ВикипедиЯ

Рекурсия

Материал из Википедии — свободной энциклопедии

Реку́рсия — определение, описание, изображение какого-либо объекта или процесса внутри самого этого объекта или процесса, то есть ситуация, когда объект является частью самого себя. Термин «рекурсия» используется в различных специальных областях знаний — от лингвистики до логики, но наиболее широкое применение находит в математике и информатике.

Рекурсивное изображение экрана

Визуальная форма рекурсии страницы Википедии

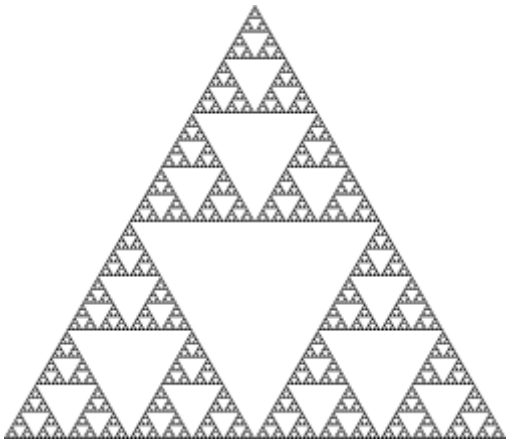
Содержание

В математике

В математике рекурсия имеет отношение к методу определения функций и числовых рядов: *рекурсивно заданная* функция определяет своё значение через обращение к себе самой с другими аргументами. При этом возможно два варианта:

- Конечная рекурсивная функция. Она задаётся таким образом, чтобы для любого конечного аргумента за конечное число рекурсивных обращений привести к одному из отдельно определённых частных случаев, вычисляемых без рекурсии. Классический пример: рекурсивно-определённый факториал целого неотрицательного числа: $n! = \begin{cases} n \cdot (n-1)!, & n > 0 \\ 1, & n = 0 \end{cases}$.
  Здесь каждое следующее рекурсивное обращение делается с аргументом, меньшим на единицу. Поскольку **n**, по определению, целое неотрицательное число, через **n** рекурсивных обращений вычисление функции гарантированно придёт к частному случаю $0! = 1$, на котором рекурсия прекратится. Таким образом, несмотря на рекурсивность определения, вычисление функции для любого аргумента из области определения окажется конечным.
- Бесконечная рекурсивная функция. Она задаётся в виде обращения к самой себе во всех случаях (по крайней мере, для некоторых из аргументов). Подобным образом могут задаваться бесконечные ряды, бесконечные непрерывные дроби и так далее. Практическое вычисление точного значения здесь, естественно, невозможно, поскольку потребует бесконечного времени. Требуемый результат находится аналитическими методами. Тем не менее, если речь идёт не о получении абсолютно точного значения, а о вычислении заданного приближения искомого значения, то тут в некоторых случаях возможно прямое использование рекурсивного определения: вычисления по нему ведутся до тех пор, пока необходимая точность не будет достигнута. Примером может служить разложение в непрерывную дробь числа *e*:

Визуальная форма рекурсии (эффект Дросте)


Треугольник Серпинского

$$e = 2 + \cfrac{2}{2 + \cfrac{3}{3 + \cfrac{4}{4 + \ldots}}} = 2 + f(2)$$, где

$$f(n) = \cfrac{n}{n + f(n+1)}$$

Прямой расчёт по приведённой формуле вызовет бесконечную рекурсию, но можно доказать, что значение f(n) при возрастании n стремится к единице (поэтому, несмотря на бесконечность ряда, значение числа Эйлера конечно). Для приближённого вычисления значения e достаточно искусственно ограничить глубину рекурсии некоторым наперёд заданным числом и по достижении его использовать вместо $f(n)$ единицу.

Другим примером рекурсии в математике является числовая последовательность, заданная рекуррентной формулой, когда каждый следующий член последовательности вычисляется как результат функции от n предыдущих членов. Таким образом с помощью конечного выражения (представляющего собой совокупность рекуррентной формулы и набора значений для первых n членов ряда) может даваться определение бесконечной последовательности.

С рекурсией тесно связана математическая индукция: она является естественным способом доказательства свойств функций на натуральных числах, рекурсивно заданных через свои меньшие значения.

В математике отдельно рассматривается класс так называемых «примитивно рекурсивных» функций. Определение примитивно рекурсивной функции также рекурсивно, оно задаёт набор примитивных функций и набор правил; функция является примитивно рекурсивной, если она может быть представлена как комбинация примитивно рекурсивных функций, образованная по этим правилам.

Примеры рекурсии в математике:

- Метод Гаусса — Жордана для решения систем линейных алгебраических уравнений является рекурсивным.
- Уже упоминавшийся факториал целого неотрицательного числа.
- Числа Фибоначчи определяются с помощью рекуррентного соотношения:

  > Первое и второе числа Фибоначчи равны 1
  > Для $n > 2$, n-е число Фибоначчи равно сумме $(n-1)$-го и $(n-2)$-го чисел Фибоначчи

- Практически все геометрические фракталы задаются в форме бесконечной рекурсии (например, треугольник Серпинского).
- Стандартный пример вычислимой рекурсивной функции, не являющейся примитивно рекурсивной — функция Аккермана: для неотрицательных целых чисел m и n следующим образом:

$$A(m,\ n) = \begin{cases} n+1, & m = 0; \\ A(m-1,\ 1), & m > 0,\ n = 0; \\ A(m-1,\ A(m,\ n-1)), & m > 0,\ n > 0. \end{cases}$$

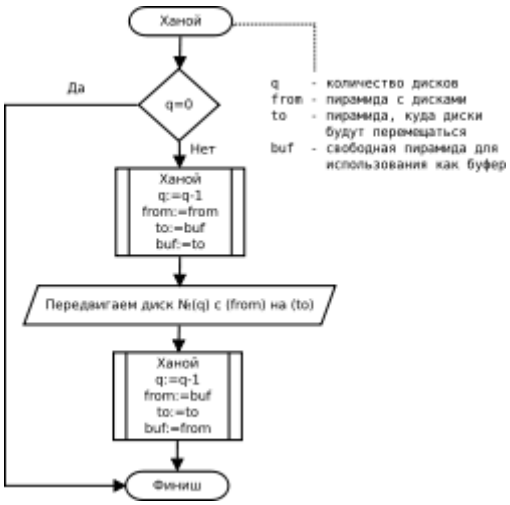
В программировании

Функции

В программировании рекурсия — вызов функции (процедуры) из неё же самой, непосредственно (*простая рекурсия*) или через другие функции (*сложная* или *косвенная рекурсия*), например, функция A вызывает функцию B, а функция B — функцию A. Количество вложенных вызовов функции или процедуры называется глубиной рекурсии. Рекурсивная программа позволяет описать повторяющееся или даже потенциально бесконечное вычисление, причём без явных повторений частей программы и использования циклов.

Структурно рекурсивная функция на верхнем уровне всегда представляет собой команду ветвления (выбор одной из двух или более альтернатив в зависимости от условия (условий), которое в данном случае уместно назвать «условием прекращения рекурсии»), имеющую две или более

альтернативные ветви, из которых хотя бы одна является *рекурсивной* и хотя бы одна — *терминальной*. Рекурсивная ветвь выполняется, когда условие прекращения рекурсии ложно, и содержит хотя бы один рекурсивный вызов — прямой или опосредованный вызов функцией самой себя. Терминальная ветвь выполняется, когда условие прекращения рекурсии истинно; она возвращает некоторое значение, не выполняя рекурсивного вызова. Правильно написанная рекурсивная функция должна гарантировать, что через конечное число рекурсивных вызовов будет достигнуто выполнение условия прекращения рекурсии, в результате чего цепочка последовательных рекурсивных вызовов прервётся и выполнится возврат.



Блок схема рекурсивного алгоритма решения Ханойской башни.

Помимо функций, выполняющих один рекурсивный вызов в каждой рекурсивной ветви, бывают случаи «параллельной рекурсии», когда на одной рекурсивной ветви делается два или более рекурсивных вызова. Параллельная рекурсия типична при обработке сложных структур данных, таких как деревья. Простейший пример параллельно-рекурсивной функции — вычисление ряда Фибоначчи, где для получения значения n-го члена необходимо вычислить (n-1)-й и (n-2)-й.

Реализация рекурсивных вызовов функций в практически применяемых языках и средах программирования, как правило, опирается на механизм стека вызовов — адрес возврата и локальные переменные функции записываются в стек, благодаря чему каждый следующий рекурсивный вызов этой функции пользуется своим набором локальных переменных и за счёт этого работает корректно. Оборотной стороной этого довольно простого по структуре механизма является то, что на каждый рекурсивный вызов требуется некоторое количество оперативной памяти компьютера, и при чрезмерно большой глубине рекурсии может наступить переполнение стека вызовов.

Вопрос о желательности использования рекурсивных функций в программировании неоднозначен: с одной стороны, рекурсивная форма может быть структурно проще и нагляднее, в особенности, когда сам реализуемый алгоритм по сути рекурсивен. Кроме того, в некоторых декларативных или чисто функциональных языках (таких как Пролог или Haskell) просто нет синтаксических средств для организации циклов, и рекурсия в них — единственный доступный механизм организации повторяющихся вычислений. С другой стороны, обычно рекомендуется избегать рекурсивных программ, которые приводят (или в некоторых условиях могут приводить) к слишком большой глубине рекурсии. Так, широко распространённый в учебной литературе пример рекурсивного вычисления факториала является, скорее, примером того, как *не надо* применять рекурсию, так как приводит к достаточно большой глубине рекурсии и имеет очевидную реализацию в виде обычного циклического алгоритма.

Имеется специальный тип рекурсии, называемый «хвостовой рекурсией» (структура рекурсивного алгоритма такова, что рекурсивный вызов является последней выполняемой операцией в функции, а его результат непосредственно возвращается в качестве результата функции). Интерпретаторы и компиляторы функциональных языков программирования, поддерживающие оптимизацию кода (исходного или исполняемого), автоматически преобразуют хвостовую рекурсию к итерации, благодаря чему обеспечивается выполнение алгоритмов с хвостовой рекурсией в ограниченном объёме памяти. Такие рекурсивные вычисления, даже если они формально бесконечны (например, когда с помощью рекурсии организуется работа командного интерпретатора, принимающего команды пользователя), никогда не приводят к исчерпанию памяти. Однако далеко не всегда стандарты языков программирования чётко определяют, каким именно условиям должна

удовлетворять рекурсивная функция, чтобы транслятор гарантированно преобразовал её в итерацию. Одно из редких исключений — язык Scheme (диалект языка Lisp), описание которого содержит все необходимые сведения.

Теоретически, любую рекурсивную функцию можно заменить циклом и стеком. Однако такая модификация, как правило, бессмысленна, так как приводит лишь к замене автоматического сохранения контекста в стеке вызовов на ручное выполнение тех же операций с тем же или большим расходом памяти. Исключением может быть ситуация, когда рекурсивный алгоритм приходится моделировать на языке, в котором рекурсия запрещена.

Доказательство корректности программ

В отличие от явно-циклических программ, для доказательства корректности рекурсивных нет необходимости искусственно вводить инвариант. Аналитическое доказательство корректности рекурсивной функции сводится к методу математической индукции, то есть к доказательству следующих утверждений:

1. **Корректность рекурсивного обращения.** Доказывается, что результат, вычисляемый в любой рекурсивной ветви функции, будет верным при условии, что параметры функции заданы корректно и соответствующие рекурсивные вызовы вернут верный результат.
2. **Корректность всех терминальных ветвей.** Доказывается, что все терминальные ветви возвращают верные значения. Как правило, это доказательство тривиально, так как терминальные ветви обычно никаких вычислений не содержат.
3. **Достижимость терминальной ветви для любого корректного набора параметров после конечного числа рекурсивных вызовов.** Доказывается, что изменение параметров вызова функции, которое производится при рекурсивном обращении, через конечное число рекурсивных вызовов приведёт к одному из наборов параметров, для которых существует терминальная ветвь.

Из суммы первого и второго утверждений следует, что в случае достижения терминальной ветви (а это значит — во всех случаях, когда вычисление функции окажется конечным) функция вернёт правильный результат. Третье положение доказывает, что конечным будет любое вычисление. Следовательно, любой вызов функции с корректными параметрами вернёт правильный результат (с очевидной «технической» оговоркой — если глубина рекурсии не окажется настолько большой, что вызовет переполнение памяти).

Данные

Рекурсивное определение данных возникает тогда, когда структура данных (запись, объект) содержит вложенный объект, структурно аналогичный самому себе или (что бывает чаще) ссылку на такой же объект. Преимущество рекурсивного определения объекта заключается в том, что такое конечное определение способно описать потенциально бесконечную структуру данных. Подобные структуры используются при описании списков и графов. Пример описания списка (C):

```c
struct element_of_list
{
  struct element_of_list *next; /* указатель на следующий элемент того же типа */
  int data; /* некие данные */
};
```

Поскольку бесконечное число вложенных объектов невозможно разместить в конечной памяти, в реальности такие рекурсивно-определённые структуры всегда конечны. В заключительных (терминальных) ячейках обычно записываются пустые указатели, являющиеся одновременно флагами, указывающими программе, обрабатывающей структуру, что достигнут её конец.

Рекурсивная структура данных зачастую обуславливает применение рекурсии для обработки этих данных. В последние годы стала популярной концепция так называемых «ленивых вычислений», согласно которой данные, обрабатываемые программой, вычисляются лишь тогда, когда в них возникает необходимость. Реализация этой концепции привела к появлению в некоторых языках (Haskell, Python) возможности описывать потенциально-бесконечные, в том числе рекурсивные последовательности без явного ограничения на порождение объектов и свободно работать с ними.

В физике

Классическим примером бесконечной рекурсии являются два поставленные друг напротив друга зеркала: в них образуются два коридора из уменьшающихся отражений зеркал.

Другим примером бесконечной рекурсии является эффект самовозбуждения (положительной обратной связи) у электронных схем усиления, когда сигнал с выхода попадает на вход, усиливается, снова попадает на вход схемы и снова усиливается. Усилители, для которых такой режим работы является штатным, называются автогенераторы.

В лингвистике

В лингвистике рекурсией называют способность языка порождать вложенные предложения и конструкции. Базовое предложение «*кошка съела мышь*» может быть за счёт рекурсии расширено как «*Ваня догадался, что кошка съела мышь*», далее как «*Катя знает, что Ваня догадался, что кошка съела мышь*» и так далее. Рекурсия считается одной из лингвистических универсалий, то есть свойственна любому естественному языку. Однако в последнее время активно обсуждается возможное отсутствие рекурсии в одном из языков Амазонии — пираха, которое отмечает лингвист Дэниел Эверетт[1].

В культуре

- Большая часть шуток о рекурсии касается бесконечной рекурсии, в которой нет условия выхода, например, известно высказывание: «чтобы понять рекурсию, нужно сначала понять рекурсию».
  - Весьма популярна шутка о рекурсии, напоминающая словарную статью:

> **рекурсия**
> см. рекурсия

- Тема рекурсии присутствует во многих рассказах и очерках аргентинского писателя Хорхе Луиса Борхеса.
- Несколько рассказов Станислава Лема посвящены (возможным) казусам при бесконечной рекурсии:

- рассказ из «Кибериады» о разумной машине, которая обладала достаточным умом и ленью, чтобы для решения поставленной задачи построить себе подобную и поручить решение ей (итогом стала бесконечная рекурсия, когда каждая новая машина строила себе подобную и передавала задание ей);
- рассказ про Ийона Тихого «Путешествие четырнадцатое» из «Звёздных дневников Ийона Тихого», в котором герой последовательно переходит от статьи о сепульках к статье о сепуляции, оттуда к статье о сепулькариях, в которой снова стоит отсылка к статье «сепульки»:

> Нашёл следующие краткие сведения:
> «СЕПУЛЬКИ — важный элемент цивилизации ардритов (см.) с планеты Энтеропия (см.). См. СЕПУЛЬКАРИИ».
> Я последовал этому совету и прочёл:
> «СЕПУЛЬКАРИИ — устройства для сепуления (см.)».
> Я поискал «Сепуление»; там значилось:
> «СЕПУЛЕНИЕ — занятие ардритов (см.) с планеты Энтеропия (см.). См. СЕПУЛЬКИ».
>
> *Лем С. «Звёздные дневники Ийона Тихого. Путешествие четырнадцатое.»*

- Рекурсивные акронимы: GNU (GNU Not Unix), PHP (PHP: Hypertext Preprocessor), WINE (*Wine Is Not an Emulator*) и т. д.
- Герб Российской Федерации является рекурсивно-определённым графическим объектом: в правой лапе изображённого на нём двуглавого орла зажат скипетр, который венчается уменьшенной копией герба. Так как на этом гербе в правой лапе орла также находится скипетр, получается бесконечная рекурсия.

Рекурсивный герб Российской Федерации

См. также

- Математическая индукция
- Корекурсия
- Рекуррентная формула
- Возвратная последовательность
- Рефлексивное отношение
- Порочный круг
- Фрактал
- Докучная сказка

Примечания

1. О рекурсии в лингвистике, её разновидностях и наиболее характерных проявлениях в русском языке рассказано в статье Е. А. Лодатко «Рекурсивные лингвистические структуры» (http://www.relga.ru/Environ/WebObjects/tgu-www.woa/wa/Main?textid=2148&level1=main&level2=articles)

Ссылки

- Статья в Энциклопедическом Фонде // Рекурсия (http://www.russika.ru/ef.php?s=4585)

Источник — https://ru.wikipedia.org/w/index.php?title=Рекурсия&oldid=111340016
