Б А К А Л А В Р И А Т

ФУНКЦИОНАЛЬНОЕ ПРОГРАММИРОВАНИЕ
НА ИМПЕРАТИВНЫХ И ФУНКЦИОНАЛЬНЫХ ЯЗЫКАХ

Под общей редакцией
А.Ю. АНИСИМОВА, **А.В. БАТИЩЕВА**,
А.Е. ТРУБИНА, **Ф.А. МАСТЯЕВА**

Рекомендовано
Экспертным советом УМО в системе ВО и СПО
в качестве **учебника** для укрупненной группы бакалавриата
«Информатика и вычислительная техника»
и направлений подготовки
«Прикладная информатика»,
«Информационные системы и технологии»,
«Прикладная математика и информатика»,
«Бизнес-информатика»

КНОРУС • МОСКВА • 2025

УДК [004.43+811.93](075.8)
ББК 32.973.26-018.1я73
Ф94

Рецензенты:
В.В. Беляев, гл. научный сотрудник отдела организации научных исследований и международных связей ФГАОУ ВО «Государственный университет просвещения», д-р техн. наук, проф.,
В.А. Буланов, проф. кафедры САПР МГТУ им. Н.Э. Баумана, д-р техн. наук,
Э.И. Ватутин, проф. кафедры вычислительной техники ФГБОУ ВО «Юго-Западный государственный университет», д-р техн. наук, доц.,
С.Н. Никольский, проф. кафедры КБ-4 Института кибербезопасности и цифровых технологий ФГБОУ ВО РТУ МИРЭА, д-р техн. наук

Ф94 **Функциональное программирование на императивных и функциональных языках** : учебник / коллектив авторов ; под общ. ред. А.Ю. Анисимова, А.В. Батищева, А.Е. Трубина, Ф.А. Мастяева. — Москва : КНОРУС, 2025. — 240 с. — (Бакалавриат).

ISBN 978-5-406-13936-3

Представлены теоретические основы, позволяющие изучить методику функционального программирования на императивных и функциональных языках, даны практические примеры, способствующие развитию навыка программирования посредством закрепления полученных теоретических знаний.

Первая глава посвящена исследованию базового методического материала, который необходим для понимания основных концепций функционального программирования и дальнейшего изучения материала учебника. Во второй и третьей главах рассматриваются практические аспекты функционального программирования на императивных и функциональных языках.

Соответствует ФГОС ВО последнего поколения.

Для студентов высших учебных заведений, обучающихся по направлениям «Прикладная информатика», «Информационные системы и технологии», «Прикладная математика и информатика», «Бизнес-информатика», «Информатика и вычислительная техника», аспирантов, преподавателей, слушателей системы повышения квалификации, а также читателей, интересующихся вопросами функционального программирования.

Ключевые слова: функциональные языки программирования; языки высокого уровня; Java; Python 3; PHP; C++; Haskell; F#.

УДК [004.43+811.93](075.8)
ББК 32.973.26-018.1я73

ФУНКЦИОНАЛЬНОЕ ПРОГРАММИРОВАНИЕ
на императивных и функциональных языках

Изд. № 697070. Подписано в печать 27.08.2025. Формат 60×90/16.
Гарнитура «Newton». Усл. печ. л. 15,0. Уч.-изд. л. 9,7.

ООО «Издательство «КноРус».
117218, г. Москва, ул. Кедрова, д. 14, корп. 2.
Тел.: +7 (495) 741-46-28. E-mail: welcome@knorus.ru www.knorus.ru

Отпечатано в полном соответствии с качеством предоставленных материалов в ООО «Фотоэксперт». 109316, г. Москва, Волгоградский проспект, д. 42, корп. 5, эт. 1, пом. I, ком. 6.3-23Н23Н

ISBN 978-5-406-13936-3

ОГЛАВЛЕНИЕ

АВТОРСКИЙ КОЛЛЕКТИВ

Алексахин Александр Николаевич, ORCID: 0000-0003-0692-2391, канд. пед. наук, заведующий кафедрой информационного менеджмента и информационно-коммуникационных технологий им. проф. В. В. Дика, факультет информационных технологий, Университет «Синергия», Москва.

Алехин Евгений Иванович, ORCID: 0009-0004-5123-6846, канд. физ.-мат. наук, доцент, доцент кафедры информационного менеджмента и информационно-коммуникационных технологий им. проф. В. В. Дика, факультет информационных технологий, Университет «Синергия», Москва.

Андреев Антон Валерьевич, ORCID: 0000-0002-8054-8257, ст. преподаватель кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Анисимов Александр Юрьевич, ORCID: 0000-0002-8113-4523, канд. экон. наук, доцент, зам. директора факультета информационных технологий, доцент кафедры информационного менеджмента и информационно-коммуникационных технологий им. проф. В. В. Дика, факультет информационных технологий, Университет «Синергия», Москва.

Батищев Александр Витальевич, ORCID: 0000-0003-4872-0608, канд. экон. наук, доцент, проректор по проектной деятельности, ФГБОУ ВО Министерства сельского хозяйства Российской Федерации «Российский государственный университет народного хозяйства имени В. И. Вернадского», М.О. г. Балашиха.

Горшкова Анастасия Анатольевна, ORCID: 0009-0007-4030-0103, ст. преподаватель кафедры информационного менеджмента и информационно-коммуникационных технологий им. проф. В. В. Дика, факультет информационных технологий, Университет «Синергия», Москва.

Гринева Елизавета Сергеевна, ORCID: 0000-0002-5940-2616, ст. преподаватель кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Громов Сергей Владимирович, ORCID: 0000-0002-8967-3644, канд. техн. наук, доцент кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Дорофеев Олег Васильевич, ORCID: 0000-0003-1868-0529, канд. техн. наук, доцент, доцент кафедры цифровой экономики, факультет информационных технологий, декан факультета бизнеса, Университет «Синергия», Москва.

Зайцев Алексей Иванович, ORCID: 0000-0002-8491-3338, канд. техн. наук, доцент, доцент кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Захаров Александр Викторович, ORCID: 0000-0002-9086-1124, канд. экон. наук, доцент, декан факультета информационных технологий, доцент кафедры информационного менеджмента и информационно-коммуникационных технологий им. проф. В. В. Дика, факультет информационных технологий, Университет «Синергия», Москва.

Корепанова Вероника Сергеевна, ORCID0000-0002-0047-0796, канд. техн. наук, доцент, доцент кафедры искусственного интеллекта и анализа данных, факультет информационных технологий, Университет «Синергия»; гл. специалист отдела разработки зрелых месторождений с применением нейронных сетей ООО «ЛУКОЙЛ-Инжиниринг», Москва.

Култыгин Олег Петрович, ORCID: 0000-0002-6519-1982, канд. экон. наук, доцент, доцент кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Любимов Алексей Владимирович, ORCID: 0009-0004-8776-1431, заместитель начальника отдела серверной инфраструктуры Департамента информационных технологий, Фонд пенсионного и социального страхования Российской Федерации; ст. преподаватель кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Люблинская Наталья Николаевна, ORCID: 0000-0001-8415-4530, канд. техн. наук, доцент, доцент кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Макарова Станислава Николаевна, ORCID: 0000-0003-3474-131X, канд. экон. наук, директор центра научных коммуникаций и междисциплинарных проектов, ФГБОУ ВО «Орловский государственный университет им. И. С. Тургенева», Орел; доцент кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Мастяев Филипп Алексеевич, ORCID: 0000-0002-8012-8594, ст. преподаватель кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Мекшенева Жанна Владимировна, ORCID: 0000-0002-1716-7857, канд. экон. наук, доцент, доцент кафедры информационного менед-

жмента и информационно-коммуникационных технологий им. проф. В. В. Дика, заведующий кафедрой математики, факультет информационных технологий, Университет «Синергия», Москва.

Новиков Сергей Владимирович, ORCID: 0000-0003-1055-0113, канд. техн. наук, доцент, доцент кафедры информационных систем и цифровых технологий, ФГБОУ ВО Орловский государственный университет имени И. С. Тургенева, Орел.

Ратанова Ольга Валентиновна, ORCID: 0000-0002-1887-6364, ст. преподаватель кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Ребус Наталья Анатольевна, ORCID: 0000-0002-3086-4200, ст. преподаватель кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Рыженков Денис Викторович, ORCID: 0009-0001-2734-3674, канд. техн. наук, доцент кафедры информационных систем и цифровых технологий, ФГБОУ ВО Орловский государственный университет имени И. С. Тургенева, Орел.

Селиверстов Сергей Николаевич, ORCID: 0000-0001-5524-5791, ст. преподаватель кафедры информационного менеджмента и информационно-коммуникационных технологий им. проф. В. В. Дика, факультет информационных технологий, Университет «Синергия», Москва.

Стычук Алексей Александрович, ORCID: 0000-0001-8424-5264, канд. техн. наук, доцент, доцент кафедры информационных систем и цифровых технологий, Институт приборостроения, автоматизации и информационных технологий, ФГБОУ ВО Орловский государственный университет им. И. С. Тургенева, Орел.

Терехов Сергей Васильевич, ORCID: 0000-0002-4560-3070, канд. филос. наук, доцент, главный инженер, департамент информационной безопасности, Банк России, Москва.

Терехова Лидия Анатольевна, ORCID: 0000-0002-5129-4278, канд. пед. наук, доцент кафедры информационного менеджмента и информационно-коммуникационных технологий им. проф. В. В. Дика, факультет информационных технологий, Университет «Синергия», Москва.

Токмакова Елена Николаевна, ORCID: 0000-0001-9963-2726, канд. экон. наук, доцент, доцент кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Трубин Александр Евгеньевич, ORCID: 0000-0002-7189-5679, канд. экон. наук, доцент, заведующий кафедрой цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Ужаринский Антон Юрьевич, ORCID: 0000-0002-0499-4781, канд. техн. наук, доцент, доцент кафедры информационных систем и цифро-

вых технологий, ФГБОУ ВО Орловский государственный университет им. И. С. Тургенева, Орел.

Цой Валентин Валерьевич, ORCID: 0009-0003-7407-3077, преподаватель кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

Чантурия Георгий Темурович, ORCID: 0000-0002-1887-6364, ст. преподаватель кафедры цифровой экономики, факультет информационных технологий, Университет «Синергия», Москва.

ВВЕДЕНИЕ

Функциональное программирование (аппликативное программирование) радикально отличается от императивного (процедурного) программирования, так как основной его механизм состоит в аппликации функции к аргументам. В функциональном программировании программа представляет собой выражение, которое соответствует математической функции. Функциональные языки программирования (Haskell, Lisp, Erlang, Clojure, F# и др.) поддерживают создание таких выражений, поскольку позволяют использовать мощные функциональные конструкции. Основная идея функционального программирования состоит в использовании функций без побочных эффектов и изменяемых состояний, что позволяет создавать более безопасный, модульный и расширяемый код.

В настоящее время функциональное программирование используется в различных областях: веб-разработка, научные вычисления, обработка данных, а также машинное обучение. Стоит также отметить, что функциональное программирование используется для создания параллельных и распределенных систем, так как функциональные языки программирования могут легко распараллеливаться для более быстрого и эффективного выполнения задач. Как и любая другая модель, функциональное программирование обладает своими преимуществами и недостатками. Так, к преимуществам можно отнести: простоту и модульность кода; удобство параллельного и распределенного программирования; безопасность и надежность кода; высокую производительность и эффективность; легкость тестирования и отладки кода. Недостатками же функционального программирования являются: неиспользование переменных; проблема осуществления ввода-вывода информации; более сложный для понимания синтаксис и ограничения на использование побочных эффектов, которые могут затруднять разработку сложных систем.

В результате изучения материала учебника обучающийся приобретет необходимые в дальнейшей профессиональной деятельности:

знания:

- основные понятия функционального программирования,
- отличие функционального стиля программирования от традиционного (императивного),

- методы интерпретации и компиляции программ, написанных на функциональных языках программирования,
- приемы функционального программирования на императивных языках,
- основные концепции функционального программирования,
- основные понятия лямбда-исчисления;

умения:

- использовать приемы и средства функционального программирования в современных языках программирования,
- составлять программы на функциональных языках;

навыки:

- применения функциональной парадигмы программирования для составления алгоритмов,
- анализа кода на предмет его соответствия функциональному стилю программирования,
- программирования на современных языках программирования,
- преобразования функциональных программ.

Первая глава данного учебника посвящена разбору основных концепций и понятий функционального программирования, таких как понятие функции, свойства функций, понятия «чистой» функции и побочных эффектов, понятие функции высшего порядка, лямбда-исчисление, сопоставление с образцом, замыкания, продолжения и каррирование.

Во второй главе рассматриваются принципы функционального программирования, а также приемы функционального программирования, применимые в таких императивных языках программирования, как Java, Python 3, PHP, C++.

Третья глава посвящена рассмотрению модели функционального программирования на чистом функциональном (Haskell) и мультипарадигмальном (F#) языках.

В конце каждой главы представлены вопросы, тестовые задания и задания для самостоятельного решения, которые позволяют закрепить полученные в процессе изучения материала знания и выработать навыки программирования.

В заключительной части учебника представлен список информационных источников, которые были использованы при подготовке материала учебника, а также рекомендуемая литература, позволяющая получить дополнительную информацию по функциональному программированию.

Глава 1
ОСНОВНЫЕ КОНЦЕПЦИИ ФУНКЦИОНАЛЬНОГО ПРОГРАММИРОВАНИЯ (ФП)[1]

1.1. ПОНЯТИЕ ФУНКЦИИ

В различных процессах или явлениях из области экономики, социальных наук или другой области знаний одни величины сохраняют свои значения, другие же принимают различные значения.

Переменной величиной называется такая величина, которая при выполнении некоторого комплекса условий может принимать различные значения.

Постоянной величиной называется такая величина, которая при выполнении некоторого комплекса условий сохраняет одно и то же значение.

Отметим, что выполнение комплекса условий является очень важным. Так, одна и та же величина может быть переменной или постоянной в зависимости от того, в каких условиях она рассматривается. Например, цена на хлеб (и некоторые другие продукты) в условиях рыночной экономики является величиной переменной. В условиях жесткого планирования экономики цена на хлеб может держаться на одном уровне и быть постоянной величиной.

Переменные величины обычно обозначаются последними буквами латинского алфавита ($x, y, z, u, \ldots$), а постоянные — первыми (a, b, c).

При изучении любых явлений обычно имеют дело с совокупностью переменных величин, которые связаны между собой так, что каждым значениям одних величин соответствуют значения других. Например, ясно, что:

- каждому значению цены товара соответствует определенная величина спроса;

[1] При подготовке данной главы были использованы следующие работы: Хендерсон, П. Функциональное программирование. Применение и реализация. — Москва : Мир, 1983. — 349 с.; Урма, Р.-Г., Фуско, М., Майкрофт, А. Современный язык Java. Лямбда-выражения, потоки и функциональное программирование. — Санкт-Петербург : Питер, 2020. — 592 с.; Основы функционального программирования : учебник / коллектив авторов ; под общ. ред. А. Е. Трубина, А. Ю. Анисимова, Ф. А. Мастяева. — Москва : КНОРУС, 2024. — 224 с. — (Бакалавриат).

- каждому значению объема производства соответствует определенная величина издержек;
- каждому году соответствует сумма накопившегося денежного вклада в банке;
- числу членов научного коллектива соответствует его продуктивность.

Во всех этих примерах общим является то, что каждому числовому значению одной величины сопоставляется определенное числовое значение другой.

Дадим теперь определение понятия функции, являющегося центральным понятием математического анализа, причем ограничимся случаем двух переменных величин.

Пусть даны два множества X и Y.

Определение. Соответствие f, которое каждому элементу $x \in X$ сопоставляет один элемент $y \in Y$, называется функцией и обозначается следующим образом: $y = f(x)$.

Независимой переменной, или аргументом, называют x, а y — зависимой переменной. О величинах x и y говорят, что они связаны функциональной зависимостью.

Термин «функция» происходит от латинского слова *functio* — исполнение, осуществление.

Задать функцию — значит задать три объекта:

- множество X;
- множество Y;
- правило f.

О функции $y = f(x)$ говорят, что она действует из X в Y, и пишут: $f: X \to Y$.

Например, соответствие f, изображенное на рис. 1.1, *а*, является функцией, а соответствие q, изображенное на рис. 1.1, *б*, не является функцией, так как не соблюдается условие однозначности.

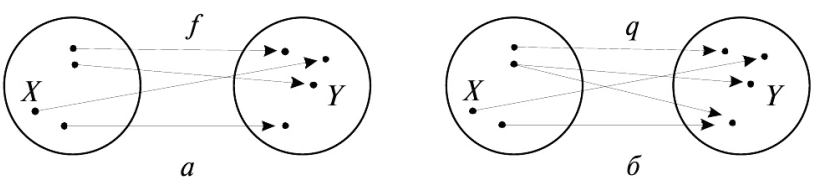

Рис. 1.1. Определение функции [1]

Область определения и область значений. Множество всех значений независимой переменной, для которых определена функция (то есть при

которых функция $y = f(x)$ вообще имеет смысл), называется *областью определения,* или областью существования этой функции, обозначается $D(f)$.

Множество всех значений функции называется *областью значений функции* и обозначается $E(f)$.

Например, областью определения функции $y = 3x^2$ является множество всех действительных чисел, а областью значений — множество $x \geq 0$; областью определения функции $y = \ln(x - 1)$ является полупрямая $x > 1$, областью значений — множество всех действительных чисел.

В социально-экономических задачах часто приходится рассматривать зависимости одной переменной от многих других. Например, национальный доход Y зависит от затрат труда L и объема производственных фондов; издержки производства зависят от материальных затрат и расходов на оплату рабочей силы. В этом случае говорят о функции нескольких переменных.

Бывают случаи, когда одной переменной соответствует несколько других. Так, некоторому значению цены на товар соответствуют определенные значения спроса и предложения, т.е. одной переменной соответствуют две другие. В таких случаях говорят о двузначной или многозначной функции (в отличие от однозначной).

Наличие функциональных зависимостей социально-экономических явлений позволяет использовать для решения экономических проблем методы математического анализа. Поэтому необходимо познакомиться со способами задания функции.

Способы задания функции. Различают три способа задания функции:

- аналитический;
- табличный;
- графический.

1. *Аналитический способ.* Если функция выражена при помощи формулы, устанавливающей, какие вычислительные операции надо произвести над x, чтобы получить y, то говорят, что она задана аналитически.

Аналитический способ удобен для выполнения математических действий над функцией и решения задач прогнозирования. Приведем пример аналитического задания функции из социально-экономической сферы (пример 1.1).

Пример 1.1

Функция Филлипса. Как и любая цена, цена труда зависит от конъюнктуры рынка. Когда на рынке труда имеет место дефицит, то рабочие могут рассчитывать на большую зарплату, и наоборот, в период существования конъюнктурной безработицы рабочим

будут платить меньше. В 1958 году профессор Лондонской школы экономики Филлипс опубликовал результаты своих исследований взаимозависимости между уровнем безработицы и изменением денежной ставки зарплаты в Великобритании в период с 1861 по 1957 г. Для первых 52 лет (1861—1913) эта зависимость выражалась уравнением:

$$y = -0{,}9 + 9{,}638x^{-1.394},$$

где y — годовой темп прироста ставки заработной платы (в процентах);
x — общий уровень безработицы (в процентах).

Это уравнение представляет аналитическое задание функции и называется *формулой Филлипса*.

К недостаткам аналитического способа задания функции можно отнести то, что он не всегда нагляден.

2. *Табличный способ.* Этот способ является наиболее простым. В одном столбце записывают значения аргумента x, а во втором — значения $f(x)$. Такой способ задания функции часто применяется в тех случаях, когда область определения состоит из конечного числа значений (таблицы цен на товары, таблицы розыгрыша лотерей и т.д.). Широко используются таблицы значений различных функций: в таблицах тригонометрических функций, логарифмов и т.п. В виде таблиц записываются результаты экспериментального исследования каких-либо процессов и явлений (пример 1.2).

Пример 1.2

Рост числа научных изданий y, начиная с 1750 г., с интервалом в 50 лет, в зависимости от года x, выглядит (округленно) следующим образом (см. табл.) [1].

Таблица

Динамика числа научных изданий

Период	1750 г.	1800 г.	1850 г.	1900 г.	1950 г.
Число научных изданий	10	100	1 000	10 000	100 000

К недостатку табличного способа можно отнести то, что представление о функциональной зависимости здесь не является полным, так как невозможно поместить в таблице все значения аргумента.

3. *Графический способ.* Аналитический и табличный способы задания функции страдают отсутствием наглядности. Графический способ не имеет такого недостатка. Графическим способом называется такой способ задания функции $y = f(x)$, при котором соответствие между аргументом x и функцией y устанавливается с помощью графика.

Графиком функции $y = f(x)$ называется множество всех точек плоскости с координатами $(x, f(x))$, т.е. таких, координаты которых обращают выражение $y = f(x)$ в тождество (пример 1.3).

Пример 1.3

Кривая Филлипса. Как известно, аналитическая зависимость между годовым темпом прироста ставки заработной платы y (в %) и общим уровнем безработицы x (в %) выражается формулой: $y = -0{,}9 + 9{,}638x^{-1.394}$.

Эта формула не дает наглядного представления о функции. График же этой функции, называемый кривой Филлипса и изображенный на рис. 1.2, позволяет как бы увидеть соответствующую зависимость.

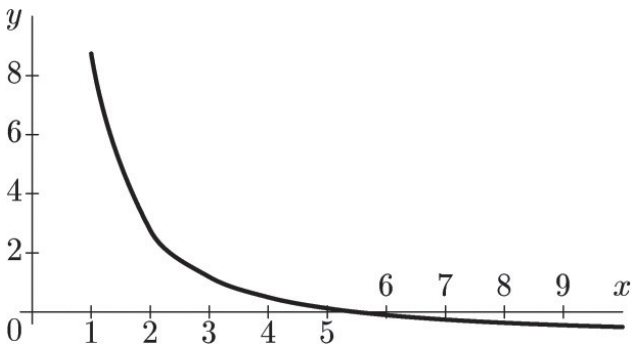

Рис. 1.2. Кривая Филлипса

Далее разберем основные свойства математических функций.

Основные свойства функций. К основным свойствам функции $y = f(x)$ относятся:

- область определения $D(f)$;
- область значений $E(f)$;
- четность, нечетность;
- монотонность;
- ограниченность;
- периодичность.

Первые два свойства функции уже были определены ранее, в начале этого подпараграфа, а остальные свойства рассматриваются далее.

Четность и нечетность. Функция $y = f(x)$ называется *четной*, если для любых значений x из области определения $f(-x) = f(x)$, и *нечетной*, если $f(-x) = -f(x)$. В противном случае функция $y = f(x)$ называется функцией общего вида (пример 1.4).

Пример 1.4

1. Функция $y = x^n$ при четном n является четной функцией, так как $f(-x) = (-x)^n = x^n = f(x)$. Заметим, что отсюда и произошло само название четной функции.
2. Функция $y = x^n$ с нечетным показателем степени n является нечетной $f(-x) = (-x)^n = -x^n = -f(x)$. Отсюда происходит название нечетной функции.
3. Функция $y = x + x^2$ является функцией общего вида. Действительно, $f(-x) = (-x) + (-x)^2 = -x + x^2 \neq f(x)$ и $f(-x) \neq -f(x)$.

График четной функции симметричен относительно оси ординат (например, график функции $y = x^2$), а график нечетной функции симметричен относительно начала координат (например, график функции $y = x^3$). Поэтому для четной функции достаточно строить лишь правую половину графика ($x \geq 0$); левая половина его ($x \leq 0$) является зеркальным отражением правой относительно оси O_y. Чтобы построить график нечетной функции, достаточно изобразить правую половину его ($x \geq 0$); левая половина графика ($x \leq 0$) получается в результате поворота правой на 180°.

Монотонность. Пусть функция $y = f(x)$ определена на множестве D, если для любых значений аргумента $x_1, x_2 \in (a; b)$ таких, что $x_1 < x_2$, выполняется неравенство:

$f(x_1) < f(x_2)$, то функция называется возрастающей;

$f(x_1) \leq f(x_2)$, то функция называется *неубывающей*;

$f(x_1) > f(x_2)$, то функция называется *убывающей*;

$f(x_1) \geq f(x_2)$, то функция называется невозрастающей (рис. 1.3).

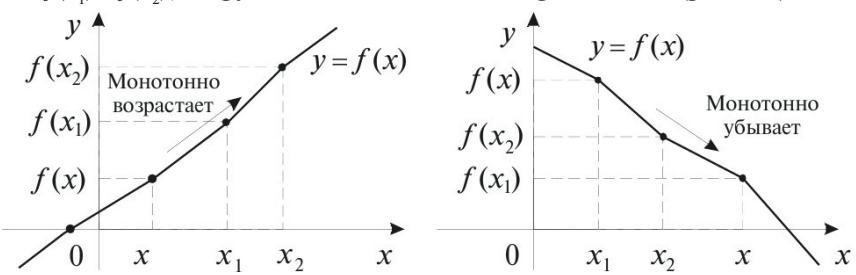

Рис. 1.3. Графики монотонных функций

Возрастающие, убывающие, невозрастающие, неубывающие функции называются монотонными. Интервалы, в которых функция монотонна, называют интервалами монотонности.

Функция $y = x^2$ строго убывает при $x \in [-\infty; 0)$ и строго возрастает при $x \in [0; +\infty)$.

Функция $y = 0,1^x$ является строго убывающей на всей действительной оси.

Ограниченность. Функция называется ограниченной на промежутке X, если существует такое положительное число $M > 0$, что $|f(x)| < M$ для любого $x \in X$ (см. примеры 1.5—1.6).

Пример 1.5

Функция $y = \sin x$ ограничена на всей числовой оси, так как $|\sin x| \leq 1$ для любого $x \in X$. График функции представлен на рис. 1.4.

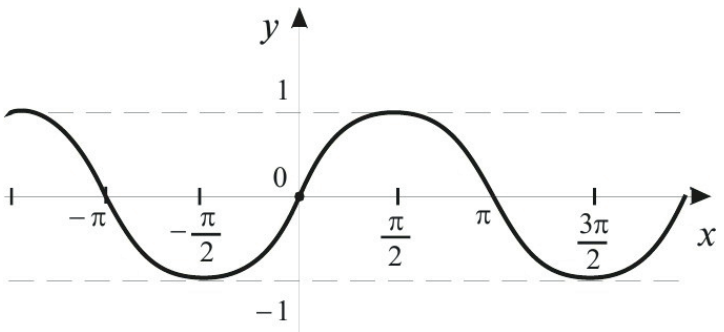

Рис. 1.4. График функции $y = \sin(x)$

Пример 1.6

Функция $y = x^3$ не является ограниченной на всей действительной оси, поскольку не существует такого положительного числа M > 0, $|x^3| < M$, что для любого $x \in R$.

Периодичность. Функция $y = f(x)$ называется периодической, если существует положительное число T такое, что $f(x + T) = f(x)$. Наименьшее число с таким свойством называется *периодом* функции.

Для построения графика периодической функции достаточно изобразить его на отрезке, длина которого равна периоду (основная область), а затем построить периодическое продолжение графика, повторяя график, нарисованный в основной области (пример 1.7).

Пример 1.7

Функция $y = tg\, x$ периодическая, ее период $T = \pi n$, $n \in Z$, $tg\,(x + \pi n) = tg\, x$. График функции представлен на рис. 1.5.

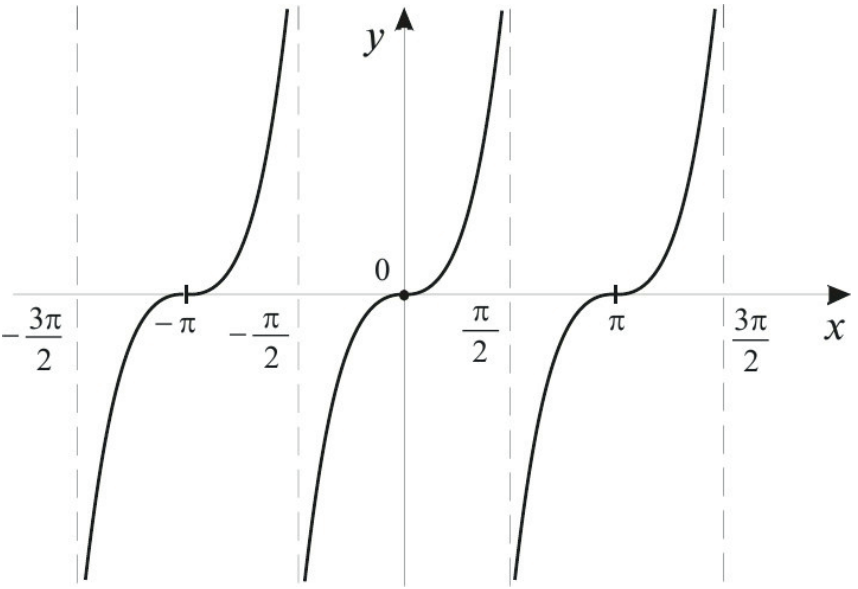

Рис. 1.5. График функции $y = tg(x)$

Функциональный стиль программирования. Функции в программировании имеют важное значение. Первые функции появились еще задолго до появления персональных компьютеров — в 60-х годах XX в. Функции в контексте программирования — это возможность писать код быстрее и делать программы читабельнее и короче.

С другой стороны, сейчас очень сложно представить современный язык программирования без использования функций. У функций есть даже собственное направление в разработке — функциональное программирование, которое предполагает применение функций по особым принципам.

В функциональном программировании программа описывается с использованием *математических функций.* Функциональная программа имеет сравнительно простую форму: последовательность определений функций, за которой следует последовательность вызовов этих функций.

Функция в программировании — это часть программного кода, который несет в себе набор инструкций или команд, решающих конкретную задачу. В программирование их ввели для того, чтобы сократить количество строк кода. Чтобы не описывать какую-то задачу десятками строк кода, можно использовать функцию, которая опишет задачу в одну строку.

К функциям можно обращаться из разных мест программы, а они в ответ возвращают какие-то значения. За счет этой возможности один и тот же алгоритм действий не нужно прописывать в разных частях кода, потому что достаточно будет только написать имя функции.

Таким образом, в языках программирования при работе с функциями можно проследить следующий процесс.

1. Функция располагается в определенной части программы, где нужно вызвать действия определенного алгоритма.
2. При обработке кода обработчик сталкивается с функцией и возвращается в ту часть программы, где прописан алгоритм.
3. Он выполняет функцию, учитывая аргументы, которые указаны.
4. После выполнения нашей функции обработчик «берет» или «не берет» результат выполнения функции и возвращается к той строке кода, которая вызвала функцию, т.е. к той строке, где написано имя функции с аргументами.
5. После возвращения обработчик переходит на следующие строки кода.

Важно понимать, что в каждом языке программирования есть два основных вида функций:

- встроенные — это те функции, которые уже находятся внутри языка, остается только пользоваться ими или не пользоваться;
- пользовательские — это те функции, которые пользователь создает самостоятельно.

Встроенные функции весьма разнообразны и изучаются в контексте конкретного языка программирования. Пользовательские функции обычно состоят из трех блоков:

1) название;
2) набор аргументов, который указывается в круглых скобках;
3) тело функции, которое указывается в фигурных скобках.

Функции могут быть простыми, а могут быть вложенными, когда одна функция находится в теле другой функции. Вычисления начинаются с вызова некоторой функции, которая в свою очередь вызывает другие функции и т.д. Каждый вызов возвращает некоторое значение в вызывающую функцию, вычисление которой после этого продолжается. Этот процесс повторяется до тех пор, пока запустившая вычисления функция не вернет конечный результат. Например, используя функцию *макс2* для нахождения наибольшего из двух чисел, наибольшее из четырех чисел можно определить так: *макс2*(*макс2*(*a*,*b*), *макс2*(*c*,*d*)).

Функции только принимают входные данные и возвращают некоторый результат. Взаимодействие между разными функциями возможно только через вызовы во время выполнения, при этом порядок вычисления подвы-

ражений не имеет значения. Результат выполнения функциональной программы — значение выражения, которое определяется в терминах базовых и (или) определенных пользователем функций.

При работе с функциональной парадигмой важно постоянно думать в терминах функций. Составлять программы с помощью функций можно и на императивных языках, однако функциональный подход имеет существенные отличия.

Функциональная парадигма не должна рассматриваться как замена императивному программированию. Она лишь представляет другой подход к разработке программ. Для решения некоторого класса задач такой подход является более эффективным.

В последнее время наблюдается рост интереса к функциональному программированию. Все больше людей интересуются методологией и технологией построения программ с использованием функциональной парадигмы. Эта парадигма наиболее приспособлена и применяется для создания систем искусственного интеллекта, разработки графических интерфейсов, создания графических приложений реального времени, разработки систем параллельных вычислений, построения компиляторов. Кроме того, удобство функционального программирования отмечают специалисты, работающие в финансовой и научной областях, а также занимающиеся техническими расчетами.

1.2. ПОБОЧНЫЕ ЭФФЕКТЫ

На современном этапе разработки программного кода все еще остро возникает вопрос о побочных эффектах. Что такое побочные эффекты в программировании? Давайте попробуем подробнее разобраться, что это такое и в каких случаях они себя проявляют в процессе рождения программного продукта.

Побочные эффекты — это действия, которые не были запланированы разработчиком программного кода, но которые могут быть вызваны при его использовании.

Например, если требуется написать код, который использует объект класса, то можно с уверенностью сказать, что при запуске этого кода объект будет создан, так как его конструктор запустит эту работу. Однако если вы забыли создать этот объект, то в итоге вы получите ошибку вида «Unassigned local variable».

Побочные эффекты в функциональном программировании — это любые действия работающей программы, изменяющие среду ее выполнения.

Побочные эффекты, к примеру, могут себя проявить при работе:

- со значениями глобальных переменных;
- с содержимым файлов и баз данных;