

ВИКТОР ПЕСТРИКОВ, АНДРЕЙ ТЯЖЕВ

# QBASIC

НА ПРИМЕРАХ

**ОСНОВНЫЕ ОПЕРАТОРЫ  
И ФУНКЦИИ**

**МЕТОДИКА НАПИСАНИЯ  
И ОТЛАДКИ ПРОГРАММ**

**ГРАФИЧЕСКИЕ  
ВОЗМОЖНОСТИ QBASIC**

**БОЛЕЕ 130 ПРИМЕРОВ  
ПРОГРАММ**

**Виктор Пестриков  
Андрей Тяжев**

# **QBASIC** **НА ПРИМЕРАХ**

Санкт-Петербург  
«БХВ-Петербург»

2010

УДК 681.3.068+800.92Qbasic  
ББК 32.973.26-018.1  
П28

**Пестриков, В. М.**

П28 QBASIC на примерах / В. М. Пестриков, А. Т. Тяжев. — СПб.: БХВ-Петербург, 2010. — 304 с.: ил.

ISBN 978-5-9775-0466-9

На многочисленных примерах рассмотрены вопросы программирования на языке QBASIC. Приведено описание основных конструкций алгоритмического языка и показано их использование при решении типовых задач. Для наглядности структуры алгоритма примеры сопровождаются блок-схемами, тексты программ — комментариями. Для закрепления материала подробно рассмотрена разработка программ для игр и создания музыкальных произведений.

*Для начинающих программистов*

УДК 681.3.068+800.92Qbasic  
ББК 32.973.26-018.1

**Группа подготовки издания:**

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Нина Седых</i>
Компьютерная верстка	<i>Ольги Сергиенко</i>
Корректор	<i>Зинаида Дмитриева</i>
Дизайн серии	<i>Инны Тачиной</i>
Оформление обложки	<i>Елены Беляевой</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 31.08.09.

Формат 70×100<sup>1/16</sup>. Печать офсетная. Усл. печ. л. 24,51.

Тираж 1000 экз. Заказ №

"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Санитарно-эпидемиологическое заключение на продукцию № 77.99.60.953.Д.005770.05.09 от 26.05.2009 г. выдано Федеральной службой по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов  
в ГУП "Типография "Наука"  
199034, Санкт-Петербург, 9 линия, 12

# Оглавление

<b>Предисловие .....</b>	<b>1</b>
<b>Введение .....</b>	<b>3</b>
<b>Глава 1. Создание программы.....</b>	<b>7</b>
1.1. Разработка программ.....	7
Пример 1.1. Экономическая задача .....	9
Пример 1.2. Олимпийские кольца.....	13
Пример 1.3. Оптимизационная задача "Диета" .....	14
Пример 1.4. Гамма .....	17
Пример 1.5. Таблица умножения .....	22
Пример 1.6. Таблица значений функции.....	23
Пример 1.7. Погружение в Бейсик.....	26
1.2. Разработка блок-схемы .....	30
1.2.1. Разработка машинно-ориентированного алгоритма .....	30
Пример 1.8. Алгоритмы экономической задачи.....	32
Пример 1.9. Диета .....	36
<b>Глава 2. Трансляция и отладка программы.....</b>	<b>41</b>
2.1. Работа со средой QBASIC.....	41
2.2. Пункты меню .....	50
2.2.1. Меню <i>Файл (File)</i> .....	50
Пункт меню <i>Новый (New)</i> .....	50
Пункт меню <i>Открыть (Open)</i> .....	51
Пункт меню <i>Сохранить (Save)</i> .....	51
Пункт меню <i>Сохранить как (Save As)</i> .....	51
Пункт меню <i>Печатать (Print)</i> .....	51
Пункт меню <i>Выход (Exit)</i> .....	52
2.2.2. Меню <i>Редактирование (Edit)</i> .....	53
Пункт меню <i>Отменить (Undo)</i> .....	53
Пункт меню <i>Вырезать (Cut)</i> .....	53
Пункт меню <i>Копировать (Copy)</i> .....	53
Пункт меню <i>Вставить (Paste)</i> .....	54
Пункт меню <i>Очистить (Clear)</i> .....	54

Пункт меню <i>Новая SUB (New SUB)</i> .....	54
Пункт меню <i>Новая FUNCTION (New FUNCTION)</i> .....	54
2.2.3. Меню <i>Просмотр (View)</i> .....	55
Пункт меню <i>SUBs</i> .....	55
Пункт меню <i>Разбить (Split)</i> .....	55
Пункт меню <i>Экран вывода (Output Screen)</i> .....	56
2.2.4. Меню <i>Поиск (Search)</i> .....	56
Пункт меню <i>Поиск (Find)</i> .....	56
Пункт меню <i>Повторить поиск (Repeat Last Find)</i> .....	57
Пункт меню <i>Замена (Change)</i> .....	57
2.2.5. Меню <i>Запуск (Run)</i> .....	57
Пункт меню <i>Запуск (Start)</i> .....	58
Пункт меню <i>Перезапуск (Restart)</i> .....	58
Пункт меню <i>Продолжить (Continue)</i> .....	58
2.2.6. Меню <i>Отладка (Debug)</i> .....	59
Пункт меню <i>Шаг (Step)</i> .....	59
Пункт меню <i>Процедура на шаг</i> .....	59
Пункт меню <i>Трассировка (Trace On)</i> .....	59
Пункт меню <i>Контрольная точка (Watchpoint)</i> .....	59
Пункт меню <i>Очистить все контрольные точки (Clear All Breakpoints)</i> .....	60
2.2.7. Меню <i>Параметры (Options)</i> .....	60
Пункт меню <i>Экран (Display)</i> .....	60
Пункт меню <i>Путь справки (Set Paths)</i> .....	60
Пункт меню <i>Проверка синтаксиса (Syntax Checking)</i> .....	60
2.2.8. Меню <i>Справка (Help)</i> .....	62
Пункт меню <i>Предметный указатель (Index)</i> .....	62
Пункт меню <i>Содержание (Contents)</i> .....	63
Пункт меню <i>Использование справки (Help on Help)</i> .....	63
<b>Глава 3. Ввод данных</b> .....	<b>65</b>
3.1. Оператор присваивания .....	65
Пример 3.1. Значения функции .....	65
Пример 3.2. Конкатенация символьных переменных .....	66
Пример 3.3. Рисование лесенки .....	67
3.2. Оператор <i>INPUT</i> .....	68
Пример 3.4. Советы постороннего .....	69
Пример 3.5. Ввод хокку .....	71
Пример 3.6. Ввод/вывод элементов матрицы .....	72
3.3. Операторы <i>READ, DATA</i> .....	73
Пример 3.7. Ввод числовой последовательности .....	74
Пример 3.8. Анкета сотрудника .....	75
Пример 3.9. Изображения созвездий .....	76
3.4. Другие возможности .....	77
Пример 3.10. Различия в применении оператора <i>LINE INPUT</i> .....	78
Пример 3.11. Задержка до нажатия любой клавиши .....	79
Пример 3.12. Ввод пароля .....	80
<b>Глава 4. Вывод данных</b> .....	<b>83</b>
4.1. Оператор <i>PRINT</i> .....	83
Пример 4.1. Использование разных разделителей в операторе <i>PRINT</i> .....	84

4.2. Операторы, совместимые с оператором <i>PRINT</i> .....	86
Пример 4.2. Вывод элементов матрицы с помощью табуляции .....	86
Пример 4.3. Вывод данных в виде таблицы.....	87
Пример 4.4. Нерегулярный вывод .....	88
4.3. Оператор <i>PRINT USING</i> .....	88
Пример 4.5. Ввод данных с помощью оператора <i>PRINT USING</i> .....	90
Пример 4.6. Вывод матрицы в виде таблицы .....	90
4.4. Другие возможности .....	91
Пример 4.7. Различия между <i>WRITE</i> и <i>PRINT</i> .....	91
Пример 4.8. Использование функций <i>POS</i> и <i>CSRLIN</i> .....	92
<b>Глава 5. Условные операторы.....</b>	<b>95</b>
5.1. Оператор <i>IF...THEN</i> .....	95
Пример 5.1. Полная линейная форма оператора <i>IF...THEN</i> .....	96
Пример 5.2. Блочная форма оператора <i>IF...THEN</i> .....	102
Пример 5.3. Рисование ломаной линии.....	103
5.2. Оператор <i>SELECT CASE</i> .....	104
Пример 5.4. Перебор вариантов с помощью оператора <i>SELECT CASE</i> .....	105
Пример 5.5. Использование <i>TO</i> и <i>IS</i> в операторе <i>SELECT CASE</i> .....	106
Пример 5.6. Символьное выражение выбора в операторе <i>SELECT CASE</i> .....	107
Пример 5.7. Использование оператора <i>SELECT CASE</i> при создании движения .....	108
5.3. Оператор безусловного перехода <i>GOTO</i> .....	110
Пример 5.8. Использование оператора <i>ON...GOTO</i> .....	111
<b>Глава 6. Операторы цикла.....</b>	<b>113</b>
6.1. Назначение циклов .....	113
Пример 6.1. Построение графика по точкам.....	113
6.2. Оператор <i>FOR...NEXT</i> .....	115
Пример 6.2. Вычисление значений функции .....	117
Пример 6.3. Вывод массива в обратном порядке .....	118
Пример 6.4. Вывод цветной ленты .....	120
Пример 6.5. Вычисление суммы .....	121
Пример 6.6. Использование вложенных циклов при работе с матрицами.....	123
Пример 6.7. Треугольник Паскаля.....	124
6.3. Оператор <i>WHILE...WEND</i> .....	125
Пример 6.8. Определение высоты подъема .....	126
Пример 6.9. Тест.....	127
Пример 6.10. Горизонтальное движение шариков .....	129
6.4. Оператор <i>DO...LOOP</i> .....	130
Пример 6.11. Задача про муху и двух путников .....	130
Пример 6.12. Финансовая пирамида.....	134
Пример 6.13. Максимальная дальность полета тела .....	135
<b>Глава 7. Массивы.....</b>	<b>139</b>
7.1. Назначение массивов.....	139
7.2. Операторы для работы с массивами .....	141
7.3. Работа с массивами.....	142
Пример 7.1. Поиск в одномерном массиве .....	143

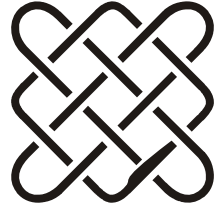
Пример 7.2. Псевдографическая цветомузыка .....	144
Пример 7.3. Обнуление элементов матрицы.....	147
Пример 7.4. Вывод платежной ведомости .....	148
Пример 7.5. Вывод трехмерной матрицы .....	149
Пример 7.6. Вывод баллов за конкурсы КВН.....	151
<b>Глава 8. Работа с графикой .....</b>	<b>153</b>
8.1. Графические операторы .....	153
Пример 8.1. Действия оператора <i>CLS</i> .....	153
Пример 8.2. Вывод символов разными цветами.....	156
Пример 8.3. Использование оператора <i>CIRCLE</i> .....	158
Пример 8.4. Использование оператора <i>LINE</i> .....	161
Пример 8.5. Движение отрезка .....	162
Пример 8.6. Построение зеркального изображения .....	165
8.2. Статическая графика .....	166
Пример 8.7. Рисование лампочки .....	168
Пример 8.8. Рисование нескольких лампочек .....	169
Пример 8.9. Гамма .....	170
8.3. Динамическая графика (анимация).....	171
Пример 8.10. Прыгающая девочка.....	175
Пример 8.11. Вращение вокруг опорной точки.....	176
Пример 8.12. Прямолинейное движение и вращение .....	177
Пример 8.13. Движение нескольких объектов одновременно .....	180
Пример 8.14. Бегущая строка.....	183
Пример 8.15. Удар молнии.....	185
<b>Глава 9. Работа с файлами .....</b>	<b>187</b>
Пример 9.1. Выбор максимального значения.....	187
9.1. Операторы, управляющие работой файла .....	188
9.2. Операторы, управляющие данными.....	190
9.3. Файлы последовательного типа доступа .....	191
Пример 9.2. Создание файла последовательного типа .....	191
Пример 9.3. Добавление данных в файл последовательного типа.....	192
Пример 9.4. Чтение данных из файла последовательного типа .....	192
Пример 9.5. Использование функции <i>EOF</i> .....	193
9.4. Другие возможности .....	194
Пример 9.6. Использование функции <i>LOF</i> .....	194
Пример 9.7. Использование функции <i>FILEATTR</i> .....	195
Пример 9.8. Использование функции <i>FREEFILE</i> .....	195
Пример 9.9. Использование оператора <i>SEEK</i> .....	197
<b>Глава 10. Работа со строковыми переменными.....</b>	<b>199</b>
10.1. Функции и операторы обработки символьных строк.....	199
10.1.1. Функции <i>CHR\$</i> и <i>ASC</i> .....	199
Пример 10.1. Использование функции <i>CHR\$</i> .....	200
Пример 10.2. Использование функции <i>ASC</i> .....	201
10.1.2. Функция <i>LEN</i> .....	202
Пример 10.3. Использование функции <i>LEN</i> .....	202

10.1.3. Функции <i>STRING\$</i> и <i>SPACES</i> .....	203
Пример 10.4. Использование функции <i>STRING\$</i> .....	203
Пример 10.5. Использование функции <i>SPACES</i> .....	204
10.1.4. Функции <i>STR\$</i> и <i>VAL</i> .....	205
Пример 10.6. Использование функции <i>STR\$</i> .....	205
Пример 10.7. Использование функции <i>VAL</i> .....	206
10.1.5. Функции <i>RIGHT\$</i> и <i>LEFT\$</i> .....	206
Пример 10.8. Сокращение слов.....	207
10.1.6. Функция и оператор <i>MID\$</i> .....	207
Пример 10.9. Использование функции <i>MID\$</i> .....	207
Пример 10.10. Использование оператора <i>MID\$</i> .....	209
10.1.7. Функция <i>INSTR</i> .....	209
Пример 10.11. Использование функции <i>INSTR</i> .....	210
10.1.8. Функции <i>HEX\$</i> и <i>OCT\$</i> .....	211
Пример 10.12. Использование функций <i>HEX\$</i> и <i>OCT\$</i> .....	211
10.2. Строковые операции .....	212
Пример 10.13. Конкатенация строк .....	212
Пример 10.14. Сортировка по алфавиту.....	213
10.3. Другие возможности .....	214
10.3.1. Функции <i>LTRIM\$</i> и <i>RTRIM\$</i> .....	214
Пример 10.15. Использование функций <i>LTRIM\$</i> и <i>RTRIM\$</i> .....	214
10.3.2. Функции <i>LCASE\$</i> и <i>UCASE\$</i> .....	215
Пример 10.16. Управление регистром.....	215
<b>Глава 11. Подпрограммы.....</b>	<b>217</b>
11.1. Подпрограммы-функции <i>FUNCTION</i> .....	219
Пример 11.1. Вычисление десятичного логарифма .....	221
Пример 11.2. Вычисление числа сочетаний.....	222
Пример 11.3. Использование ключевого слова <i>STATIC</i> .....	222
Пример 11.4. Рекурсивная функция вычисления факториала .....	223
Пример 11.5. Выбор слов заданной длины .....	224
11.2. Подпрограммы-процедуры <i>SUB</i> .....	225
Пример 11.6. Динамическая смена дня и ночи .....	226
Пример 11.7. Сортировка строковых массивов .....	228
Пример 11.8. Игра в "ромбы" .....	230
11.3. Подпрограммы <i>GOSUB...RETURN</i> .....	232
Пример 11.9. Ньютон и яблоко .....	233
Пример 11.10. Стража, шагающая по стене крепости .....	235
Пример 11.11. Использование оператора <i>ON...GOSUB</i> .....	238
11.4. Функция <i>DEF FN</i> .....	239
Пример 11.12. Решение квадратного уравнения .....	240
Пример 11.13. Определение длины слов.....	241
<b>Глава 12. Программирование игр.....</b>	<b>243</b>
<b>Глава 13. Программирование музыки.....</b>	<b>259</b>
Пример 13.1. Мелодия к русскому романсу "День-день-день" .....	266
Пример 13.2. Мелодия к песне о бедном зайчике .....	269
Пример 13.3. Мелодия песни Ю. Визбора "Ты у меня одна" .....	271



---

<b>Приложение 1. Язык программирования BASIC .....</b>	<b>277</b>
GWBasic — первое поколение языка .....	278
QuickBasic — второе поколение языка .....	278
Visual Basic — третье поколение языка .....	279
<b>Приложение 2. Сообщения об ошибке .....</b>	<b>281</b>
<b>Приложение 3. Примеры операторов .....</b>	<b>285</b>
<b>Литература .....</b>	<b>295</b>



# ГЛАВА 1

## Создание программы

### 1.1. Разработка программ

Для того чтобы приступить к программированию, нужно иметь задачу для программирования. После того как она появится, имеющуюся задачу следует формализовать, т. е. описать ее на языке математики или же используя другие формализованные правила. Способ формализации задачи определяет, как правило, структуру вход-выходных данных, которые могут быть представлены отдельными переменными и константами, массивами и файлами сложной структуры.

Затем необходимо продумать ход решения задачи, т. е. разработать алгоритм ее решения. Алгоритм может быть описан словесно, графом (обычно это блок-схема) и на языке программирования (это и будет программа).

Разработка алгоритма является определяющим этапом в процессе программирования, поскольку задает логическую структуру программы. В соответствии с основной теоремой структурного программирования, доказанной Э. Дейкстрой<sup>1</sup>, алгоритм любой сложности можно реализовать, используя только три конструкции: следование (оператор за оператором), повторение (цикл), выбор (альтернатива).

При разработке программы следует иметь в виду, что в укрупненном виде любой алгоритм, а следовательно, и программа, состоит из трех частей:

- ◆ ввод данных;
- ◆ обработка данных;
- ◆ вывод результата.

Язык программирования имеет свой алфавит и зарезервированные слова для команд или операторов, а также служебных инструкций. Зарезервированные слова нельзя использовать в качестве меток или имен переменных, констант и процедур. Кроме команд или операторов, в программах используются числовые и символьные переменные, а также константы. В табл. 1.1 приведены имеющиеся в языке функции, а в *приложении 3* — примеры операторов.

---

<sup>1</sup> Эдсгер Дейкстра — почетный заведующий кафедрой, созданной компаниями Schlumberger и Centennial на факультете информатики (Computer Sciences) университета штата Техас. Он является выдающимся представителем научного направления, которое принято называть теоретическим программированием. Вся его научная деятельность была направлена на разработку средств и методов для создания "правильных" программ, т. е. таких программ, корректность которых может быть доказана формальными методами.

Таблица 1.1. Операции и функции в BASIC

Операции, функции	Вид в BASIC
Сложение	+
Вычитание	-
Умножение	*
Деление	/
Возведение в степень	^
Целочисленное деление	\
$e^x$	EXP (x)
$ x $	ABS (x)
$\sin x$	SIN (x)
$\cos x$	COS (x)
$\operatorname{tg} x$	TAN (x)
$\ln x$	LOG (x)
$\lg x$	LOG (x) / LOG (10)
$\sqrt{x}$	SQR (x)
$\operatorname{arctg} x$	ATN (x)
Остаток от деления	i MOD j
Генератор случайных чисел	RND (n)
Равно	=
Больше	>
Меньше	<
Не равно	<>
Меньше или равно	<=
Больше или равно	>=

### Примечание

1. Согласно приоритету операций в BASIC сначала выполняется возведение в степень, затем умножение и деление, после сложение, вычитание. Операции, размещенные в одной строке и имеющие равный приоритет, выполняются последовательно слева направо.
2. BASIC вычисляет выражение в скобках в первую очередь, даже если операции в скобках более низкого приоритета, чем вне скобок. Сначала вычисляются внутренние скобки, затем наружные.
3. Обратные функции вычисляются через имеющиеся встроенные:

секанс:  $\operatorname{SEC}(x) = 1 / \operatorname{COS}(x)$ ;

косеканс:  $\operatorname{COSEC}(x) = 1 / \operatorname{SIN}(x)$ ;

арксинус:  $\operatorname{ARCSIN}(x) = \operatorname{ATN}(x / \operatorname{SQR}(1 - x * x))$ ;

арккотангенс:  $\operatorname{ARCCTN}(x) = 1.570796 - \operatorname{ATN}(x)$ ;

арккосинус:  $\operatorname{ARCCOS}(x) = 1.570796 - \operatorname{ATN}(x / \operatorname{SQR}(1 - x * x))$ .

Рассмотрим процесс разработки программы на несложном экономическом примере.

### Пример 1.1. Экономическая задача

Требуется разработать программу расчета времени накопления заданной суммы средств  $Kk$  от начальной суммы вклада  $Kn$ , если процентная ставка равна  $p$ , при условии начисления простых и сложных ежегодных процентов. Сравнить эти варианты.

Задача формализована, поскольку имеются формулы для вычисления суммы при заданных видах процентов:

$$S = Kn \cdot (1 + t \cdot p/100) \quad \text{— для простых процентов,}$$

$$S = Kn \cdot (1 + p/100)^t \quad \text{— для сложных процентов,}$$

где  $Kn$  — начальная сумма вклада;  $p$  — годовая процентная ставка;  $t$  — срок вклада.

Для полной ясности скажем, что простые проценты — это проценты, ежегодно (в нашем случае) начисляемые лишь на начальную сумму вклада, а при сложных процентах происходит капитализация процентов, т. е. проценты начисляются и на проценты, прибавляемые к начальной сумме вклада. Понятно, что при начислении сложных процентов заданная сумма накопится быстрее.

Алгоритм решения задачи в виде блок-схемы приведен на рис. 1.1. Процесс построения блок-схем подробно рассмотрен в *разд. 1.2*. Если располагать текст программы и соответствующую блок-схему рядом (т. е. параллельно), то можно проследить, как блоки блок-схемы переходят в блоки или строки разрабатываемой программы.

#### Программа 1.1

```
'lekonom.bas      Экономическая задача
10 CLS
20 PRINT "Пример 1.1"
30 PRINT "Расчет времени накопления заданной суммы средств"
40 PRINT "при условии начисления простых или сложных ежегодных"
50 PRINT "процентов. Сравнение этих двух вариантов.": PRINT
60 PRINT "Простые проценты – 1"
70 PRINT "Сложные проценты – 2"
80 PRINT "Сравнение вариантов – 3"
90 INPUT "Выбор – ", vib
100 IF vib = 1 OR vib = 2 OR vib = 3 THEN 120 ELSE
110 PRINT "Такого варианта нет": SLEEP 2: GOTO 10
120 PRINT: INPUT "Начальная сумма вклада в руб. = ", Kn
130 INPUT "Ставка годовых % = ", p
140 INPUT "Сумма накопления в руб. = ", Kk
'==1==== Блок выбора =====
150 IF vib = 1 THEN 170
160 IF vib = 2 THEN 240
'==2==== Блок вычисления простых процентов =====
170 DO
180 t1 = t1 + 1
190 S1 = Kn * (1 + t1 * p / 100)
200 IF S1 >= Kk THEN EXIT DO
210 LOOP
220 t = t1: PRINT
230 IF vib = 3 THEN ELSE 330
```

```
'==3==== Блок вычисления сложных процентов =====
240 DO
250 t2 = t2 + 1
260 S2 = Kn * (1 + p / 100) ^ t2
270 IF S2 >= Kk THEN EXIT DO
280 LOOP
290 t = t2: PRINT
300 IF vib = 3 THEN ELSE 330
'==4==== Блок сравнения вариантов =====
310 t = t1 - t2
320 PRINT "Разница во времени накопления - ", t: GOTO 340
'==5==== Блок вывода результата =====
330 PRINT "Время накопления заданной суммы"; t;
340 IF t < 5 THEN ELSE PRINT "лет"
350 IF t = 1 THEN PRINT "год" ELSE IF t < 5 THEN PRINT "года"
'=====
360 END
```

Программу рекомендуется начинать со строки комментариев, в которой в качестве названия программы использовать название файла, в который записана эта программа. Так в данном случае это будет — `lekonom.bas`. Затем в этой же строке желательно дать небольшой комментарий с описанием программы.

Как правило, программу в BASIC начинают с оператора `CLS` — очистка экрана, чтобы на экране была только информация, относящаяся к этой программе, а посторонняя информация не мешала бы работе.

Теперь о нумерации строк программы. Она обязательна только для тех строк, на которые есть ссылки в операторах `GOTO`, `GOSUB`, `IF...THEN` и т. д. Нумерация в программе удобна в том случае, если необходимо комментировать эту программу. В BASIC принято нумеровать строки с шагом 10 для того, чтобы при внесении изменений в текст программы нумеровать новые строки в промежутках существующей нумерации, например между строками с номерами 10 и 20 можно вставить строки с номерами 11, 12, 13 и т. д. При этом номера других линий останутся без изменений, а значит не нужно корректировать ссылки в указанных операторах.

После оператора `CLS` желательно с помощью оператора `PRINT` вывести на экран название программы, а может быть, и краткую формулировку поставленной задачи. В программе 1.1 заголовок выводится оператором `PRINT` в строке 20, а краткая формулировка дана в строках 30—50. Операторы `PRINT` в строках 60—80 выводят на экран своеобразное меню, подсказывающее варианты выбора. Таким образом, блок 1 блок-схемы на рис. 1.1 отражен строками 60—90 программы 1.1. Строки 100—110 программы осуществляют анализ сделанного выбора. В литературе часто это называют "защитой от дурака". Авторы не согласны с такой формулировкой и предлагают называть это "блокировкой любознательного", слишком любознательного.

Блок 2 блок-схемы реализуется с помощью операторов `INPUT` в строках 120—140 программы. Оператор приостанавливает работу программы и ожидает ввода данных с клавиатуры. Ввод заканчивается нажатием клавиши `<Enter>`. Переменные в операторах `INPUT` в строках 120—140 являются числовыми, поэтому программа ожидает ввода чисел. Попытка ввода текста вызовет появление сообщения "Ввод сначала". Выражение в кавычках при операторе `INPUT` обычно называют "подсказкой" или "приглашением к вводу", и оно является весьма желательной вещью для правильного оформления программы. Подробнее оператор `INPUT` будет рассмотрен в *разд. 3.2*.

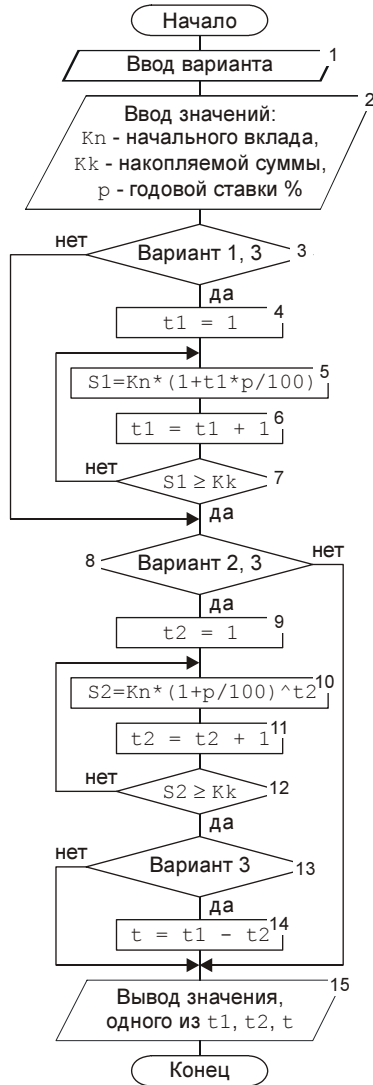


Рис. 1.1. Блок-схема экономической задачи

Ветвление, задаваемое блоками 3, 8, 13 блок-схемы, осуществляется посредством условного оператора IF...THEN в строках 150—160, 230, 300 программы 1.1. Между ключевыми словами IF и THEN условного оператора помещается условие. Если условие выполняется, то осуществляется переход по метке, находящейся после ключевого слова THEN, а если после слова THEN находятся операторы, то они выполняются. Подробнее условный оператор IF...THEN будет рассмотрен в *разд. 5.1*.

Программа 1.1 содержит два цикла DO...LOOP по годам ( $t1 = t1 + 1$  и  $t2 = t2 + 1$ ), первый из которых создан для вычисления времени накопления заданной суммы при простых процентах (цикл в строках 170—210, соответствующая формула вычисляется в строке 190), а второй — для вычисления времени накопления заданной суммы при сложных процентах (цикл в строках 240—280, формула в строке 260). Отметим, что циклический оператор

`DO...LOOP` предназначен для организации цикла с заранее неизвестным числом повторений, а завершение цикла происходит по заданному условию. Подробнее о циклическом операторе `DO...LOOP` читайте в *разд. 6.4*.

В строках 180 и 250 на каждом шаге циклов происходит увеличение на 1 счетчика циклов по годам (от 1 до искомого значения). Выход из цикла организован одинаково в обоих вариантах в строках 200 и 270 с помощью условного оператора `IF...THEN`, в который поставлен оператор `EXIT DO`. Условие выхода из циклов — накопление заданной суммы ( $S1 \geq Kk$  и  $S2 \geq Kk$ ). Строки 220 и 290 подготавливают к выводу на экран найденного времени накопления заданной суммы, осуществляемого посредством оператора `PRINT` в строке 330. Операторы `PRINT` в строках 220 и 290 задают пустую строку, отделяющую вывод времени накопления от предыдущего текста. Условные операторы `IF...THEN` в строках 340—350 предназначены для согласования величины значения  $t$  при выводе с последующим пояснением ("год", "года", "лет").

Теперь пройдемся по ветвлениям программы 1.1.

Пусть выбран первый вариант — простые проценты. Тогда условный оператор `IF...THEN` в строке 150 задает переход на строку 170. Начинает работать цикл `DO...LOOP` в строках 170—210 по вычислению времени накопления заданной суммы в случае простых процентов. По завершении работы этого цикла условный оператор `IF...THEN` в строке 230 задает по `ELSE` переход на линию 330, где находится выводящий на экран оператор `PRINT`.

Теперь пусть выбран второй вариант — сложные проценты. Тогда условный оператор `IF...THEN` в строке 160 задает переход на строку 240. Начинает работать второй цикл `DO...LOOP` в строках 240—280 по вычислению времени накопления заданной суммы в случае сложных процентов. После завершения вычисления времени накопления условный оператор `IF...THEN` в строке 300 задает переход на линию 330 для вывода результата на экран.

И, наконец, последний — третий вариант. В этом случае условные операторы `IF...THEN` в строках 150—160 пропускают в цикл `DO...LOOP` в строках 170—210. Затем условный оператор `IF...THEN` в строке 230 пропускает в цикл `DO...LOOP` в строках 240—280. После чего условный оператор `IF...THEN` в строке 300 передает управление на строку 310 вычисления разницы во времени накопления при условии простых и сложных процентов ( $t = t1 - t2$ ). Результат на экран выводит оператор `PRINT` в строке 320.

Итак, блоки 4—7 блок-схемы на рис. 1.1 переходят в строки 170—210 программы 1.1, причем блок 5 реализуется строкой 190, блок 7 — строкой 200, а блоки 4, 6 — строкой 180.

```

Пример 1.1
Расчет времени накопления заданной суммы средств
при условии начисления простых или сложных ежегодных
процентов. Сравнение этих двух вариантов.

Простые проценты - 1
Сложные проценты - 2
Сравнение вариантов - 3
Выбор - 1

Начальная сумма вклада в руб. = 1000
Ставка годовых % = 10
Сумма накопления в руб. = 2000

Время накопления заданной суммы 10 лет

```

Рис. 1.2. Вывод результата вычисления времени накопления заданной суммы при условии простых процентов

Аналогично, блоки 9—12 отображаются в строки 240—280, блоки 9, 11 реализуются строкой 250, блок 10 — строкой 260, а блок 270, определяющий условие выхода из цикла, — строкой 270.

Блок 14 рассматриваемой блок-схемы переходит в строку 310 программы 1.1, а блок 15 реализуется операторами PRINT в строках 320—350.

Результат работы программы 1.1 приведен на рис. 1.2.

## Пример 1.2. Олимпийские кольца

Требуется разработать программу, рисующую олимпийские кольца.

Этот пример показывает, как можно, используя массивы, применить циклы для обработки нерегулярных данных. В программе, рисующей олимпийские кольца, такими нерегулярными данными является цвет колец: синий — 11, черный — 8, красный — 12, желтый — 14 и зеленый — 10 (номера цветов BASIC смотрите в табл. 8.2 в разд. 8.1). Рекомендуется перед разработкой программы предполагаемый рисунок изобразить на бумаге в клетку, что позволит определить ориентировочные координаты опорных точек фигуры, которые будут уточняться в процессе рисования. Подобный эскиз приведен на рис. 1.3.

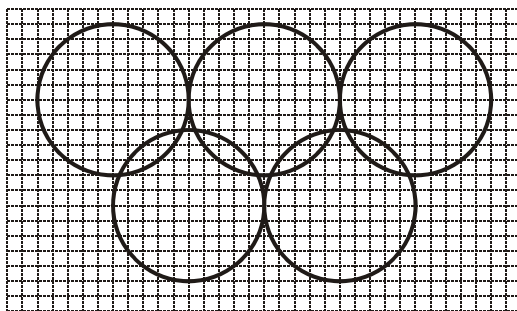


Рис. 1.3. Олимпийские кольца

Блок-схему для этой программы предлагается читателю разработать самостоятельно, и поэтому обратимся непосредственно к программе.

### Программа 1.2

```
'OlimpKol.bas      Олимпийские кольца
10  CLS              'очистка экрана
20  SCREEN 9        'установка экранного режима
30  COLOR 14, 9     'установка экранных цветов (9 — цвет фона)
40  DIM cvet(5)     'объявляется массив для записи номеров цвета
'==1===== Блок рисования колец =====
50  xn = 200: xk = 400
60  FOR y = 100 TO 150 STEP 50
70  FOR x = xn TO xk STEP 100
80  DATA 11,8,12,14,10
90  k = k + 1: READ cvet(k)
100 CIRCLE (x, y), 50, cvet(k)
110 CIRCLE (x, y), 49, cvet(k)
```



```

120 CIRCLE (x, y), 48, cvet(k)
130 NEXT x
140 xn = 250: xk = 350
150 NEXT y
'=====
160 END

```

Кольцо рисуется операторами `CIRCLE` в строках 100—120 (3 оператора со смежными значениями радиуса — 50, 49, 48 для создания толщины). Координаты центров колец задаются посредством двойного цикла (внешний в строках 60—150 задает координату  $y$ , а внутренний в строках 70—130 задает  $x$ ). Цвет колец вводят операторы `READ` и `DATA` в строках 80—90. Подробнее об этих операторах читайте в *разд. 3.3*. Поскольку в верхнем ряду 3 кольца, а в нижнем — только 2, то и начальные и конечные значения счетчика цикла в строке 70 будут меняться в зависимости от ряда. Для верхнего (3 кольца) они задаются в строке 50, а для нижнего (2 кольца) — в строке 140.

Теперь рассмотрим решение оптимизационной задачи.

### Пример 1.3. Оптимизационная задача "Диета"

Фирма занимается составлением диеты, содержащей по крайней мере 20 единиц белков, 30 единиц углеводов, 10 единиц жиров и 40 единиц витаминов. Как дешевле всего достичь соблюдения диеты при указанных в табл. 1.2 ценах (в рублях) на 1 кг (или 1 л) имеющихся пяти продуктов?

**Таблица 1.2.** Содержание элементов диеты в пяти продуктах и их цена

	Хлеб	Соя	Сушеная рыба	Фрукты	Молоко
Белки	2	12	10	1	2
Углеводы	12	0	0	4	3
Жиры	1	8	3	0	4
Витамины	2	2	4	6	2
Цена	12	36	32	18	10

Блок-схема алгоритма решения этой задачи приведена на рис. 1.4.

#### Программа 1.3

```

'1Dieta.bas      Диета
10 CLS
20 Cmin = 10000
'==1===== Блок оптимизации =====
30 FOR x1 = 0 TO 20
40 LOCATE 1, 1: PRINT "cikl ="; x1
50 FOR x2 = 0 TO 20
60 FOR x3 = 0 TO 10
70 FOR x4 = 0 TO 20
80 FOR x5 = 0 TO 20
90 u1 = 0: u2 = 0: u3 = 0: u4 = 0

```

```
'---- Блок проверки условий -----
100 IF 2 * x1 + 12 * x2 + 10 * x3 + x4 + 2 * x5 >= 20 THEN u1 = 1
110 IF 12 * x1 + 4 * x4 + 3 * x5 >= 30 THEN u2 = 1
120 IF x1 + 8 * x2 + 3 * x3 + 4 * x5 >= 10 THEN u3 = 1
130 IF 2 * x1 + 2 * x2 + 4 * x3 + 6 * x4 + 2 * x5 >= 40 THEN u4 = 1
'-----

140 IF u1 + u2 + u3 + u4 = 4 THEN ELSE 180
150 c = 12 * x1 + 36 * x2 + 32 * x3 + 18 * x4 + 10 * x5
160 IF c < Cmin THEN ELSE 180
170 Cmin = c: Xo1 = x1: Xo2 = x2: Xo3 = x3: Xo4 = x4: Xo5 = x5
180 NEXT x5, x4, x3, x2, x1
'==2==== Блок вывода результата =====
190 PRINT "Минимальная стоимость диеты"; Cmin; "рублей"
200 PRINT "Оптимальное количество хлеба = "; Xo1; "буханок"
210 PRINT "Оптимальное количество сои = "; Xo2; "кг"
220 PRINT "Оптимальное количество сушеной рыбы = "; Xo3; "кг"
230 PRINT "Оптимальное количество фруктов = "; Xo4; "кг"
240 PRINT "Оптимальное количество молока = "; Xo5; "литров"
'=====
250 END
```

Результат работы программы 1.3 показан на рис. 1.5.

Быстродействие современных компьютеров позволяет решать оптимизационные задачи самым простым образом — путем перебора, осуществляемого в рассматриваемом примере по пяти видам продуктов. Введем следующие обозначения для переменных:

- ◆ x1 — для хлеба;
- ◆ x2 — для сои;
- ◆ x3 — для сушеной рыбы;
- ◆ x4 — для фруктов;
- ◆ x5 — для молока.

Таким образом, для осуществления перебора следует организовать пятерной вложенный цикл (с помощью операторов цикла FOR...NEXT в строках 30—180 программы 1.3), в теле которого на каждом шаге проверяются условия по количеству белков (блок 8 блок-схемы на рис. 1.4), углеводов (блок 11), жиров (блок 14) и витаминов (блок 17). Данные для условий берутся из табл. 1.2. Например, для белков: x1 единиц хлеба дает 2 \* x1 единиц белка, x2 единиц сои дает 12 \* x2 единиц белка, x3 единиц сушеной рыбы дает 10 \* x3 единиц белка и т. д. Следовательно, условие по белку будет выглядеть следующим образом:

$$2 * x1 + 12 * x2 + 10 * x3 + x4 + 2 * x5 \geq 20$$

и реализуется в блоках 8—10 блок-схемы и посредством условного оператора IF...THEN в строке 100 программы 1.3.

Аналогично определяются условия по углеводам (блоки 11—13 и строка 110):

$$12 * x1 + 4 * x4 + 3 * x5 \geq 30,$$

по жирам (блоки 14—16 и строка 120):

$$x1 + 8 * x2 + 3 * x3 + 4 * x5 \geq 10,$$

по витамину (блоки 17—19 и строка 130):

$$2 * x1 + 2 * x2 + 4 * x3 + 6 * x4 + 2 * x5 \geq 40.$$

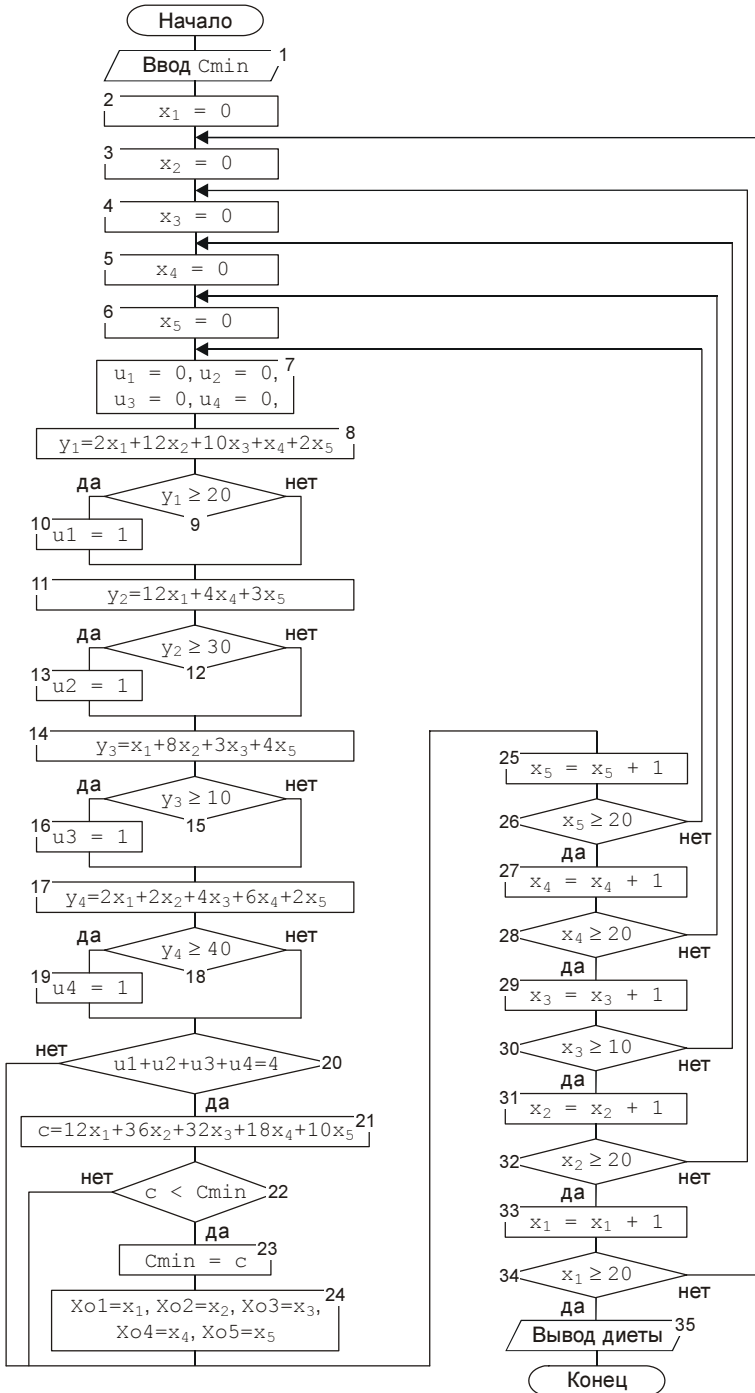


Рис. 1.4. Блок-схема оптимизационной задачи

```

Минимальная стоимость диеты 152 рублей
Оптимальное количество хлеба = 0 буханок
Оптимальное количество сои = 0 кг
Оптимальное количество сушеной рыбы = 0 кг
Оптимальное количество фруктов = 4 кг
Оптимальное количество молока = 8 литров

```

Рис. 1.5. Вывод программы "Диета"

В случае одновременного выполнения всех четырех условий (блок 20 и строка 140, все вспомогательные переменные  $u_1$ — $u_4$  равны 1, а, значит, их сумма равна 4), вычисляется стоимость набора продуктов (блок 21 и строка 150), после чего эта стоимость сравнивается с предыдущей минимальной стоимостью  $C_{\min}$  (блок 22 и строка 160). Если вычисленная стоимость меньше предыдущей минимальной, то она принимается за новую минимальную (блок 23 и строка 170). Описанная процедура повторяется до окончания полного перебора.

Верхние границы перебора определяются по табл. 1.2 по строке "Витамины" для всех продуктов, за исключением фруктов. Для них верхняя граница перебора определяется по строке "белки". Поясним логику выбора на примере хлеба. Условие по витаминам выполняется при  $x_1 = 20$  ( $2 * x_1 = 40$ ), и при этом заведомо выполняются соответствующие условия и по остальным показателям — по углеводам  $12 * x_1 = 240 > 30$ , по белкам —  $2 * x_1 = 40 > 20$  и по жирам —  $x_1 = 20 > 10$ .

В строках 190—240 программы 1.3 посредством шести операторов PRINT производится вывод на экран состава минимальной по стоимости диеты и ее цена.

Следующий пример на совместное использование графики и звука взят из книги [9].

## Пример 1.4. Гамма

Разработать программу, играющую гамму, т. е. воспроизводящую ноты первой октавы вначале в прямом, а затем в обратном порядке. В ходе выполнения программы на экран выводится нотный стан с нотами, причем звучащая нота выделяется другим цветом.

Для решения поставленной задачи разработана программа 1.4, результат работы которой показан на рис. 1.6.

Блок-схема алгоритма решения задачи, поставленной в примере 1.4, приведен на рис. 1.7. Из рисунка видно, что программа должна состоять из четырех циклов:

- ◆ цикл для ввода нот — блоки 1—4;
- ◆ цикл для рисования нотного стана и нот — блоки 5—9;
- ◆ цикл для озвучивания нот в прямом порядке и выделения звучащей ноты другим цветом — блоки 10—14;
- ◆ цикл для озвучивания нот в обратном порядке и выделения звучащей ноты другим цветом — блоки 15—19.

В программе 1.4 эти четыре цикла реализуются посредством операторов цикла FOR...NEXT в строках 50—120 (внешний цикл) и в строках 90—120 (вложенный цикл), а также условного оператора SELECT CASE, являющегося телом вложенного цикла. Оператор FOR...NEXT с числом повторений 4, образующий внешний цикл, обеспечивает реализацию всех четырех перечисленных циклов. А оператор цикла FOR...NEXT в строках 90—120 с изменяемыми параметрами счетчика цикла конкретизирует указанные циклы.