О.П. Зеленяк

ПРАКТИКУМ ПРОГРАММИРОВАНИЯ НА TURBO PASCAL

ЗАДАЧИ, АЛГОРИТМЫ И РЕШЕНИЯ

Издание третье,
переработанное и дополненное

ДМК ПРЕСС
издательство

торгово-издательский дом
DiaSoft

Москва, Санкт-Петербург

ББК 32.973.2
УДК 681.3. 06(075)
З 58

Зеленяк О.П.
З 58 Практикум программирования на Turbo Pascal. Задачи, алгоритмы и решения.— 3-е изд., испр. и доп. — СПб.: ДиаСофтЮП, М.: ДМК Пресс. — 320 с.

ISBN 5-93772-187-X
ISBN 5-94074-355-2

В книге содержится более 200 задач по программированию в среде Turbo Pascal, снабженных решениями.

В начале каждой главы даётся соответствующий справочный материал. Внутри глав задачи расположены в порядке возрастания трудности. Большое внимание уделено графическим построениям. Последняя глава содержит задачи олимпиад различного уровня по информатике.

Издание предназначено для старшеклассников, студентов, преподавателей.

ББК 32.973.2
УДК 681.3. 06(075)

Рецензенты:

заведующий кафедрой математики Кировоградского государственного педагогического университета, доктор физико-математических наук, профессор Волков Ю.И.

заведующая кафедрой прикладной математики Харьковского государственного политехнического университета, доктор технических наук, профессор Курпа Л.В.

Все права зарезервированы, включая право на полное или частичное воспроизведение в какой бы то ни было форме.

Материал, изложенный в данной книге многократно проверен. Но поскольку вероятность технических ошибок все равно остается, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

Все торговые знаки, упомянутые в настоящем издании, зарегистрированы. Случайное неправильное использование или пропуск торгового знака или названия его законного владельца не должно рассматриваться как нарушение прав собственности.

ISBN 5-93772-187-X
ISBN 5-94074-355-2

© Зеленяк О.П.
© ДиаСофтЮП
© Оформление, ДиаСофтЮП
© Обложка, издание ДМК Пресс

Оглавление

Предисловие 5

Глава 1. **Арифметика вещественных чисел. Вычисления по формулам** **7**
Целый тип 7
Вещественный тип 8
Математические функции 10
Запись выражений на паскале 11

Глава 2. **Разветвления** **17**
Логический тип 17
Символьный тип 18
Строковый тип 18
Ветвление в краткой и полной форме 19
Вложенные ветвления 20
Оператор выбора Case 29

Глава 3. **Циклы** **32**
Цикл с параметром 33
Цикл с предусловием 40
Цикл с постусловием 41

Глава 4. **Сочетание цикла и разветвления** **49**
Фундаментальные управляющие структуры 49
Способы ввода данных в программу 49

Глава 5. **Одномерные массивы** **57**
Инициализация массивов 58
Стандартные задачи на одномерные массивы 59
Поиск элемента в массиве 67
Включение (удаление) элемента в заданную позицию массива. Циклический сдвиг массива 71
Случайная выборка элементов массива 77
Методы внутренней сортировки массивов 78

Глава 6. **Вложенные циклы** **81**
Вложенные циклы в матричных задачах 83
Вложенные циклы в переборных задачах 93

Глава 7. **Подпрограммы. Рекурсия** **103**
Процедуры 104
Функции 113
Рекурсия 117

Глава 8. **Строки, множества, записи** **129**
Обработка последовательностей символов 129
Множества 144
Записи 150

Глава 9. **Графические построения** **154**
Программы, в которых используемые данные преимущественно константы 158
Программы, в которых используется координатное задание геометрических преобразований 167
Программы, в которых используются геометрические соотношения и масштабирование 177
Программы построения кривых на плоскости 188
Программы, использующие случайные числа и рекурсию 195
Занимательная графика 203

Глава 10. **Файлы** **212**
Текстовые файлы 213
Типизированные файлы 220
Бестиповые файлы 229

Глава 11. **Модули** **231**

Глава 12. **Задачи олимпиад по информатике** **237**

Рисунки к главе 9 . . . . . **296**

Список использованной и рекомендованной литературы **308**

Предметный указатель **309**

Предисловие

Идея *обучения на задачах* не нова. Она давно и успешно применяется при обучении математике, физике, химии и другим дисциплинам, по которым имеется достаточное количество сборников задач, в том числе и с решениями. По программированию, особенно для начинающих, подобной литературы очень мало. Выходят, в основном, книги-справочники, описывающие ту или иную среду программирования, её операторы и функции. Примеры в них только иллюстрируют применение языковых конструкций. Такие книги, безусловно, нужны, но на личном опыте преподавания информатики и математики я убедился, насколько важно решать содержательные и взаимосвязанные задачи. *При обучении программированию необходимо составлять, анализировать, запоминать и записывать алгоритмы*. “Алгоритмы встречаются в науке на каждом шагу: умение решать задачу в общем виде всегда означает, по-существу, владение некоторым алгоритмом”[19].

Как в математике важно довести задачу до окончательного ответа, так в программировании не менее важно получить законченную программу. “Считаю важным, показав программу в окончательном виде, обращать внимание на детали, поскольку именно в них кроется трудность программирования. Описание принципов самого алгоритма и его математический анализ могут стимулировать “академический ум” и бросить ему вызов, но это было бы бесчестно по отношению к программисту-практику. Поэтому я строго придерживаюсь правила давать программы в окончательном виде, на том языке, на котором они действительно могут быть выполнены на вычислительной машине”, – пишет Н. Вирт ([3], стр.13). Большинство преподавателей согласятся с этим высказыванием и с тем, что не должно быть приоритета идеи алгоритма или отлаженной программы на начальной стадии обучения программированию. *Для записи алгоритмов наилучшим образом подходит язык паскаль, который и создавался как учебный язык программирования*.

Примеры в этом пособии расчитаны на начинающих программировать школьников, студентов, пользователей. Они могут быть использованы для самостоятельного обучения и, надеюсь, окажутся полезными преподавателям, проводящим практические занятия. Часть условий задач взята из сборника **[1]** с сохранением нумерации задач в нём. В решениях внимание уделяется алгоритмам и используются только основные структуры данных: массивы, строки, записи, множества, файлы.

Последующие шаги в сторону углублённого изучения программирования – это знакомство с особенностями операционной системы, изучение разделов дискретной математики и линейного программирования, конструирование данных с динамической структурой. При разработке сложных программ возникает необходимость в использовании принципов и техники объектно-ориентированного программирования.

Книга содержит 200 задач, снабженных решениями. Решение задачи, т.е. законченную программу, следует прочесть после того, как задача решена самостоятельно. Однажды сделанное разъяснение, как правило, не повторяется в последующих задачах, поэтому рекомендуется решать задачи подряд. В начале каждой главы даётся соответствующий справочный материал. Внутри глав задачи расположены в порядке возрастания трудности. Дополнительные задачи для самостоятельного решения можно найти в указанном сборнике.

Большое внимание уделено *графическим построениям*, что обусловлено следующими причинами: сегодня наиболее динамично развиваются компьютерные информационно-графические технологии – интерфейс операционной системы Windows, гипертекстовый формат Web-страниц в сети Internet и т.п; изучение компьютерной графики нельзя отождествлять с изучением или применением программного обеспечения, подобного графическим редакторам, – значительно важнее создавать графические объекты, используя компьютерные геометрические модели; при изучении машинной графики можно относительно просто демонстрировать и визуализировать многие важные понятия и приёмы программирования – представление данных, кодирование данных, вложенные циклы, параметры процедур.

Задачи последней главы – это *задачи олимпиад различного уровня* по информатике. Они могут быть использованы на кружковых занятиях для подготовки учащихся и студентов к олимпиадам по программированию.

1. Арифметика вещественных чисел. Вычисления по формулам

♦ Целый тип

В Паскале определены *пять типов (диапазонов значений) для целых чисел*. Областью значений каждого из них является подмножество множества целых чисел.

Тип	Диапазон	Байт
ShortInt – короткое целое со знаком	-128 .. 127	1
Integer – целое со знаком	-32 768 .. 32 767	2
LongInt – длинное целое со знаком	-2 147 483 648 .. ..2 147 483 647	4
Byte – короткое целое без знака	0 .. 255	1
Word – целое без знака	0 .. 65 535	2

В последней колонке таблицы показано, что числу из каждого диапазона отводится фиксированное количество разрядов в памяти ЭВМ. Это позволяет эффективно использовать память и влиять на скорость вычислений. *Все целые типы являются упорядоченными*, поэтому к их значениям применимы две специальные функции Succ(i) и Pred(i), где i – целое, возвращающие следующее и предшествующее i целое число. Определены также две константы MaxInt и MaxLongInt, значениями которых являются числа 32767 и 2147483647, т.е. верхние границы двух соответствующих диапазонов.

Чаще всего целые числа используются в простых арифметических выражениях и выступают в программах в качестве различных счётчиков и значений индексов. Над ними *определены пять основных операций*, результатами которых являются также целые числа:

+	–	сложение;
-	–	вычитание;
*	–	умножение;
div	–	частное от деления;
mod	–	остаток от деления.

Все пять операций – двухместные (бинарные), т.е. применимы к двум аргументам, но операции сложения и вычитания могут использоваться и как одноместные (унарные). При изменении знака числа, например, допустима запись a:= -a.

При делении с точностью до целых получается два результата – частное и остаток. Операция div обозначает целочисленное деление. Знак результата берётся по обычным правилам, а полученный остаток игнорируется. Например, 15 div 2 = 7; 3 div 7 = 0; (-9) div 4 = -2; (-9) div (-4) = 2. Операция mod даёт остаток при делении двух целых чисел. 15 mod 2 = 1; 3 mod 7 = 3; (-9) mod 4 = -1; 9 mod 3 = 0. Операции *, div, mod имеют одинаковый более высокий ранг, а операции + и - также имеют одинаковый, но более низкий ранг.

♦ Вещественный тип

Областью значений вещественного типа является подмножество множества всех вещественных чисел. *Оно не является упорядоченным*. Для представления чисел из широкого диапазона, от очень маленьких до очень больших привычное представление с фиксированной запятой не подходит, т.к. любой вещественный тип в ЭВМ должен иметь ограничение на количество знаков. Если выводятся числа шириной 12 знаков, то, например, число 2 001 000 000 000 555 уже не будет изображено. Поэтому вещественные числа *представляются в ЭВМ в форме с плавающей точкой,* называемой экспоненциальной. Они имеют цифровую часть (мантиссу) и порядок, следующий за знаком e. Приведенное выше число примет вид 2.001e+15. Ясно, что такая форма представления чисел дает возможность выполнять *операции над вещественными числами только приближённо*. Вообще, *нельзя предполагать точное равенство никаких двух чисел с плавающей точкой*. Количество цифр в мантиссе характеризует точность.Чем больше цифр имеет мантисса, тем погрешность меньше.

Погрешность зависит от природы вычислений. Программист должен организовывать вычисления так, чтобы избегать серьёзных ошибок.

Свои особенности имеет использование математического сопроцессора: аппаратные ограничения всегда существуют. Дополнительные трудности возникают также при тестировании программ, содержащих операции над вещественными числами. Следует различать *синтаксические и логические ошибки* и повышать устойчивость программ. Этим важным вопросам следует уделить внимание и для более детального знакомства с ними обратиться к специальной литературе.

Тип	Диапазон	Цифр	Байт
Real – стандартный	2.9e-39 .. 1.7e+38	11-12	6
Single – одинарной точности	1.5e-45 .. 3.4e+38	7-8	4
Double – двойной точности	5.0e-324..1.7e+308	15-16	8
Extended – повышенной точности	3.4e-4932.. 1.1e+4932	19-20	8
Comp – 64-битное целое	-9.2e+18..9.2e+18	19-20	8

Стандартный вещественный тип Real используется без математического сопроцессора, а типы *Single, Double, Extended* относятся к *расширенному набору вещественных типов и рассчитаны на работу с сопроцессором,* применение которого значительно увеличивает точность расчётов и ускоряет их выполнение. Паскаль дает возможность эмулировать работу математического сопроцессора при его отсутствии программным путём с помощью *директив компилятора* {$N+,E+}, которые заключаются в фигурные скобки и имеют отличительный признак $.

Тип *Comp* содержит 64-битные целые числа от $-2^{63}+1$ до $2^{63}-1$, но в вещественном формате. Компилятор пропустит присваивания a:= 5e-01, a:= a + a для переменной a типа Comp и выдаст результат 0 вместо 1, т.к. значение переменной не является целым числом. В выражениях тип совместим со всеми вещественными и целыми типами. Переменная типа Comp может служить вещественным аргументом функции. Применяется тип для выполнения операций с длинными целыми числами. Цепочка операторов a:= 111111; b:= 999999; Write(a*b:0:0); выведет точный целый результат 111110888889, если переменные a и b типа Comp, что не удастся сделать, если они будут иметь тип LongInt (см. пр.28, пр.40).

Новый знак (/) вводится только для операции деления вещественных чисел, причём результат операции всегда вещественный, даже когда оба операнда целые числа.

♦ Математические функции

В паскале имеется системная библиотека математических процедур и функций, которые ещё называют стандартными или встроенными. Применяя их в выражениях или для конструирования новых функций, необходимо следить за типами аргументов и возвращаемых значений. Выполняя действия с любыми числами, следует помнить основные математические факты:

нельзя делить на нуль;

не существует квадратный корень из отрицательного числа;

не существует логарифм числа, которое меньше или равно нуля.

Функция	Тип аргумента	Тип значения	Математическая запись
Odd (x)	целый	логический	2k+1, k -целое
Pi	–	вещественный	π
Abs (x)	цел / вещ	цел / вещ	$\|x\|$
Sqr (x)	цел / вещ	цел / вещ	x^2
Sqrt (x)	цел / вещ	цел / вещ	$\sqrt{x}$
Sin (x)	вещественный	вещественный	sinx
Cos (x)	вещественный	вещественный	cosx
ArcTan (x)	вещественный	вещественный	arctgx
Exp (x)	вещественный	вещественный	e^x
Ln (x)	вещественный	вещественный	lnx
Int (x)	вещественный	вещественный	[x], x≥0
Frac (x)	вещественный	вещественный	{x}, x≥0
Trunc (x)	вещественный	LongInt	
Round (x)	вещественный	LongInt	
Random	–	[0,1) - вещ	
Random(x)	Word	[0,x) - Word	

Операция возведения в степень в паскале отсутствует. Значение степени $a^x = (e^{\ln a})^x$ даёт выражение Exp(x*Ln(a)), использующее две встроенные функции Exp и Ln (3^7 = Exp(7*Ln(3)); $(-3)^7 = -3^7$ = -Exp(7*Ln(3)), т.к. Ln(-3) не существует).

Функции Int(x) и Frac(x) похожи на математические функции [x] и {x}, обозначающие соответственно целую и дробную части числа, но совпадают с ними только на множестве неотрицательных чисел. Если аргумент отрицательный, то они просто отбрасывают дробную или целую часть. [-5,8] = -6, Int (-5.8) = -5; {-5,8}= 0,2, Frac(-5.8) = -0.8 (пр.18).

Функция Trunc(x) отличается от Frac(x) типом возвращаемого значения. Round(x) – функция округления. Odd(x) возвращает логическое значение true, если аргумент x – нечётное число.

Функции Random, Random(x) и сопутствующая им процедура Randomize предназначены для получения псевдослучайных чисел (стр.54).

Целочисленные значения можно присваивать вещественным переменным, а *присваивание вещественного значения целочисленной переменной является ошибкой программирования*. Вещественные значения в последнем случае преобразовывают в целочисленные с помощью одной из предназначенных для этого встроенных функций: Trunc(x) или Round(x). Операции div и mod не могут употребляться с вещественными числами.