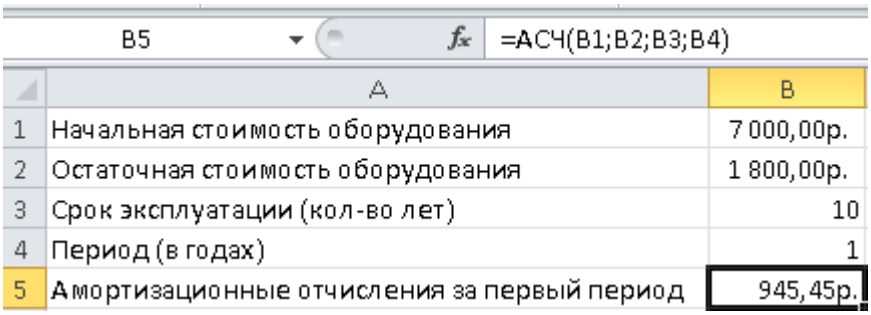
Функция *АСЧ* возвращает величину амортизации за текущий период с использованием метода суммы чисел.

При использовании метода суммы чисел выполняется списание по сумме чисел лет срока полезного использования. Годовая сумма амортизационных отчислений определяется путем умножения амортизируемой стоимости актива на дробь, в числителе которой — число лет, остающихся до конца срока полезного использования объекта, а в знаменателе — сумма чисел лет срока полезного использования объекта.

Сумма чисел лет срока полезного использования определяется путем простого суммирования (например, если срок полезного использования объекта 6 лет, то сумма чисел лет этого срока равна 21 (1 + 2 + 3 + 4 + 5 + 6)).

Синтаксис функции:
АСЧ (нач_стоимость, ост_стоимость, срок_эксплуатации, период).
Аргументы функции *АСЧ:*

— *нач_стоимость* является обязательным аргументом и определяет начальную стоимость актива;
— *ост_стоимость* является обязательным аргументом и определяет стоимость в конце периода амортизации;
— *срок_эксплуатации* является обязательным аргументом и определяет число периодов амортизации актива (срок полезного использования актива);
— *период* является обязательным аргументом и определяет временной период, за который рассчитывается амортизация.



	A	B
1	Начальная стоимость оборудования	7 000,00р.
2	Остаточная стоимость оборудования	1 800,00р.
3	Срок эксплуатации (кол-во лет)	10
4	Период (в годах)	1
5	Амортизационные отчисления за первый период	945,45р.

Рис. 3.2. Расчет амортизации с помощью функции *АСЧ*

На рис. 3.2 представлен результат расчета амортизации с помощью функции *АСЧ* за первый год эксплуатации оборудования. В последующие годы величина амортизационных отчислений будет уменьшаться.

3.4. Функция ДДОБ (метод двойного уменьшения остатка)

Функция *ДДОБ* используется для расчета ускоренной амортизации методом двойного уменьшения остатка, который состоит в том, что размер амортизации вычисляется по остаточной стоимости актива на начало отчетного периода и суммы амортизации, рассчитанной, исходя из срока полезного использования и коэффициента ускорения.

Метод двойного уменьшения остатка вычисляет амортизацию, используя увеличенный коэффициент. В *MS Excel* значение коэффициента автоматически полагается равным 2. Амортизация максимальна в первый период, в последующие периоды уменьшается.

Синтаксис функции:

ДДОБ (*нач_стоимость, ост_стоимость, время_эксплуатации, период, [коэффициент]*).

Аргументы функции *ДДОБ:*

— *нач_стоимость* является обязательным аргументом и определяет начальную стоимость актива;

— *ост_стоимость* является обязательным аргументом и определяет стоимость в конце периода амортизации. Остаточная стоимость может быть равна нулю;

— *время_эксплуатации* является обязательным аргументом и определяет количество периодов, за которые собственность амортизируется;

— *период* является обязательным аргументом и определяет временной период, за который рассчитывается амортизация;

— *коэффициент* является необязательным аргументом, определяет процентную ставку снижающегося остатка. Если коэффициент опущен, он полагается равным 2 (метод удвоенного процента со снижающегося остатка).

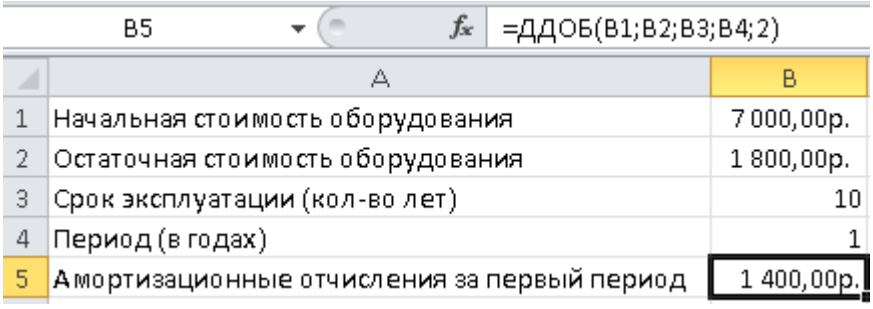


Рис. 3.3. Расчет амортизации с помощью функции *ДДОБ*

3.5. Практические задания «Использование финансовых функций и построение таблиц данных»

Задание 1. Оценить эффективность инвестиций на основе таблицы данных и функции ЧПС.

Исходные данные

Объем капвложений на конец года составил 1350 млн руб. При ставке дисконтирования 10 % инвестиционный проект принесет 430, 570, 680, 720 млн руб. соответственно за каждый год из последующих четырех лет.

Порядок выполнения работы

1. Используя функцию ЧПС, рассчитайте чистую приведенную стоимость инвестиционного проекта:
   — создайте новую рабочую книгу;
   — дайте имя Листу 1 *Инвестиционный проект*;
   — введите в ячейки листа исходные данные;
   — для ввода функции *ЧПС* вызовите *Мастер функций* и заполните поля аргументов;
   — в поле аргумента *Значение 1* введите адрес ячейки, содержащей значение капвложения, со знаком минус;
   — в поле аргумента *Значение 2* введите абсолютный адрес диапазона ячеек, содержащего значения поступлений.
2. Постройте таблицу данных и проанализируйте зависимость чистой стоимости проекта от различных ставок дисконтирования и начальных капвложений:
   — введите в ячейки листа, расположенные правее ячейки с формулой, арифметическую прогрессию, составленную из значений капвложений в пределах от 1350 до 1510 млн руб. с шагом 20 млн руб.;
   — введите в ячейки листа, расположенные ниже ячейки с формулой арифметическую прогрессию, составленную из значений ставки дисконтирования в пределах от 11 до 22 % с шагом 1 %;

— выделите прямоугольную область листа, включающего ячейку с формулой и две прогрессии, и выполните команду *Таблица данных*, открыв список *Анализ Что-если* на вкладке *Данные*;
— в поле *Подставлять значения по столбцам* введите адрес ячейки, содержащей исходное значение капвложения;
— в поле *Подставлять значения по строкам* введите адрес ячейки, содержащей исходное значение ставки дисконтирования.
— проанализируйте полученную таблицу данных.

Замечание. Если чистая приведенная стоимость инвестиционного проекта имеет положительное значение, то проект принесет прибыль; причем, чем больше это значение, тем выгоднее вкладывать деньги в данный проект. Если чистая приведенная стоимость инвестиционного проекта имеет отрицательное значение, то проект не принесет прибыли.

3. Постройте график зависимости *ЧПС* от ставки дисконтирования, используя в качестве ряда данных второй столбец таблицы данных.

Задание 2. Рассчитать амортизационные отчисления с помощью финансовых функций *АПЛ*, *АСЧ*, *ДДОБ*.

Исходные данные

Начальная стоимость имущества предприятия — 2 млн руб. Срок эксплуатации имущества — 10 лет. Остаточная стоимость имущества — 250 тыс. руб.

Порядок выполнения работы

1. Переименуйте Лист 2, используя имя *Расчет амортизации*.
2. Введите в ячейки листа исходные данные.
3. Сформируйте таблицу, представленную на рис. 3.4.
4. Представьте графически изменение амортизационных отчислений по годам для всех функций. Расположите график на листе *Расчет амортизации*.

C	D	E	F
Год	**ДДОБ**	**АСЧ**	**АПЛ**
1	400 000,00р.	318 181,82р.	175 000,00р.
2	320 000,00р.	286 363,64р.	175 000,00р.
3	256 000,00р.	254 545,45р.	175 000,00р.
4	204 800,00р.	222 727,27р.	175 000,00р.
5	163 840,00р.	190 909,09р.	175 000,00р.
6	131 072,00р.	159 090,91р.	175 000,00р.
7	104 857,60р.	127 272,73р.	175 000,00р.
8	83 886,08р.	95 454,55р.	175 000,00р.
9	67 108,86р.	63 636,36р.	175 000,00р.
10	18 435,46р.	31 818,18р.	175 000,00р.

Рис. 3.4. Расчет амортизации с помощью финансовых функций

Задание для самостоятельной работы

Постройте таблицу данных для выполнения анализа амортизационных отчислений по методу суммы чисел в зависимости от года и разных начальных стоимостей имущества (от 2 млн руб. до 4 млн 200 тыс. руб. с шагом 200 тыс. руб.).

Раздел 4.
Основы программирования на VBA (Visual Basic For Applications)

Язык программирования *VBA* (*Visual Basic for Applications*) был разработан фирмой Microsoft для работы с приложениями Microsoft Office (*Word, Excel, Access, PowerPoint, Outlook, FrontPage, InfoPath*). Данный язык программирования может также использоваться для работы с другими приложениями фирмы Microsoft (*Visio* и *Microsoft Project*), а также с приложениями, разработанными другими фирмами (*CorelDraw*, *AutoCAD* и т. п.).

Использование *VBA* в приложениях Microsoft Office, в частности в программе MS Excel, позволяет автоматизировать повторяющиеся операции, которые могут потребовать от пользователя большого количества времени. Эти операции могут быть связаны с такими задачами, как форматирование табличных данных, поиск нужных данных в табличных базах, сортировка и построение сводных таблиц. Язык *VBA* позволяет также выполнять работу по интегрированию нескольких офисных приложений, например, выводить результаты обработки табличных данных в документ MS Word.

VBA встроен в среду приложений Microsoft Office. Для работы с кодами *VBA* используются команды, расположенные на вкладке *Разработчик* окна программы MS Excel. Команда *Visual Basic* позволяет перейти в окно редактора *VBA* и приступить к созданию кода.

4.1. Особенности объектно-ориентированного программирования на VBA в MS Excel

Объектная модель приложения в MS Excel представляет собой набор следующих объектов, перечисленных в соответствии с иерархией: приложение *Application*, книга *Workbook* (семейство или коллекция *Workbooks*), лист *Worksheet* (семейство или коллекция *Worksheets*), диапазон ячеек листа *Range*.

У каждого вида объекта существует свой набор методов и свойств, который позволяет управлять этим объектом. В разделе 4.3 объекты и работа с объектами будет рассмотрена более подробно.

Классы в *VBA* оформляются в виде отдельных модулей классов. В окне модуля класса объявляется имя класса, свойства, методы (обычные процедуры и функции *VBA*). *VBA* поддерживает наследование через встраивание объектов.

Процедуры и модули VBA

Модуль — это совокупность объявлений (описательная часть) и процедур, хранящихся как единое целое.

Процедура — любая совокупность кода *VBA*, рассматриваемая как единое целое. Любая процедура состоит из определенного набора операторов в зависимости от задачи, которую выполняет данный код. Каждая процедура должна иметь уникальное имя (идентификатор). Для создания модулей и процедур используется окно редактора *VBA* (рис. 4.1). Для получения возможности ввести код необходимо выполнить три действия:

1) перейти в окно редактора *VBA;*
2) в окне редактора кода создать модуль (*Insert — Module*);
3) в окне модуля создать конструкцию процедуры — написать слово *Sub*, затем ввести имя и скобки.

Sub (от слова *Subroutine*) — подпрограмма (макрос). Программа автоматически поставит оператор, закрывающий конструкцию — *End Sub*.

```
Sub Процедура1 ()
End Sub
```

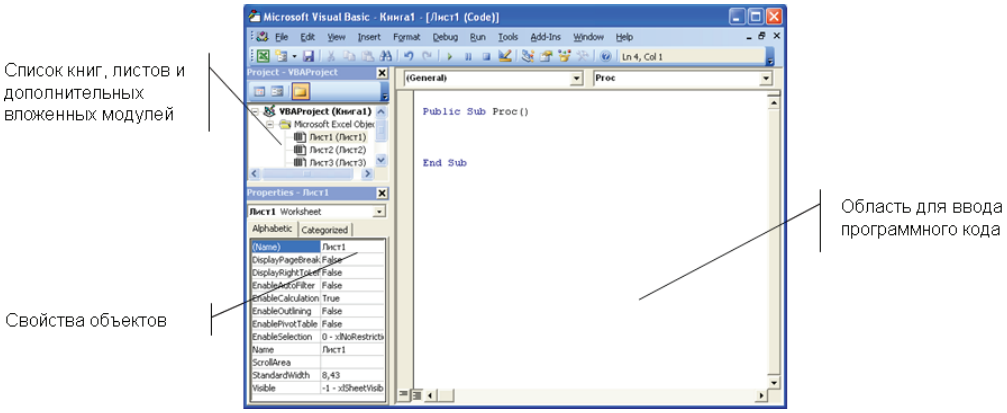


Рис. 4.1. Окно ректора *VBA*

Модули *VBA*

Для хранения кодов в *VBA* используются модули, которые хранятся в книге. Книга может содержать сколько угодно модулей. Каждый модуль в свою очередь может содержать множество процедур (макросов). Модули делятся на пять основных типов:

1) стандартный модуль;
2) модуль листа;
3) модуль книги;
4) модуль пользовательской формы;
5) модуль класса.

Стандартный модуль — это тип модулей, который используется чаще других. Данный модуль создается в окне редактора *VBA* по команде *Insert — Module*. Именно этот тип модуля используется при записи макроса с помощью макрорекордера. Стандартные модули предназначены для хранения основных процедур и открытых переменных, которые могут быть доступны из любого модуля.

Модуль листа — это тип модулей, который создается для каждого листа в рабочей книге. Модуль листа открывается в окне редактора *VBA* при выполнении команды *Исходный текст*, которая выбирается из контекстного меню, открытого на ярлычке листа. В модуле листа содержатся встроенные событийные процедуры, каждая из которых отвечает за обработку определенного события на этом листе.

Процедуру любого события, доступного для выбранного листа, можно выбрать в правом списке окна модуля (при активном объек-

те *Worksheet* в левом списке окна модуля). События возникают в следующих случаях:

— *Activate* — при активации самого листа;
— *BeforeDoubleClick* — при двойном клике мыши на любой ячейке листа;
— *BeforeRightClick* — при клике правой кнопкой мыши на любой ячейке листа;
— *Calculate* — при пересчете функций на листе;
— *Change* — при изменении значений ячеек на листе;
— *Deactivate* — при переходе с этого листа на другой лист этой же книги;
— *FollowHyperlink* — при переходе по гиперссылке, созданной на этом листе;
— *SelectionChange* — при изменении адреса выделенной ячейки/области.

Пример процедуры обработки события *Change*. Процедура имеет параметр с именем *Bt* (тип — *Range*). При выполнении процедуры на экране появляется окно сообщения, в котором присутствует текст *Адрес измененной ячейки* и сам адрес, полученный из диапазона *Tat*.

```
Sub Worksheet_Change (ByVal Bt As Range)
MsgBox «Адрес измененной ячейки: « & Bt.Address
End Sub
```

Модуль книги — это тип модулей, который создается для каждой книги. Для открытия модуля книги необходимо выполнить двойной щелчок по строке *ЭтаКнига* в окне *Project Explorer* редактора *VBA*. В модуле книги аналогично модулю листа содержатся встроенные процедуры обработки событий.

Процедуру любого события, доступного для выбранной книги, можно выбрать в правом списке окна модуля (при активном объекте *Workbook* в левом списке окна модуля).

Модуль формы — это тип модулей, который создается для каждой формы (*UserForm*). Модуль формы содержит процедуры обработки событий самой формы и ее объектов. Объектами формы являются элементы управления, расположенные на форме (кнопки, текстовые поля, поля со списками и т. п.). Пользовательские формы создаются для управления обработкой больших по количеству полей и строк таблиц, позволяя одновременно контролировать действия пользователя и регулировать степень доступа к табличным данным.

4.2. Типы данных. Типы процедур. Синтаксис VBA

Таблица 4.1

Типы данных, используемые в *VBA*

Тип данных	Описание и диапазон значений
Array	Массив переменных любого встроенного типа данных
Boolean	*True* (истина) или *False* (ложь)
Byte	Положительное число от 0 до 255
Currency	Используется для денежных вычислений с фиксированным количеством десятичных знаков. От –922 337 203 685 477,5808 до 922 337 203 685 477,5807
Date	Дата и время. Диапазон дат: от 01.01.0100 г. до 31.12.9999 г. Диапазон времени: от 00:00:00 до 23:59:59
Decimal	Десятичное представление данных в целочисленной или вещественной форме
Double	Число с плавающей точкой двойной точности. Отрицательные числа: от –1,79769313486232E+308 до –4,94065645841247E–324. Положительные числа: от 4,94065645841247E–324 до 1,79769313486232E+308
Тип данных	Описание и диапазон значений
Integer	Целое число от –32 768 до 32 767
Long	Длинное целое число от –2 147 483 648 до 2 147 483 647
Object	Ссылка на объект
Single	Число с плавающей точкой обычной точности. Отрицательные числа: от –3,402823E+38 до –1,401298E–45. Положительные числа: от 1,401298E–45 до 3,402823E+38
String (переменной длины)	Длина строки от 0 до ~2 миллиардов символов
String (фиксированной длины)	Строка от 0 до ~65 000 символов
Variant	Может использоваться для хранения любого типа данных, кроме строк фиксированной длины. Диапазон зависит от фактически сохраняемых данных
Определяемый пользователем тип данных	Используется для описания сложных данных на основе базовых типов

Синтаксис *VBA*. Переменные

В качестве идентификаторов переменных может использоваться произвольная последовательность букв и цифр длиной до 255 символов (начинается с буквы). Нельзя использовать зарезервированные слова языка и имена библиотечных объектов.

Способы объявления переменных.

1) неявный — в коде используется переменная (обычно в операторе присваивания) без предварительного представления имени переменной и типа данных переменной. Программа резервирует память для хранения этой переменной. Пример: *a = 10;*

2) явный — имя и тип переменной определяются в начале кода с помощью оператора *Dim* (*Dim <имяПеременной> [As <типДанных>]*).

Преимущества явного объявления переменных:

— ускорение выполнения кода;

— уменьшение количества ошибок;

— код становится более понятным.

Пример: *Dim a as Integer: a = 10.*

Для принудительного включения явного и обязательного объявления переменных используется оператор *Option Explicit*. Оператор должен быть расположен в самом начале модуля (раздел *Declarations*).

Константы

1. Неименованные константы — это фактические значения данных определенного типа. Можно использовать без объявления, непосредственно в выражениях.
2. Именованные константы — для использования в программе должны быть предварительно объявлены с ключевым словом *Const*.
   Const <имяКонстанты> [As <типДанных>] = <значение1>
   Const intP As Integer = 100
   Const L% = 50

Операции

1. Математические (четыре арифметических, перемена знака, целочисленное деление, остаток от деления, возведение в степень).
2. Операции отношения (>, <, =, <>, ≥, ≤).

3. Логические операции (*And, Or, Xor, Not, Imp* — импликация, *Eqv* — эквивалентность).
4. Конкатенация — операция присоединения строк (&).

Оператор присваивания

Оператор присваивания предназначен для определения значений переменных, констант, свойств объектов.

Синтаксис оператора: *[Let] <имяЭлемента> = <выражение>.*

Пример: элементу массива *Card* с номером 25 присваивается значение 200 (*Card (25)=200).*

Для присваивания переменной ссылки на объект применяется инструкция *Set*. Синтаксис: *Set <объектная Переменная> = [New] <объектноеВыражение> | Nothing.*

New — ключевое слово, которое используется при создании нового экземпляра класса. *Nothing* — ключевое слово, которое позволяет освободить все системные ресурсы и ресурсы памяти, выделенные для объекта.

Пример: переменной *Rang1* присваивается ссылка на диапазон ячеек *A*1:*B*1. *Range1 = Range (“A*1*:B*1*”).*

4.3. Процедуры SUB

В *VBA* существует два типа процедур: подпрограмма *SUB* и подпрограмма-функция *Function*. Подпрограмма *SUB* может иметь параметры. В этом случае она работает именно как подпрограмма, т. е. вызывается из другой процедуры для выполнения каких-либо действий, при этом список формальных параметров меняется на список фактических параметров. Подпрограмма-функция *Function* всегда имеет параметры (аргументы) и вызывается путем присваивания имени функции какой-либо переменной.

Подпрограмма *SUB*

Синтаксис: *[Private | Public] [Static] Sub <Имя > ([параметры])*
<Операторы>
End Sub

Подпрограмма-функция *FUNCTION*

Синтаксис: *Function <Имя > ([аргументы]) [As Type]*
<Операторы>
End Function

Область видимости процедур

Если перед словом *Sub* стоит слово *Public,* то это значит, что данная процедура — открытая процедура, т. е. ее можно вызвать из любой части программы, например, из того же модуля, или из другого модуля, или из другого проекта.

Public Sub ИмяПроцедуры () или *Sub ИмяПроцедуры ()* — по умолчанию все процедуры считаются открытыми, поэтому слово *Public* можно опустить.

Если перед словом *Sub* стоит слово *Private,* то это — локальная процедура, которую можно вызвать только из того же модуля, в котором она расположена (*Private Sub ИмяПроцедуры ()).*

Static — ключевое слово перед именем процедуры, которое означает, что значения локальных переменных процедуры сохраняются в промежутках времени между вызовами этой процедуры (*Static Sub ИмяПроцедуры ()).*

Процедура *Sub* может создаваться с параметрами и без параметров. Процедуру с непустым списком параметров можно вызвать только из другой процедуры или функции, используя ее имя со списком фактических значений параметров.

Существует два варианта передачи параметров:

— по значению (*by value*);

— по ссылке (*by reference*).

Передача параметров по ссылке (*by reference*) выполняется, если присутствует ключевое слово *ByRef* или ключевое слово, означающее способ передачи параметра, отсутствует (по умолчанию). Процедура получает значение параметра из той области памяти, где данная переменная хранится. В результате выполнения процедуры значение переменной (параметра) может изменяться.

Пример вызова процедуры, которая выводит значение фактического параметра в окне сообщений:

Private Sub F1 ()
Dim A As Integer
*A = 232 * 3*

ВызываемаяПроцедура (A)' A — фактический параметр
End Sub
Sub ВызываемаяПроцедура (ByRef X1 As Integer) 'X1 — формальный параметр
MsgBox X1
End Sub

Способы вызова процедуры с параметрами

1. Для вызова процедуры указывается только имя с фактическими параметрами (*<имяПроцедуры> <списокПараметров>).*
2. Процедура вызывается с помощью оператора *Call* (*Call <имяПроцедуры> (<списокПараметров>)).*
3. Если вызываемая процедура описана как *Public*, тогда для ее вызова используется полное имя (путь): *ИмяПроекта.ИмяМодуля.ИмяПроцедуры.*

Пример. Процедура *Primer* вызывает процедуру *Prim1* и передает предварительно инициализированные переменные:

a, b, c (фактические параметры)
Sub Primer ()
Dim a As Long, b As Long, c As Long ' фактические параметры
a = 10: b = 10: c = 10
Prim1 a, b, c ' передача фактических параметров
Prim1.Print "a = " & a & "; b = " & b & "; c = " & c
End Sub
Sub Prim1 (x As Long, ByVal y As Long, ByRef z As Long)
*x = x * 2: y = y * 3: z = z * 4*
Debug.Print "x = " & x & "; y = " & y & "; z = " & z
End Sub

Результат выполнения процедур:

x = 20; y = 30; z = 40
a = 20; b = 10; c = 40.

Данный пример демонстрирует, что при передаче параметра *ByRef* или по умолчанию происходит изменение значение параметра (параметры *a* и *c*), а при передаче *ByVal* значение параметра сохраняется прежним (параметр *b*).

Процедура *Function*

Полный синтаксис объявления функции:

Function <имяФункции> [(<списокПараметров>)] [As <типФункции>]

<операторы>

<имяФункции> = <возвращаемое_значение>

[<операторы>]

End Function

В данной конструкции используются следующие обозначения:

— *<списокПараметров>* — список формальных параметров процедуры;

— *<возвращаемое_значение>* — результат, передаваемый в вызывающую программу.

В процессе работы подпрограммы-функции необходимо соблюдать выполнение следующих правил:

1) возвращаемое процедурой *Function* значение присваивается самому имени процедуры;
2) возвращаемое процедурой *Function* значение можно использовать в выражениях в программе;
3) процедура *Function* должна имеет тип, который определяет тип возвращаемого значения (по умолчанию тип *Variant*);
4) для вызова процедуры *Function* необходимо указать имя и параметры функции в правой части оператора присваивания (функция может входить в состав выражения).

Пример функции, вычисляющей объем видеопамяти при заданных параметрах (*m* — горизонтальное разрешение экрана, *n* — вертикальное разрешение экрана, *b* — глубина цвета).

Function Vidio (m As Single, n As Single, B As Single) As Single

*Vidio = (m*n*b)/8*

End Function

В *VBA* процедура *Function* вызывается точно так же, как и любая встроенная функция:

V = Vidio (x, y, z).

Процедура *Function* позволяет создавать пользовательские функции, которые попадают в категорию *Определенные пользователем* и вызываются с помощью инструмента *Мастер функций*.

Прерывание подпрограммы

В случае необходимости выполнение процедуры или функции может быть прервано досрочно с помощью инструкции прерывания *Exit Sub*. Например, необходимо организовать выход по условию для прерывания бесконечного цикла. В этом случае синтаксис объявления процедуры имеет следующий вид:

Sub <имяПроцедуры> [(<список параметров>)]
<операторы>
Exit Sub
<операторы>
End Sub

4.4. Процедуры ввода-вывода

Процедуры ввода-вывода предназначены для ввода или вывода исходных, конечных или промежуточных результатов в специальные окна диалога. Данные процедуры называют также функциями ввода-вывода.

Процедура для ввода данных в окно диалога *InputBox*

InputBox (<сообщение> [,< заголовок >] [,< значение по умолчанию >] [,< позиция по оси X >] [,<позиция по оси Y>])

Можно использовать упрощенный вариант функции *InputBox*:

InputBox (<сообщение>).

Пример вывода на экран диалога с приглашением ввести конкретное значение переменной *x*. Аргумент функции *InputBox* — это то сообщение, которое появляется в заголовке окна диалога:

Dim x As Single
x = InputBox («Введите значение x для подстановки»)

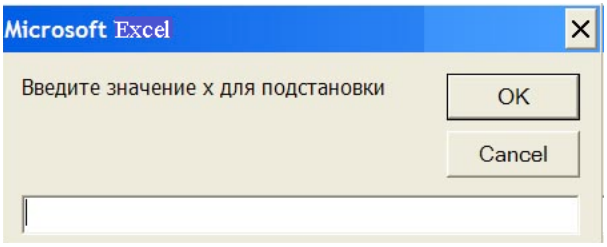


Рис. 4.2. Окно функции *InputBox*

Процедура для вывода данных в окно диалога *MsgBox*

MsgBox (< сообщение > [, < кнопки >] [,< заголовок >])

Можно использовать упрощенный вариант функции *MsgBox*:

MsgBox (< сообщение >).

Аргумент *Сообщение* — это текст, который появляется в окне диалога.

Кнопки, которые необходимо поместить в диалог, должны быть заданы с помощью второго аргумента (константы). Связь между именем константы и видом кнопки представлена в табл. 4.2. Кнопка *ОК* присутствует в диалоге по умолчанию.

Таблица 4.2

Некоторые константы и соответствующие числовые значения, связанные с командными кнопками

Константа	Кнопка	Число
vbOK	ОК	1
vbCancel	Отмена	2
vbAbort	Прервать	3
vbYes	Да	6
vbNo	Нет	7

Пример процедуры, в которой выполняется подсчет числа ячеек в заданном диапазоне:

```
Sub fff ()
Range (“a1: e5”).Select
N1 = Selection.Count
MsgBox “Область содержит “ & N1 & “ ячеек”
End Sub
```

Результат вычисления выводится в окне диалога (рис. 4.3).

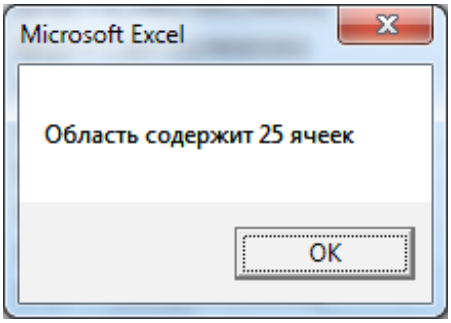


Рис. 4.3. Окно функции *MsgBox*

Пример вывода окна диалога с кнопками:

Dim nResult As Integer

nResult = MsgBox («Нажмите кнопку», vbYesNo, «Окно сообщения»)

В данном примере используются константы *vbYesNo*, связанные с кнопками «Да», «Нет» окна диалога функции *MsgBox* (рис 4.4).

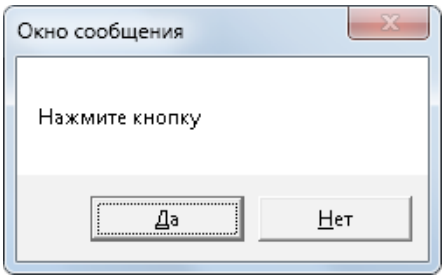


Рис. 4.4. Окно функции *MsgBox*

4.5. Управляющие конструкции VBA

Условный оператор IF

Синтаксис:

IF <условие> THEN <оператор 1> ELSE <оператор 2> End IF.

Если ветвление выполняется по трем и более направлениям, используется следующий синтаксис:

IF <условие 1> Then
<блок кода 1>
ElseIF <условие 2> Then
<блок кода 2>
ElseIF <условие 3> Then
<блок кода 3>
Else <блок кода 4>
End IF

В блоке *IF* допускается любое количество предложений *ElseIF*, но ни одно не может находиться после предложения *Else*.

Пример:

IF x = –1,57 Then
y = –1

ElseIF x = 0 Then
y = 0
ElseIF x = 1,57 Then
y = 1
Else y = Sin (x)
End IF

Оператор выбора *Select Case*

Используется для выбора одного из нескольких операторов (блоков операторов) — ветвление по трем и более направлениям.

Синтаксис:

Select Case <переменная или выражение>
Case <значение 1>
<оператор (блок операторов) 1>
Case <значение 2>
<оператор (блок операторов) 2>
Case <значение 3>
<оператор (блок операторов) 3>
End Select

Пример:

Function PR (ByVal S As Single, ByVal P As Integer) As Single
Select Case P
Case 0
*PR = S*0*
Case 1
*PR = S*0.10*
Case 2
*PR = S*0.15*
Case 3
*PR = S*0.20*
End Select
End Function

Цикл по счетчику *For … Next*

Цикл выполняется от начального до конечного значения параметра цикла с заданным шагом. Ниже представлен пример формирования и вывода массива *a (5)* на текущий лист во вторую строку (к = 2). Номер столбца определяется переменной цикла *i,* значение которой

при каждом прохождении цикла увеличивается на 1. В данном примере параметр цикла *i* меняется от 1 до 5.

Пример:

```
Sub Cycle ()
Dim a (5) As Integer
Dim i As Integer, k As Integer
K=2
For i=1 To 5
a (i) = Int (Rnd (i)*100)
Cells (k, i+1) = a (i)
Next i
End Sub
```

Цикл *For Each … Next*

Данный цикл позволяет перебрать и выполнить определенные действия со всеми элементами какой-либо коллекции (набора объектов или массива), начиная с первого элемента. В качестве коллекции могут использоваться листы книги (объект *Worksheets*), ячейки диапазона (объект *Range*), сводные таблицы книги, элементы указанного массива и т. д.

В следующем примере двойной цикл *For … Next* перебирает все элементы из заданного двумерного массива и присваивает каждому элементу значение из ячейки с адресом, соответствующим номеру элемента. Цикл *For Each* позволяет определить сумму всех элементов заполненного массива. Для работы с текущим элементом массива в цикле *For Each* используется переменная *b*.

```
Sub Перебор ()
Dim i, j As Integer
Dim x (1 To 5, 1 To 5), S As Single
Worksheets ("Лист1").Activate
For i = 1 To 5
For j = 1 To 5
x (i, j) = Cells (i, j)
Next j
Next i
For Each b In x
S = S + b
Next b
```

MsgBox (“S =”) & S
End Sub

В следующем примере в цикле *For Each* выполняется заполнение ячеек диапазона *A*20:*I*25 на листе 1. В каждую ячейку вводится число 10.

Sub Заполнение ()
For Each k In WorkSheets (“Лист1”).Range (“A20:I25”)
k.Value=10
Next k
End Sub

Цикл *With… End With*

Данная конструкция используется для упрощения записи ссылок на объект. Например, полная ссылка на объект *Range* должна содержать имя книги, имя листа, имя диапазона. Если необходимо выполнить несколько операций с данным диапазоном в текущей книге, то достаточно указать ссылку на лист, содержащий диапазон, один раз в начале конструкции, а затем использовать только имя диапазона, которое записывается после точки.

В примере, представленном ниже, выполняются четыре действия с данными, расположенными на листе с номером 2. Чтобы четыре раза не указывать ссылку на этот лист, применяется конструкция *With… End With*.

Пример:

Sub Работа_С_Ячейками ()
Dim k1 As Single, k2 As Single
With Worksheets (“Лист2”)
k1 =.Range (“A1”).Value
k2=.Range (“B1”).Value
.Range (“C1”).Value = k1 + k2
.Range (“D1”).Formula = “=A1+B1”
End With
End Sub

Ниже представлен тот же пример без использования конструкции *With*:

Sub Работа_С_Ячейками ()
Dim k1 As Single, k2 As Single
k1 = Worksheets (“Лист2”).Range (“A1”).Value

k2 = Worksheets (“Лист2”).Range (“B1”).Value
Worksheets (“Лист2”).Range (“C1”).Value = k1 + k2
Worksheets (“Лист2”).Range (“D1”).Formula = “=A1+B1”
End Sub

Таким образом, использование конструкции *With… End With* позволяет упростить запись ссылок на объекты и уменьшить количество операторов в коде.

4.6. Основные объекты VBA Excel

Наиболее часто используемыми объектами приложения *MS Excel* являются:

— *Application*;
— *Workbook* (семейство или коллекция *Workbooks*);
— *Worksheet* (семейство или коллекция *Worksheets*);
— *Range;*
— *Selection.*

Полная ссылка на объект состоит из последовательности имен вложенных друг в друга объектов. Их имена в этой последовательности разделяются точками. Последовательность начинается с объекта *Application* и заканчивается именем используемого объекта.

Пример: *Application.Workbooks (“Продажи”).Worksheets (“ПродажаК1”).Range (“A1”).*

Коллекции *Workbooks* и *Worksheets* имеют два важных метода и два важных свойства. Это метод *Add* (добавляет элемент в коллекцию) и метод *Remove* (удаляет элемент из коллекции). Свойство *Count* возвращает количество элементов в коллекции; свойство *Item* возвращает элемент коллекции.

Объект *Application*

Объект *Excel.Application* задает приложение *Excel*. Объект *Application* занимает самый верхний уровень иерархии объектов *Excel*. Данный объект управляет установками и параметрами уровня приложения, т. е. позволяет программно установить те параметры, которые можно найти в диалоговом окне *Параметры Excel*.

При использовании встроенных функций *Excel* необходимо указывать объект *Application* и аргументы функций, которые также являются объектами.

Пример: *m = Application.Max (Sheets ("Список").Range ("F5:F15"))*

Переменной *m* присваивается результат работы функции *Max,* которая определяет максимальное число в диапазоне ячеек *F*5:*F*15.

Свойства объекта *Application*

ActiveWorkbook возвращает активную (текущую) книгу.

ActiveSheet возвращает активный лист в активной рабочей книге.

ActiveCell возвращает активную ячейку на активном листе активной рабочей книги.

ThisWorkbook возвращает рабочую книгу, в которой выполняется процедура.

Selection определяет текущее выделение. Выделением может быть диапазон ячеек, элементы диаграммы и т. п.

Пример: *MsgBox Application. ThisWorkbook.Name*

Методы объекта *Application*

Calculate вызывает принудительные вычисления во всех открытых рабочих книгах.

Quit применяется для выхода из приложения *Excel.*

Run выполняет макросы, записанные в стиле *Excel.*

ConvertFormula () позволяет преобразовать формулу двумя способами:

— поменять стиль адресации ячеек;

— поменять абсолютные ссылки на относительные либо относительные на абсолютные.

Объект *Workbook*

Объект *Workbook* представляет собой файл рабочей книги.

Свойства объекта *Workbooks*

ActiveSheet возвращает активный лист книги.

ActiveChart возвращает активную диаграмму книги.

Count возвращает количество объектов семейства.

Sheets, Worksheets, Charts возвращают семейства всех рабочих листов книги и всех диаграмм соответственно.

Примеры: *kol = Application.Workbooks.Count*
MsgBox ActiveWorkbook.ActiveSheet.Name
В диалоговом окне выводит имя активного рабочего листа.

Методы объекта *Workbooks*

Метод *Activate* активизирует первый рабочий лист рабочей книги.

Метод *Add* (*Template*) создает новый объект для семейства *Workbooks*.

Аргумент *Template* задает шаблон (путь к шаблону), на основе которого создается новая рабочая книга.

Метод *NewWindow* открывает указанную книгу в новом окне.

Метод *Close* закрывает рабочую книгу.

Метод *Open FileName* открывает рабочую книгу с именем, указанным в параметре *FileName*.

Примеры:
WorkBooks ("Книга1").Activate
Workbooks.Add "c:\Мои документы\Список.xls"
Workbooks.Open "Книга1.xls"
WorkBooks ("Книга1").NewWindow
WorkBooks ("Книга1").Close

Объект *Worksheets*

Worksheet представляет рабочий лист книги и входит в семейство *Worksheets*. Ссылку на объект можно получить через команду *Worksheets* (*Index*). Команда возвращает ссылку на объект *Worksheet* по индексу в семействе. В качестве индекса может выступать имя листа или его номер в книге.

Пример:
Worksheets ("Лист1").Activate
Worksheets (1).Activate

Activesheet — возвращает ссылку на активный лист.

Пример:
Activesheet.Range ("*A*1") *=1*

Свойства коллекции *Worksheets* и объекта *Worksheet*

Свойство *ActiveCell* возвращает активную ячейку активного рабочего листа.

Свойство *Cells (<строка>, <столбец>)* возвращает ссылку на ячейку с указанными координатами (номер строки, номер столбца).

Свойство *Columns (<столбец>)* возвращает ссылку на столбец с указанным номером.

Свойство *Rows (<строка>)* возвращает ссылку на строку с указанным номером.

Свойство *Item (Index)* возвращает ссылку на лист из коллекции *Worksheets*, где *Index* — это номер или имя листа в коллекции.

Свойство *Count* возвращает количество листов в книге.

Свойство *Name* возвращает имя активного рабочего листа.

Свойство *Range (<диапазон ячеек>)* возвращает ссылку на указанный диапазон ячеек.

Примеры:

Worksheets (1).Columns (1) 'ссылка на первый лист, первую колонку

Worksheets (1).Rows (1) 'ссылка на первый лист, первую строку

kol =Workbooks ("Книга1").Sheets.Count 'в переменная kol помещается результат вычисления количества страниц в Книге1

Worksheets.Item (1).Activate 'активируется Лист1

Методы объекта *Worksheets*

Метод *Activate* активизирует рабочий лист.

Метод *Add ()* добавляет в начало рабочей книги новый лист.

Метод *Add (Before [After]:=<ссылка на лист>, [count])* добавляет в рабочую книгу новый лист перед (после) указанным листом. Необязательный параметр *Count* используется, если необходимо добавить определенное количество листов.

Метод *Delete* удаляет рабочий лист.

Метод *Copy* копирует лист в конец рабочей книги.

Метод *Copy (Before [After]:=<ссылка на лист>)* копирует лист перед (после) указанного листа.

Пример для добавления трех дополнительных листов после листа с номером 3. Затем удаляется лист с именем *Лист3:*

Dim L As Worksheet

Set L = ThisWorkbook.Worksheets.Add (after:=Лист3, Count:=3)

Worksheets ("Лист3").Delete

Объект *Range*

Объект *Range* может представлять одну ячейку, несколько ячеек (смежных или несмежных) или целый лист.

Синтаксис объекта *Range: Range (Cell1 [, Cell2]).*

Параметр *Cell1* может быть задан несколькими способами:

— именем ячейки, например, «*A*1»;

— диапазоном ячеек, например, «A1:B5»;

— двумя несмежными диапазонами ячеек, содержащими операцию объединения (запятая), например «*A*1:*B*5, *F*1:*G*8»;

— двумя несмежными диапазонами ячеек, содержащими операцию пересечения (пробел), например «A1:B5 A3:G8».

При определении диапазона могут быть заданы оба параметра *Cell1* и *Cell2*.

Пример:

Set MR1 = Range (“E1”, “E6”)

Первая ячейка определяет левую верхнюю ячейку, вторая — правую нижнюю ячейку диапазона *MR1*.

Часто при создании кода требуется создать не просто диапазон, а диапазон как объект *Range*. В этом случае необходимо выполнить следующие действия:

1) объявить переменную с типом *Range*;
2) использовать оператор *Set* для присвоения данной переменной ссылки на объект *Range*.

Примеры:

```
Dim R1 as Range
Set R1 = Range («D5») ‘ссылка на диапазон на текущем листе
Dim R2 As Range
Set R2 = Worksheets («Лист1»).Range («A1:D10») ‘ссылка на диапазон в текущей книге
Dim R3 as Range
Set R3 = Range (Cells (1, 1), Cells (5, 3))
```

Относительные ссылки объекта *Range*

Особенность работы с объектом *Range* состоит в том, что номера строк и столбцов в диапазоне, который описан как объект, привязываются к адресу первой ячейки диапазона, т. е. ссылки на ячейки внутри диапазона являются относительными.

Пример:

```
Dim M As Range
Set M = Range (“D1:D4”)
M. Range (“A1”) = 8
M. Range (“B1”) = 8
```

M. Range (“A2”) = “=D1+E1”
M. Range (“A3: A5”) = “=A3+A4”
End Sub

В данном примере создается объект *M*, заданный диапазоном *D1:D4*. Для заполнения первой ячейки данного диапазона с использованием имени объекта *M* необходимо поставить вызов *M. Range* («*A*1»). Таким образом, определяется ячейка с адресом относительно объекта *M*. Реально заполняется ячейка *D*1 (абсолютный адрес). В формулах, в правых частях последних двух операторах данного примера используются абсолютные адреса *D*1, *E*1, *A*3 и *A*4 (рис. 4.5).

	A	B	C	D	E
1				8	8
2				16	
3	2			7	
4	5			5	
5				0	

Рис. 4.5. Пример использования относительных ссылок в работе с объектом *Range*

Свойства объекта *Range*

Value — свойство, позволяющее получить значение из какой-либо ячейки диапазона или ввести значение в ячейку диапазона.

Примеры:

x = Range (‘C1’).Value ‘переменной x присваивается значение из ячейки C1.

Worksheets («Лист1»).Range («A1:A10»).Value=10 ‘каждой ячейке диапазона A1: A10 присваивается значение 10.

Range («A1»).Value = “Привет” ‘в ячейку A1 помещается слово Привет.

Cells (1, 1).Value = “Привет” ‘в ячейку A1 помещается слово Привет.

Formula — свойство, позволяющее присваивать диапазону формулы.

Range (“B2”).Formula = “=B1+2”
Range (“B3:B4”).Formula = “=A1+A2”

Когда формула присваивается диапазону ячеек, то относительные адреса формулы автоматически изменяются при переходе к очередной ячейке диапазона. В ячейке *B*4 формула будет иметь вид: =*A*2+*A*3.

Свойство *Cells (<строка>, <столбец>)* возвращает ссылку на ячейку с указанными координатами.

Пример:

Создается объект на листе 1 — переменной *s1* присваивается ссылка на диапазон *D*1.

Dim s1 As Range

Set s1= Worksheets ("Лист1").Cells (1, 4).

Свойство *ActiveCell* определяет активную ячейку.

Свойство *Row (Rows)* возвращает номер первой строки в диапазоне (ссылку на строки заданного диапазона).

Свойство *Column (Columns)* возвращает номер первой колонки в диапазоне (ссылку на колонки заданного диапазона).

Свойство *Count* определяет количество элементов в заданном диапазоне.

Пример:

Range ("A1:C3").Rows.Count

Range ("A1:C3").Columns.Count

Свойство *Item* (*Index*) возвращает ссылку на диапазон ячеек из набора ячеек листа, где *Index* — это диапазон ячеек.

Свойство *Name* возвращает имя ячейки или диапазона ячеек.

Пример: *Range ("A1:B2").Name = "Итоги"*

Методы объекта *Range*

Метод *Select* выбирает указанный диапазон ячеек, в дальнейшем к этому диапазону можно обращаться через свойство *Selection*.

Пример: *ActiveSheet.Cells (Rows.Count, 3).Select.*

На активном листе в третьем столбце (*C*) будет выделена ячейка в последней строке (1048576).

Метод *Offset* возвращает диапазон, смещенный относительно исходного диапазона на величины, заданные параметрами.

Синтаксис: *Offset (RowOffset, ColumnOffset).*

Параметры *RowOffset*, *ColumnOffset* — целые числа, определяющие, на какое количество строк и столбцов будет выполнено смещение указанной ячейки или диапазона ячеек относительно начального положения на рабочем листе.

Пример:

Range («A1»).Offset (1, 2) = Range («A1»).Value 'содержимое ячейки A1 будет помещено в ячейку C2.

ActiveCell.Offset (0, 3) = ComboBox2.Value 'содержимое элемента управления Поле со списком (текущее значение) будет помещено в ячейку, смещенную относительно текущей ячейки на три столбца.

ActiveCell.Offset (1, 2) = TextBox2.Value 'содержимое элемента управления Поле (текущее значение) будет помещено в ячейку, смещенную относительно текущей ячейки на одну строку и два столбца.

Метод *Address* возвращает в строковом формате адрес ячейки или диапазона ячеек.

Синтаксис: *Address (RowAbsolute, ColumnAbsolute, ReferenceStyle, External, RelativeTo).*

Если параметры *RowAbsolute* и *ColumnAbsolute* имеют значение *True*, то метод возвращает абсолютный адрес ячейки.

Пример:

MsgBox Cells (1,1).Address ()

MsgBox Cells (1,1).Address (rowAbsolute:=True, ColumnAbsolute: = False).

Примечание. В результате выполнения функции *MsgBox* в диалоговом окне будет выведен адрес *$A$1* в первом случае и адрес *A$1* во втором случае. По умолчанию применяется стиль ссылок *A*1. Чтобы использовать нотацию *R*1*C*1, параметру *ReferenceStyle* необходимо присвоить значение *xlR*1*C*1.

Метод *End (<параметр>)* определяет направление для расширения диапазона.

В соответствии с направлением движения по листу *Excel* в методе *End* можно использовать один из четырех параметров, который будет соответствовать нажатию одной из следующих клавиш:

1) «стрелка вверх» — *xlUp*;

2) «стрелка вниз» — *xlDown*;

3) «стрелка влево» — *xlToLeft*;

4) «стрелка вправо» — *xlToRight*.

Пример:

Range («B4»).End (xlUp).Select 'выделение последней заполненной ячейки при движении вверх по активному листу от ячейки B4.

ActiveSheet.Cells (Rows.Count, 3).End (xlUp).Select 'выделение первой заполненной ячейки третьего столбца при движении вверх от последней строки листа.

Метод *Copy* позволяет выполнять операции копирования.

Операции копирования могут выполняться несколькими способами:

1) копирование содержимого диапазона активного листа в указанный диапазон этого же листа. Пример:
   Range ("A1:A3").Copy Range ("C1:C3")
   Range ("B1").Copy Cells (Rows.Count, 1).End (xlUp).Offset (1)
   Копирование из ячейки *B*1 в первую пустую ячейку столбца 1 (если столбец не был заполнен, то значение попадет во вторую ячейку);
2) копирование содержимого диапазона листа, не являющегося активным. Пример: *Sheets ("Лист1").Range ("A1: A3").Copy Sheets ("Лист2").Range ("F1:F3")*

Метод *Cut* переместит содержимое исходного диапазона.

Пример: *Range ("A1").Cut Range ("B1")*

Метод *Sort* позволяет отсортировать содержимое ячеек диапазона.

Синтаксис: *Sort Key1:=<ячейка>, Order1:=<порядок>; Orientation:=<направление>.*

Параметры метода:

— *Key1* — ключевое поле (столбец), по которому будет производиться сортировка;
— *Order1* — указывает порядок сортировки (*xlAscending* — по алфавиту от «А» до «Я», *xlDescending* — в обратном направлении);
— *Orientation* — указывает направление (*xlSortRows* — сортировка данных в строке; *xlSortColumns* — сортировка данных в столбце).

Пример:

```
Sub bmv ()
Worksheets (1).Activate
Range ("A1: D10").Sort Key1:=Range ("B1"), Order1:=xlDescending
'сортируются значения, расположенные в столбцах диапазона A1:D10 по убыванию.
End Sub
```

AutoFit — метод автоматически настраивает ширину столбца и высоту строки.

Пример:

```
Worksheets («Лист6»).Activate
Columns.AutoFit 'настраивается ширина столбцов по введенным значениям в ячейки листа 6
```

(рис. 4.6).

	A	B	C	D
1	89	89	89	89
2	56	56	56	56
3	45	45	45	45
4	45	12	87	12
5	23	23	23	23
6	12	12	12	12
7	5	5	5	5
8	3	3	3	3
9	2	2	2	2
10	1	1	1	1

Рис. 4.6. Результат работы оператора *Columns.AutoFit*

Метод *Clear* используется для очищения диапазона ячеек. Данный метод может использоваться для удаления только содержания (*ClearContents)*, только комментариев (*ClearComments*), только форматов (*ClearFormats*) или только примечаний (*ClearNotes*).

Пример: *Range («A1:G37»).Clear.*

Метод *Insert* используется для вставки ячейки или диапазона ячеек.

Пример: *Worksheets («Лист1»).Rows (4).Insert 'вставка дополнительной строки над 4 строкой.*

Метод *AutoFill* (автозаполнение) автоматически заполняет ячейки диапазона элементами последовательности.

Синтаксис: *объект.AutoFill (<диапазон>, <тип>).*

Параметр *Тип* метода *AutoFil* имеет следующие допустимые значения:

— *xlFillDefaultш* (по умолчанию);
— *xlFillSeries* (номера);
— *xlFillFormats* (форматы);
— *xlFillValues* (значения);
— *xlFillDays* (имена дней недели);
— *xlFillMonths* (имена месяцев);
— *xlFillYears* (годы).

Пример:

```
Sub Автозаполнение ()
Range ("D1:D2").AutoFill Destination:=Range ("D1:D10"), Type:=xlFillDefault
End Sub
```

Метод *Select* используется для выделения диапазона.

Примеры:

Range ("A1: B10").Select

Range (Cells (1,1), Cells (10,2)).Select

Range («G1: K15»).Cells (3, 2).Select

В результате выполнения последнего примера будет выделена ячейка *H*3, т. к. номера строк и столбцов в диапазоне привязываются к адресу *G*1 (первой ячейки диапазона).

Объект *Selection*

Объект *Selection* создается автоматически после применения метода *Select,* т. е. объект *Selection* — это любые выделенные ячейки на рабочем листе. При работе с объектом *Selection* можно использовать свойства и методы объекта *Range*.

Пример:

Range ("A1:A10").Select 'выделяется диапазон A1:A10.

Selection.Range ("A1") = 10 'в ячейку A1 выделенного диапазона заносится число 10.

Selection.Range ("A2") = "=C1+2" 'в ячейку A2 выделенного диапазона заносится формула.

Selection.Range ("A3:A4") = "=C1+C2" 'в диапазон ячеек A3:A4 заносятся формулы.

Метод *AutoFilter*

Синтаксис: *Объект. AutoFilter (Field, Criteria1, Operator, Criteria2).*

В круглых скобках после имени метода задаются параметры фильтрации.

Параметры метода позволяют выполнить следующие действия:

— *Field* — (тип — целое), указать поле, в котором будет производиться фильтрация данных;

— *Criteria1, Criteria2* — задать критерии отбора (условия фильтрации поля);

— *Operator* — определить логическую связь между критериями отбора.

Параметр *Operator* имеет следующие допустимые значения: *xlAnd* — логическое объединение; *xlOr* — логическое сложение; *xlTop10Items* — используется для показа первых десяти (наибольших или наименьших) отфильтрованных значений.

Пример:

Sub Пример1 ()

Range («A1:C25»).Select

Selection.AutoFilter field:=3, Criteria1:=»= Екатеринбург «, Operator:=xlOr, Criteria2:=”=Москва”

End Sub

В результате выполнения данного макроса производится фильтрация списка, расположенного в диапазоне *A1:C25* — выбираются записи, в которых присутствует значение *Екатеринбург* или значение *Москва* в третьем столбце указанного диапазона.

4.7. Практические задания «Работа с объектами Application, Worksheets, Range»

Задание 1. Написать код для вычисления и вывода в ячейки листа максимального, минимального и среднего значений сумм исходного списка.

Порядок выполнения работы

1. Создайте файл с именем *Учебные процедуры*. Поместите на Лист 1 данные продаж из лабораторной работы 1. Дайте имя листу *Продажи*.
2. Перейдите на лист *Продажи* и щелкните правой кнопкой по ярлычку листа.
3. Выберите в контекстном меню команду *Исходный текст*. После выполнения команды автоматически создается модуль, связанный с текущим листом.
4. Введите следующий код в окне редактора *VBA* (диапазон сумм — это диапазон ячеек, содержащий числовые значения сумм):

Public Sub Вычисления ()

Dim mi As Integer, m As Integer, sr As Single

m = Application.Max (Sheets («имя листа»).Range («диапазон сумм»))

mi = Application.Min (Sheets («имя листа «).Range («диапазон сумм»))

sr = Application.Average (Sheets («имя листа «).Range («диапазон сумм»))

With Application.Sheets («имя листа»)

.Cells (2, 9) = «МИН=»
.Cells (2, 10) = mi
.Cells (3, 9) = «МАКС=»
.Cells (3, 10) = m
.Cells (4, 9) = «СРЕДНЕЕ=»
.Cells (4, 10) = sr
End With
End Sub

Замечание 1. Если для объекта *Range* или *Cells* не указан лист, которому принадлежит данный диапазон, то в качестве такого листа будет выступать лист, в модуле которого создан данный код, независимо от того, какой лист активный.

Замечание 2. Чтобы выполнить процедуру из рабочего листа, нажмите комбинацию клавиш <*Alt*+*F*8> и в диалоговом окне *Макрос* дважды щелкните на имени процедуры. Этот пример показывает, что для использования встроенных функций *MS Excel* необходимо указывать объект *Application* и аргументы функций, которые также являются объектами.

Задание 2. Изучить работу по созданию объекта.

Порядок выполнения работы

1. Перейдите в открытой книге на чистый лист и заполните любым текстом ячейки *A*1, *A*2, *A*3.
2. Щелкните инструмент *Visual Basic* на вкладке *Разработчик*.
3. В окне редактора создайте новый модуль, выполнив команду *Insert — Module*.
4. В окне модуля введите код процедуры:

Public Sub Просмотр ()
MsgBox Range («A1»).Text
MsgBox Range («A2»).Text
MsgBox Range («A3»).Text
End Sub

5. В том же модуле введите процедуру, выполняющую действия по созданию объектов — приложения *Word* и нового документа:

Public Sub Прил ()
Set Wor = CreateObject («Word.Application»)
'Создается объект — Приложение Word

```
Wor.Visible = True
'Свойству Visible (видимый) присваивается значение Истина
Set Dc = Wor.Documents.Add ()
'Создается объект — документ Word
Dc.Activate 'Активируется созданный документ
Wor.Selection.TypeText «Вставляемый текст»
'В активном месте документа вводится текст
End Sub
```

6. Напишите начало новой процедуры (три первых оператора) для создания объекта *Excel.Application* и объекта *Книга 1*. Используйте следующие имена переменных, которым будут присваиваться ссылки на объекты: приложение *Excel* — *Ex*, книга — *Kn*, лист — *L*.

```
Public Sub Прил1 ()
Kn.Worksheets.Add ().Name = «Список»
Set L = Kn.Worksheets («Список») 'создается объект — лист Список
Set myR = L. Range («C1: C4»)
'создается объект Range как часть листа Список
myR. Range («A1») = 8
myR. Range («B1») = 8
myR. Range («A2») = «=A3+5»
myR. Range («A3:A5») = «=A3+A4»
End Sub
```

7. Заполните ячейки *A*3:*A*5 любыми числами и проанализируйте работу формул в ячейках *C*2:*C*5.

Замечание. В левой части операторов присваивания, относящихся к объекту *myR,* адреса *A*1 и *A*2 — это ссылки на ячейки диапазона *myR*, а в правых частях формул адреса *A*3 и *A*4 — это ссылки на ячейки активного листа.

8. Сохраните книгу с именем *Список* и закройте ее.

Задание 3. Изучить работу по передаче параметров в процедуре-подпрограмме и процедуре-функции

Порядок выполнения работы

1. В окне модуля введите код процедуры.
2. Для просмотра результатов вывода с помощью оператора *Debug. Print* добавьте окно *Immediate* (*View — Immediate.Window*).

```
Sub PrimerPS ()
Dim a As Byte, b As Byte, c As Byte 'фактические параметры
a = 10: b = 10: c = 10
PS a, b, c 'вызов Sub PS и передача фактических параметров
Debug.Print «a = « & a & «; b = « & b & «; c = « & c
End Sub
Sub PS (x As Byte, ByVal y As Byte, ByRef z As Byte)
x = x * 2: y = y * 3: z = z * 4
Debug.Print «x = « & x & «; y = « & y & «; z = « & z
End Sub
```

3. Создайте процедуру-функцию для вычисления значения объема видеопамяти при заданных параметрах (*m* — горизонтальное разрешение экрана, *n* — вертикальное разрешение экрана, *b* — глубина цвета).

```
Function Vidio (m As Single, n As Single, B As Single) As Single
Vidio = (m*n*b)/8
End Function
```

4. Вызовите процедуру-функцию из процедуры *PrimerPS*, задав фактические параметры. Используйте функцию *MsgBox* для вывода результата.
5. Выделите любую ячейку Листа 2 текущей книги. Вызовите *Мастер функций*, выберите категорию *Определенные пользователем* и выполните вычисления с помощью функции *Vidio*.

Задание 4. Изучить работу управляющих конструкций *VBA*.

Порядок выполнения работы

1. В окне модуля введите код процедуры:

```
Sub ПроверкаРаботыЦиклов ()
Dim I as Integer, K as Single
    Do                          'Бесконечный цикл
        For i = 1 To 100        'Цикл выполняется 100 раз
          K = Int (Rnd * 10)    'Генерирует случайное число
          Select Case K         'Анализирует случайное число
            Case 9: Exit For    'Если 9, завершает цикл For… Next
            Case 2: Exit Do     'Если 2, завершает цикл Do… Loop
            Case 5: Exit Sub    'Если 5, завершает процедуру Sub
        End Select
```

```
        Exit For
    Next I
    Exit Do
Loop
Exit Sub
End Sub
```

2. Запустите выполнение процедуры по шагам с помощью управляющей клавиши *F*8. Последовательно нажимайте клавишу и следите за состоянием переменных в окне *Locals* (*View — Locals Window*).

Задание 5. Изучить работу объекта *Range.*

Порядок выполнения работы

1. В окне редактора *VBA* введите коды нижерасположенных процедур.
2. Введите комментарий к каждой процедуре, расположив его перед оператором *End Sub*:

```
Public Sub ФункцииВв ()
Dim N As Integer
N = MsgBox («Нажмите кнопку», vbYesNo, «Это заголовок окна»)
ThisWorkbook.Worksheets (1).Range («A1»).Value = «Вы нажали кнопку « & N
ThisWorkbook.Worksheets (1).Range («A1»).Columns.AutoFit
End Sub

Sub Пример1 ()
Worksheets (1).Activate
Range («a1») = 6
Range («a2») = 7
Range («a3») = 8
Range («a4») = 9
Range («a1: d4»).FillRight
End Sub

Sub Пример2 ()
Worksheets (1).Activate
Range («d1») = 1
```

```
Range («d2») = 2
Range («d3») = 3
Range («d4») = 4
Range («a1: d4»).FillLeft
End Sub

Sub Пример3 ()
Worksheets (1).Activate
Range («a1») = 1
Range («a1: b10»).FillDown
End Sub
Sub Prima3 ()
Worksheets (2).Activate
For I = 1 To 10
Cells (I, 1) = I
Next I
Range («b1»).Formula = «=sin (a1)»
Range («b1»).AutoFill Range («b1: b10»)
End Sub

Sub Пример4 ()
With Worksheets («Лист3»)
For I = 1 To 3
For j = 1 To 3
.Cells (I, j) = Int (Rnd (I * j) * 100)
Next j
Next I
.Range («D1: F3»).FormulaArray = «=MINVERSE (a1: c3)»
End With
End Sub

Sub Пример5 ()
With Worksheets («Лист4»)
For I = 1 To 3
For j = 1 To 3
.Cells (I, j) = Int (Rnd (I * j) * 100)
Next j
Next I
```

```
.Range («D1: F3»).FormulaR1C1 = «=MINVERSE (R1C1: R3C3)»
End With
x = Range («c1»).Value
Range («a11: b14»).Value = 12
MsgBox x
End Sub
```

4.8. Практические задания «Работа с объектом Worksheets»

Задание 1. Изучить работу объекта *Worksheets*.

Порядок выполнения работы

1. Откройте книгу *Маска*.
2. Перейдите в открытой книге на чистый лист и щелкните инструмент *Visual Basic* на вкладке *Разработчик*.
3. В окне редактора создайте новый модуль, выполнив команду *Insert — Module*.
4. В окне модуля введите код процедуры. Код имеет три смысловых части. Сделайте комментарий к каждой части.

```
Sub Study ()
MsgBox Application.ThisWorkbook.Name
'**********************
Dim oSheet As Worksheet
Set oSheet = ThisWorkbook.Worksheets.Add ()
oSheet.Name = «Список1»
'**********************
Dim k1 as Byte
Worksheets (3).Activate
k1 = ThisWorkbook.Sheets.Count
Range («a1») = k1
MsgBox Cells (1, 1).Address (ReferenceStyle:=xlR1C1)
End Sub
```

5. Внесите изменения в код — добавьте три листа после Листа 3, не давая имен новым листам, используйте метод *Add* с параметрами (*after*:=*Лист3*, *Count*:=*3*).

6. Введите следующую процедуру для просмотра имен и соответствующих индексов листов текущей книги.

Sub УдалениеЛистов ()
Dim L As Worksheet
For Each L In ThisWorkbook.Worksheets
MsgBox «имя листа» & L. Name & « « & «номер листа» & L. Index
Next
End Sub

7. Внесите изменения в код процедуры для удаления трех последних добавленных листов. Используйте условный оператор и метод *Delete*.

Замечание. Для определения удаляемых листов используйте свойство *Index* или *Name*.

Задание 2. Работа с функцией *ВПР*.

Порядок выполнения работы

1. Перейдите в открытой книге на лист *Список* и добавьте пустой столбец перед полем *Наименование*. Дайте имя новому полю списка — *Артикль*.
2. Выделите диапазон списка и присвойте ему имя *Список*.
3. Создайте процедуру для заполнения столбца целыми случайными числами в диапазоне от 10000 до 20000 с помощью функции *СЛЧИС*.

Sub ЗаполнениеДиапазона ()
Dim d As Range
Set d = Worksheets («Список»).Range («a2: a940»)
*d.FormulaLocal = «=Целое (СЛЧИС ()*10000+10000)»*
d.Copy
‘диапазон a2: a940 листа Список копируется в буфер обмена (для копирования в другой ‘диапазон необходимо последний указать после метода Copy)
d.Select
Selection.PasteSpecial Paste:=xlPasteValues ‘вставляются только значения
End Sub

4. Напишите процедуру для копирования данных продаж с листа *Список* на лист *Список1*.

5. Введите в ячейки *L*1:*N*1 листа *Список1* заголовки *Артикль, Наименование, Сумма*.
6. Последовательно скопируйте любые три артикля и вставьте в ячейки *L*2:*L*4.
7. В ячейку *M*2 введите формулу — (функцию *ВПР*) — для поиска значения артикля и вывода соответствующего наименования продукта. Скопируйте формулу в диапазон *M*3:*M*4. Аналогично заполните столбец *N*.

Замечание. Если функция *ВПР* возвращает ошибку #Н/Д, отсортируйте поле *Артикль* по возрастанию.

8. Выполните программно поиск артикля и вывод соответствующего наименования продукта.
   1) создайте на листе массив значений артиклей, заполнив ячейки *L*8:*L*10;
   2) напишите макрос, используя команду *With* и оператор цикла *For;*
   3) организуйте внешний и внутренний циклы. Переменная *i* внешнего цикла меняется от 2 до 940, переменная *j* внутреннего цикла — от 8 до 10;
   4) во внутреннем цикле используйте условный оператор *IF* для проверки совпадения значений артиклей из списка и из массива. В случае совпадения выведите в диапазон ячеек *M*8:*M*10 названия продуктов, соответствующих артиклям из массива.
9. Используйте функцию *ВПР* для заполнения нового столбца списка с именем *Премия*. Формула введет значение премии в зависимости от значения суммы. Далее:
   1) создайте дополнительную таблицу, введя данные в диапазон *L*13: *M*18 (рис. 4.7). Поиск будет выполняться в первом столбце, а выбор соответствующего найденному значения будет выполняться из второго столбца таблицы;
   2) добавьте имя *Премия* в строку имен списка;
   3) выделите любую ячейку списка и выполните команду *Форматировать как таблицу*;
   4) введите формулу (функцию *ВПР*) в ячейку *J*2. При заполнении диалога функции введите число 1 в поле четвертого аргумента (для поиска ближайшего наименьшего значения).

Таблица	Размер премии
-1000	0
30000	5000
50000	10000
70000	15000
90000	20000

Рис. 4.7. Данные дополнительной таблицы

Задание 3. Работа с текущей ячейкой.

Порядок выполнения работы

1. В окне редактора создайте новый модуль, выполнив команду *Insert — Module*.
2. В окне свойств модуля введите новое имя *ActiveCell*.
3. В окне модуля введите код процедуры. Код позволяет выполнять удаление строки, соответствующей текущей ячейке.

```
Sub УдалениеСтроки1 ()
With ActiveSheet
.Rows (ActiveCell.Row & «:» & ActiveCell.Row).Select
Selection.Delete shift:=xlUp
End With
End Sub
```

4. Дополните макрос так, чтобы перед удалением строки выводился адрес выделенной ячейки:

— с помощью функции *MsgBox* и свойств *ActiveCell.Row* и *ActiveCell.Column*;

— с помощью функции *MsgBox* и свойства *Address*.

5. Введите следующую процедуру для выделения диапазона, включающего текущую ячейку:

```
Sub ВыдДиап ()
Range (ActiveCell, ActiveCell.End (xlDown)).Select
End Sub
```

6. Запустите процедуру два раза, предварительно выполнив следующие действия:
   — перед первым запуском выделите любую ячейку;
   — перед вторым запуском заполните числами несколько ячеек любого столбца и выделите первую ячейку.

7. Напишите аналогичную процедуру для выделения диапазона, расположенного в любой строке. Используйте параметр *xlToRight*.
8. С помощью макроса удалите содержимое текущей ячейки. Используйте метод *Clear*.
9. С помощью макроса выполните следующие действия:
   — добавьте в книгу пустой лист после листа, содержащего список *Продажи*;
   — скопируйте на новый лист заголовки списка *Продажи*, считая, что список начинается с *A*1, а количество полей в списке неизвестно;
   — активизируйте новый лист;
   — настройте ширину заполненных колонок по содержимому ячеек (используйте метод *AutoFit*);
   — заполните поле *Артикль* порядковыми номерами (10 ячеек);
   — сделайте активной ячейку, предназначенную для ввода первого значения второго поля.

Раздел 5.
Создание пользовательской формы

Форма — это объект, который содержит некоторый набор элементов управления (ЭУ), позволяющих выполнять определенные действия с табличными данными. К элементам управления относятся такие элементы как флажок, список, поле со списком, кнопка, текстовое поле, надпись и др.

Наличие на листе рабочей книги пользовательской формы с определенным набором элементов управления предоставляет дополнительные возможности по обработке данных, такие как проверка вводимых значений в табличную базу данных, поиск нужных значений, быстрое удаление и обновление записей табличной базы.

В программе можно автоматически создать готовую форму для работы с существующим списком (табличной базой) по команде *Форма*. Для добавления команды на ленту окна программы необходимо выполнить команду *Файл — Параметры — Настройка ленты*, выбрать из списка *Команды не на ленте*, создать новую вкладку и добавить на нее команду *Форма*.

Команда *Форма* вызывается для работы с готовым списком, расположенным на активном листе рабочей книги. После вызова команды на экране появляется готовая форма (рис. 5.1).

Форма позволяет:

— просматривать записи;

— быстро перемещаться по списку;

— редактировать записи;

— добавлять новые записи.

Элементы управления формы

Для выполнения действий, связанных с элементами управления формы необходимо в окне программы перейти на вкладку *Разработ-*

чик и выполнить команду *Вставить*. На появившейся панели будут представлены две группы элементов управления (рис. 5.2). Первая группа — это элементы управления формы, вторая группа — элементы *ActiveX*.

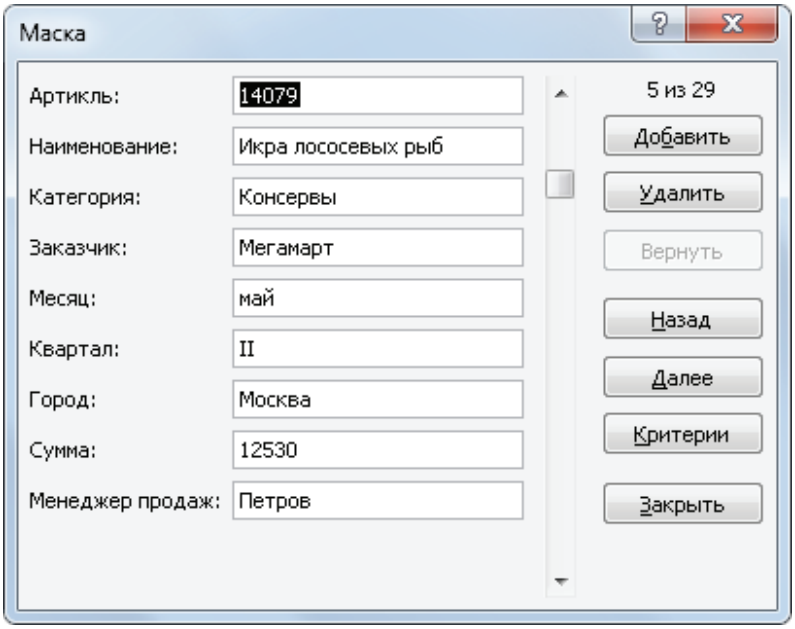


Рис. 5.1. Автоматическая форма *Продажи*

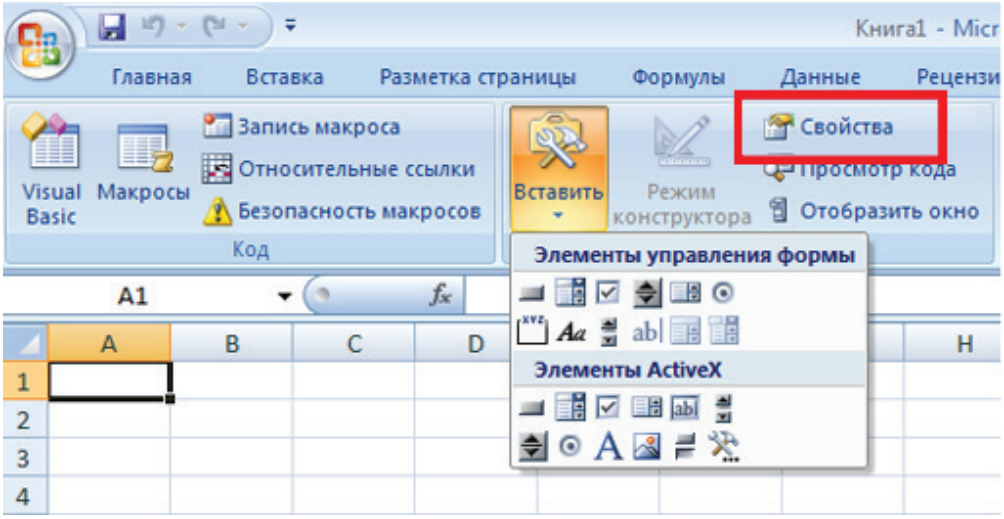


Рис. 5.2. Выбор элемента управления на ленте *Разработчик*

Элементы управления формы позволяют организовать интерфейс пользователя без использования программирования, а элементы группы *ActiveX* ориентированы на использование в процедурах *VBA*.

Наиболее часто в формах используются такие элементы, как *Label* (надпись — для отображения и ввода текста), *TextBox* (поле — для отображения и ввода строковых и числовых данных), *ComboBox* (поле со списком — для отображения списка значений), *CheckBox* (флажок — для отображения данных логического типа), *CommandButton* (кнопка — для выполнения любых действий, прописанных в коде обработки события).

Любой элемент управления имеет набор свойств, которые можно устанавливать вручную или программно. Рассмотрим пример установки свойств элемента управления *ComboBox*.

Для выполнения настройки необходимо выполнить команду *Свойства* и в диалоге *Формат элемента управления* установить следующие свойства:

— цвет;

— границы;

— шрифт;

— защита;

— формировать список по диапазону *B*2:*B*6;

— связь с ячейкой — *F*2.

Свойство *Формировать список по диапазону* позволяет задать тот набор значений, который будет открываться по нажатию кнопки со стрелкой. Для примера, представленного на рис. 5.3, это диапазон, относящийся к полю *Наименование*.

Свойство *Связь с ячейкой* позволяет задать адрес ячейки текущего листа, в которую будет выводиться номер значения, выбранного из списка данного элемента управления.

F3 =ИНДЕКС(A2:D8;F2;4)

	A	B	C	D	E	F	G	H
1	Артикль	Наименование	Категория	Сумма				
2	10025	Творог	Винно-водочные	68523		4	Мандарины	
3	10055	Крабовые палочки	Напитки	25984		32587		
4	10078	Ячмень	Консервы	21354				
5	10100	Мандарины	Фрукты	32587				
6	10134	Минтай	Морепродукты	96541				

Рис. 5.3. Работа с ЭУ *Поле со списком*

5.1. Создание формы в редакторе VBA

В окне редактора *VBA* для создания формы необходимо выполнить ряд действий.

1. Нажать кнопку на панели инструментов *Insert UserForm*.

Откроется окно дизайнера форм. По умолчанию форма будет называться *UserForm1* (или *UserForm2* — в зависимости от количества уже созданных форм в данной книге). Автоматически откроется панель с элементами управления — *Toolbox* (рис. 5.4).

2. Поместить в форму необходимые элементы управления (количество и положение элементов зависит от конкретной задачи, для которой создается форма).
3. Написать коды процедур обработки событий:

— инициализация формы;
— изменение значения списка;
— нажатие кнопки и др.

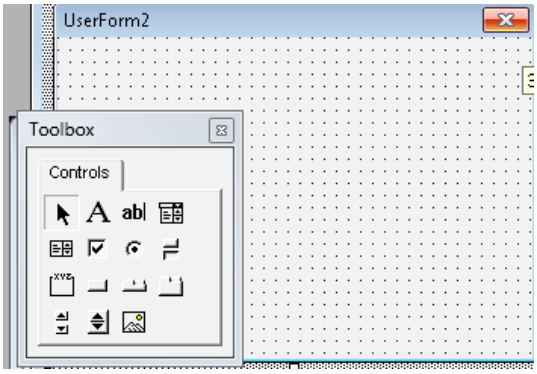


Рис. 5.4. Новая форма в окне редактора *VBA*

Свойства формы

После создания формы необходимо установить свойства формы. Установка свойств выполняется в окне свойств, которое открывается по команде *View — Properties Window*. Рассмотрим наиболее часто используемые свойства формы.

1. *Name* — определяет имя формы. Имя формы используется в программном коде по умолчанию (*UserForm*).
2. *Caption* — определяет заголовок формы (по умолчанию совпадает с именем формы).

3. *Enabled* — свойство имеет два значения: *True* (по умолчанию) и *False*. Значение *False* используется для временного отключения формы.

Методы форм

Метод *UserForm1.Show* выполняет следующие действия:

— если форма уже была загружена в память, она становится видимой;

— если форма не загружена в память, она будет автоматически загружена (произойдет событие *Load*).

Метод *UserForm1.Hide* уберет форму с экрана, но оставит в памяти. Затем при помощи метода *Show* () форму можно вызвать в том же состоянии, в каком она была на момент сокрытия.

Метод *Unload UserForm1* позволяет удалить форму из памяти с помощью команды *Unload*.

События форм

Событие *Initialize* происходит при подготовке формы к открытию. Код обработки данного события содержит операторы, связанные с соединением с базой данных, настройкой элементов управления в форме, присвоением элементам управления значений по умолчанию и т. п.

Событие *Click* выбирается по умолчанию как реакция на одиночный или двойной щелчок мыши.

Событие *Error* используется при возникновении ошибки в форме.

Элемент управления *TextBox*

Текстовое поле (*TextBox*) используется для выполнения следующих действий:

1) приема каких-либо данных, вводимых пользователем;
2) вывода пользователю данных с возможностью их редактирования;
3) вывода пользователю данных с возможностью копирования и печати, но без возможности изменения.

Свойства элемента управления *TextBox*

Свойство *Value* используется для занесения исходного значения и для приема значения, введенного пользователем.

Свойство *AutoSize* дает возможность для поля автоматически менять свой размер, подстраиваясь под размер данных.

Свойство *ControlSource* дает ссылку на источник данных для поля. При изменении пользователем данных в поле со списком автоматически изменится значение в источнике, определенном в свойстве *ControlSource*.

С помощью свойства *Locked* пользователь сможет выделять и копировать данные из поля, но не сможет их редактировать.

Свойство *MaxLength* — это возможность установить максимальную длину для значений, которые вводятся в данное поле.

Элемент *ActiveX ComboBox*

Поле со списком (комбинированный список) *ComboBox* используется в двух случаях:

1) пользователю необходимо предоставить возможность выбрать одно или несколько значений из списка;
2) список позиций для выбора необходимо формировать динамически на основании данных из источника (базы данных, листа *Excel* и т. п.).

Базовые свойства и методы элемента управления *ComboBox*

ColumnCount позволяет задать количество столбцов в списке.

ColumnWidth позволяет задать ширину столбцов.

ColumnHeads определяет, отображать (значение *True*) или не отображать (значение *False*) заголовки столбцов.

RowSource позволяет задать диапазон для элементов списка.

Value (*Text*) возвращает текущее значение выбранного элемента списка.

Метод *AddItem* используется для заполнения *ComboBox* в коде обработки события *Initialize* для формы, содержащей данный элемент управления.

Пример:

```
Private Sub UserForm_Initialize ()
ComboBox1.AddItem “Екатеринбург”
ComboBox1.AddItem “Свердловская область”
ComboBox1.AddItem “Москва”
ComboBox1.AddItem “Московская область”
End Sub
```

Главное событие ЭУ *TextBox и ComboBox*

Событие *Change* определяет изменение содержимого поля (выбор элемента списка).

Обычно к данному событию привязывается проверка вводимых пользователем значений или синхронизация выбранного значения с другими элементами управления (например, сделать доступной кнопку, изменить текст надписи и т. п.)

Пример:

```
Private Sub ComboBox1_Change ()
With ActiveSheet
ComboBox2.RowSource = Range (Cells (2, 1), _
Cells (Cells (Rows.Count, 1).End (xlUp).Row, 1)).Address ()
End With
End Sub
```

5.2. Практические задания «Использование VBA для разработки форм различной степени сложности»

Задание 1. Изучить работу элементов управления.

Порядок выполнения работы

1. Создайте новую книгу *Формы*.
2. Переименуйте Лист 1 в лист *Форма1*.
3. Скопируйте на второй лист список *Продажи*. Переименуйте лист, используя имя *Продажи*.
4. Перейдите на вкладку *Разработчик* и нажмите кнопку *Вставить*.
5. Выберите в разделе *Элементы ActiveX* инструмент *Кнопка*.
6. На листе *Форма1* нарисуйте кнопку (*ActiveXButton).*

Замечание. Редактирование кнопки выполняется в режиме конструктора (инструмент *Конструктор* выбран на вкладке *Разработчик*). Для возможности использования кнопки по назначению (вызов формы) режим *Конструктор* необходимо деактивировать.

7. Откройте контекстное меню на кнопке и выберите команду *Свойства*. Установите следующие свойства элемента:

— *Caption* — введите текст *Вызов формы*;
— *Back Color* — откройте список и выберите на вкладке *Palette* любой цвет;
— *Font* — выберите размер шрифта 12 пт и полужирное начертание.

8. Закройте окно свойств и выберите из контекстного меню кнопки команду *Исходный текст*.
9. В окне редактора *VBA* введите в процедуру кнопки оператор *UserForm1.Show:*

```
Private Sub CommandButton1_Click ()
UserForm1.Show
End Sub
```

10. Создайте новый модуль, выполнив команду *Insert — UserForm*.
11. Активизируйте форму и заполните свойства формы (для отображения окна свойств выполните команду *View — Properties Window*):
— *Caption* — введите текст *Продажи*;
— *Back Color* — выберите вкладку *Palette* и цвет;
— *Font* — выберите размер шрифта 12 пт и полужирное начертание.
12. Щелкните 2 раза по форме и внесите изменения в код:
— удалите процедуру *Private Sub UserForm1_Click ();*
— введите процедуру, представленную ниже.

```
Private Sub UserForm_Initialize ()
With UserForm1
.Height = 200
.Width = 600
End With
End Sub
```

13. Перейдите на лист *Форма1* и деактивируйте режим *Конструктор*.
14. Проверьте работу кнопки — нажатие кнопки должно приводить к появлению на экране формы.
15. Закройте форму и перейдите в окно редактора *VBA*.
16. Используя инструменты *Label* и *ComboBox* панели *ToolBox*, поместите на форму надпись и поле со списком. Присвойте надписи имя *Листы*.
17. Отредактируйте код инициализации формы. Поместите следующий фрагмент кода после оператора *End With:*

```
Dim N As Integer
For N = 1 To Worksheets.Count
With ComboBox1
.AddItem Worksheets (N).Name
End With
Next N
```

18. Перейдите на лист *Форма1* и откройте форму для просмотра. Удостоверьтесь, что в списке *ComboBox1* присутствуют имена всех листов книги.
19. Закройте форму и перейдите в окно редактора *VBA*.
20. Выполните дойной щелчок на элементе *ComboBox1* и введите код в открывшуюся процедуру:

```
Private Sub ComboBox1_Change ()
Dim S As String
S = ComboBox1.Value
If ComboBox1.Value <> "" And ComboBox1.Value <> "Форма" Then
Sheets (S).Activate
'активируется лист, имя которого выбрано в списке формы
End If
End Sub
```

21. Проверьте работу формы.
22. В окне редактора поместите 4 элемента *Надпись* в форму, расположив их в одну линию ниже поля со списком.
23. Используйте следующие свойства для форматирования надписей:
    — *Height* — 12;
    — *Width* — 100;
    — *Font* — полужирный, 10 пт;
    — *TextAlign* — 2 (по центру).
24. Добавьте в форму кнопку *CommandButton1*. Установите следующие свойства:
    — *Font* — полужирный, 10 пт;
    — *Caption* — слово Ввести;
    — *BackColor* — цвет Синий;
    — *ForeColor* — цвет Белый.
25. Выполните двойной щелчок на элементе *CommandButton1* и введите код в открывшуюся процедуру:

```
Private Sub CommandButton1_Click ()
Label2 = ActiveSheet. [a1].Value
```

Label3 = ActiveSheet. [b1].Value
Label4 = ActiveSheet. [c1].Value
Label5 = ActiveSheet. [d1].Value
End Sub

26. Проверьте работу кода.
27. Поместите в форму еще одно поле со списком (*ComboBox2*) и три поля ввода: *TextBox1*, *TextBox2*, *TextBox3*. Все четыре элемента управления должны быть расположены в одну линию под надписями.
28. Используйте следующие свойства для форматирования полей:
    — *Height* — 20;
    — *Width* — 100;
    — *Font* — полужирный, 10 пт.
29. Допишите код процедуры *ComboBox1*. Вставьте фрагмент кода после оператора *End If*. Фрагмент содержит код для формирования диапазона листа как источника данных второго поля со списком.

With ActiveSheet
ComboBox2.RowSource=Range (Cells (2,1), Cells (Cells (Rows.Count,1). End (xlUp).
Row,1)).Address ()'продолжение предыдущей строки
End With

Замечание. Диапазон указывается от ячейки *Cells* (2,1) до ячейки с адресом, который вычисляется и возвращается свойством *Address*. Вычисление выполняется с помощью свойства *Count* (определяет количество рядов в диапазоне) и метода *End* с параметром *xlUp* (определяет направление для расширения диапазона — на сколько рядов нужно подняться вверх от вычисленной ячейки).

30. Выполните дойной щелчок на элементе *ComboBox2* и введите код в открывшуюся процедуру:

Private Sub ComboBox2_Change ()
Dim Nr As Long, Nc As Long
TextBox1 = ""
TextBox2 = ""
TextBox3 = ""
If ComboBox2 <> "" Then
Dim Z As Variant
Z = ComboBox2.Value

```
ActiveSheet.Range (Cells (2,1), Cells (Cells (Rows.Count,1).End (xlUp).
Row,1)).
Find (what:=Z, lookat:=xlWhole).Activate
‘ две предыдущие строки вводятся в одну линию
Nr = ActiveCell.Row
Nc = ActiveCell.Column
TextBox1.Value = ActiveSheet.Cells (Nr, Nc + 1).Value
TextBox2.Value = ActiveSheet.Cells (Nr, Nc + 2).Value
TextBox3.Value = ActiveSheet.Cells (Nr, Nc + 3).Value
End If
End Sub
```

Замечание. В переменную *Z* помещается значение артикля, выбранное из списка, по этому значению ведется поиск в первом столбце исходного списка. Найденная ячейка активируется. В переменные *Nr* и *Nc* помещаются номер строки и номер столбца соответственно. В дальнейшем номер столбца увеличивается на 1 для определения каждого следующего значения поля исходного списка (в строке с номером *Nr*).

Задание для самостоятельной работы

1. Допишите код, связанный с элементом управления *ComboBox2*, для выделения в исходном списке записи, значения полей которой появились в форме.
2. Поместите в форму кнопку с именем *Удаление записи*.
3. При нажатии кнопки *Удаление записи* в исходном списке должна быть удалена запись, представленная в форме.

Замечание. Используйте метод *Delete* с параметром *shift:=xlUp* (сдвиг строк вверх).

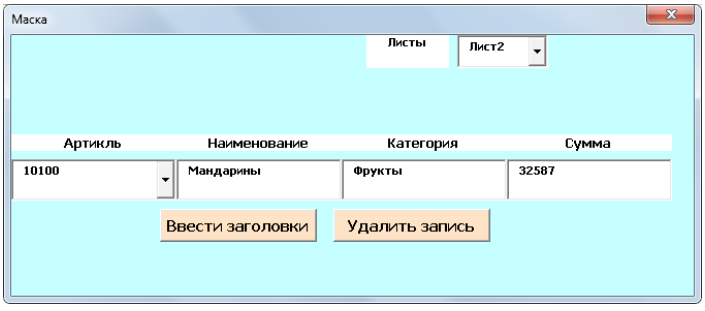


Рис. 5.5. Форма *Продажи*

Библиографический список

1. Агальцов В. П. Информатика для экономистов: учеб. для студентов экон. специальностей вузов / В. П. Агальцов. — М. : ФОРУМ: ИНФРА-М, 2007.
2. Гарнаев А. Excel, VBA, Internet в экономике и финансах / А. Гарнаев. — СПб. : БХВ-Петербург, 2005.
3. Гарнаев А. VBA / А. Гарнаев. — СПб. : БХВ-Петербург, 2005.
4. Информатика : учеб. для студентов экон. специальностей вузов / под ред. Н. В. Макаровой. — М. : Финансы и статистика, 2012.
5. Ивасенко А. Г. Информационные технологии в экономике : учеб. пособие / А. Г. Ивасенко, А. Ю. Гридасов, В. А. Павленко. — М. : КНОРУС, 2008.
6. Олбрайт К. Моделирование с помощью Microsoft Excel и VBA. Разработка систем поддержки принятия решений / К. Олбрайт. — М.; СПб. : Киев : Вильямс, 2005.
7. Роман С. Использование макросов в Excel / С. Роман. — 2-е изд. — СПб. : Питер, 2004.


ИЗДАТЕЛЬСТВО УНИВЕРСИТЕТСКОЙ КНИГИ
2015


9 785799 619497