Main page
Contents
Current events
Random article
About Wikipedia
Contact us
Donate

Contribute
Help
Community portal
Recent changes
Upload file

Tools
What links here
Related changes
Special pages
Permanent link
Page information
Cite this page
Wikidata item

Print/export
Download as PDF
Printable version

Article Talk Read View source View history Search Wikipedia

Recursion

From Wikipedia, the free encyclopedia

For other uses, see Recursion (disambiguation).

This article **needs additional citations for verification**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed.
Find sources: "Recursion" – news · newspapers · books · scholar · JSTOR *(June 2012)* *(Learn how and when to remove this template message)*

Recursion (adjective: *recursive*) occurs when a thing is defined in terms of itself or of its type. Recursion is used in a variety of disciplines ranging from linguistics to logic. The most common application of recursion is in mathematics and computer science, where a function being defined is applied within its own definition. While this apparently defines an infinite number of instances (function values), it is often done in such a way that no infinite loop or infinite chain of references can occur.

A visual form of recursion known as the *Droste effect*. The woman in this image holds an object that contains a

Contents [hide]

In other projects

Wikimedia Commons

Languages

中文
Français
Español
Deutsch
العربية
हिन्दी
Русский
Tagalog
한국어

45 more

Edit links

smaller image of her holding an identical object, which in turn contains a smaller image of herself holding an identical object, and so forth. 1904 Droste cocoa tin, designed by Jan Misset

Formal definitions

In mathematics and computer science, a class of objects or methods exhibits recursive behavior when it can be defined by two properties:

- A simple *base case* (or cases) — a terminating scenario that does not use recursion to produce an answer
- A *recursive step* — a set of rules that reduces all other cases toward the base case

For example, the following is a recursive definition of a person's *ancestor*. One's ancestor is either:

- One's parent (*base case*), *or*
- One's parent's ancestor (*recursive step*).

The Fibonacci sequence is another classic example of recursion:

$$\mathrm{Fib}(0) = 0 \text{ as base case 1,}$$

$$\mathrm{Fib}(1) = 1 \text{ as base case 2,}$$

Ouroboros, an ancient symbol depicting a serpent or dragon eating its own tail.

$$\text{For all integers } n > 1,\ \ \mathrm{Fib}(n) := \mathrm{Fib}(n-1) + \mathrm{Fib}(n-2).$$

Many mathematical axioms are based upon recursive rules. For example, the formal definition of the natural numbers by the Peano axioms can be described as: "Zero is a natural number, and each natural number has a successor, which is also a natural number."[1] By this base case and recursive rule, one can generate the set of all natural numbers.

Other recursively defined mathematical objects include factorials, functions (e.g., recurrence relations), sets (e.g., Cantor ternary set), and fractals.[2]

There are various more tongue-in-cheek definitions of recursion; see recursive humor.

Informal definition

Recently refreshed sourdough, bubbling through fermentation: the recipe calls for some sourdough left over from the last time the same recipe was made.

Recursion is the process a procedure goes through when one of the steps of the procedure involves invoking the procedure itself. A procedure that goes through recursion is said to be 'recursive'.[3]

To understand recursion, one must recognize the distinction between a procedure and the running of a procedure. A procedure is a set of steps based on a set of rules, while the running of a procedure involves actually following the rules and performing the steps.

Recursion is related to, but not the same as, a reference within the specification of a procedure to the execution of some other procedure.

When a procedure is defined as such, this immediately creates the possibility of an endless loop; recursion can only be properly used in a definition if the step in question is skipped in certain cases so that the procedure can complete.

But even if it is properly defined, a recursive procedure is not easy for humans to perform, as it requires distinguishing the new from the old, partially executed invocation of the procedure; this requires some administration as to how far various simultaneous instances of the procedures have progressed. For this reason, recursive definitions are very rare in everyday situations.

In language

Linguist Noam Chomsky, among many others, has argued that the lack of an upper bound on the number of grammatical sentences in a language, and the lack of an upper bound on grammatical sentence length (beyond practical constraints such as the time available to utter one), can be explained as the consequence of recursion in natural language.[4][5]

This can be understood in terms of a recursive definition of a syntactic category, such as a sentence. A sentence can have a structure in which what follows the verb is another sentence: *Dorothy thinks witches are dangerous*, in which the sentence *witches are dangerous* occurs in the larger one. So a sentence can be defined recursively (very roughly) as something with a structure that includes a noun phrase, a verb, and optionally another sentence. This is really just a special case of the mathematical definition of recursion.

This provides a way of understanding the creativity of language—the unbounded number of grammatical sentences—because it immediately predicts that sentences can be of arbitrary length: *Dorothy thinks that Toto suspects that Tin Man said that...*. There are many structures apart from sentences that can be defined recursively, and therefore many ways in which a sentence can embed instances of one category inside another.[6] Over the years, languages in general have proved amenable to this kind of analysis.

Recently, however, the generally accepted idea that recursion is an essential property of human language has been challenged by Daniel Everett on the basis of his claims about the Pirahã language. Andrew Nevins, David Pesetsky and Cilene Rodrigues are among many who have argued against this.[7] Literary self-reference can in any case be argued to be different in kind from mathematical or logical recursion.[8]

Recursion plays a crucial role not only in syntax, but also in natural language semantics. The word *and*, for example, can be construed as a function that can apply to sentence meanings to create new sentences, and likewise for noun phrase meanings, verb phrase meanings, and others. It can also apply to intransitive verbs, transitive verbs, or ditransitive verbs. In order to provide a single denotation for it that is suitably flexible, *and* is typically defined so that it can take any of these different types of meanings as arguments. This can be done by defining it for a simple case in which it combines sentences, and then defining the other cases recursively in terms of the simple one.[9]

A recursive grammar is a formal grammar that contains recursive production rules.[10]

Recursive humor

Recursion is sometimes used humorously in computer science, programming, philosophy, or mathematics textbooks, generally by giving a circular definition or self-reference, in which the putative recursive step does not get closer to a base case, but instead leads to an infinite regress. It is not unusual for such books to include a joke entry in their glossary along the lines of:

> Recursion, *see Recursion*.[11]

A variation is found on page 269 in the index of some editions of Brian Kernighan and Dennis Ritchie's book *The C Programming Language*; the index entry recursively references itself ("recursion 86, 139, 141, 182, 202, 269"). Early versions of this joke can be found in "Let's talk Lisp" by Laurent Siklóssy (published by Prentice Hall PTR on December 1, 1975 with a copyright date of 1976) and in "Software Tools" by Kernighan and Plauger (published by Addison-Wesley Professional on January 11, 1976). The joke also appears in "The UNIX Programming Environment" by Kernighan and Pike. It did not appear in the first edition of *The C Programming Language*. The joke is part of the Functional programming folklore and was already widespread in the functional programming community before the publication of the aforementioned books.

Another joke is that "To understand recursion, you must understand recursion."[11] In the English-language version of the Google web search engine, when a search for "recursion" is made, the site suggests "Did you mean: *recursion*."[12] An alternative form is the following, from Andrew Plotkin: *"If you already know what recursion is, just remember the answer. Otherwise, find someone who is standing closer to Douglas Hofstadter than you are; then ask him or her what recursion is."*

Recursive acronyms are other examples of recursive humor. PHP, for example, stands for "PHP Hypertext Preprocessor", WINE stands for "WINE Is Not an Emulator" GNU stands for "GNU's not Unix", and SPARQL denotes the "SPARQL Protocol and RDF Query Language".

In mathematics

Recursively defined sets

Main article: Recursive definition

Example: the natural numbers

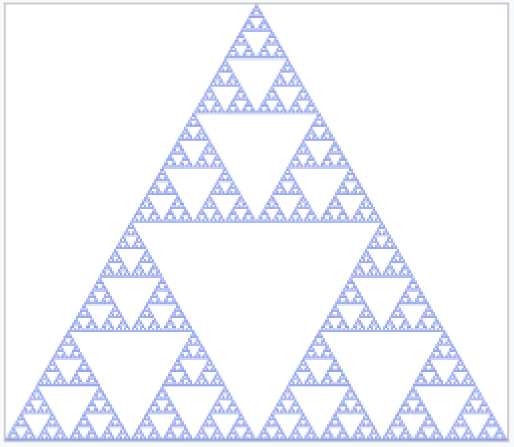
See also: Closure (mathematics)

The canonical example of a recursively defined set is given by the natural numbers:

0 is in $\mathbb{N}$

if n is in $\mathbb{N}$, then $n + 1$ is in $\mathbb{N}$

The set of natural numbers is the smallest set satisfying the previous two properties.



The Sierpinski triangle—a confined recursion of triangles that form a fractal

In mathematical logic, the Peano axioms (or Peano postulates or Dedekind–Peano axioms), are axioms for the natural numbers presented in the 19th century by the German mathematician Richard Dedekind and by the Italian mathematician Giuseppe Peano. The

Peano Axioms define the natural numbers referring to a recursive successor function and addition and multiplication as recursive functions.

Example: Proof procedure

Another interesting example is the set of all "provable" propositions in an axiomatic system that are defined in terms of a proof procedure which is inductively (or recursively) defined as follows:

- If a proposition is an axiom, it is a provable proposition.
- If a proposition can be derived from true reachable propositions by means of inference rules, it is a provable proposition.
- The set of provable propositions is the smallest set of propositions satisfying these conditions.

Finite subdivision rules

Main article: Finite subdivision rule

Finite subdivision rules are a geometric form of recursion, which can be used to create fractal-like images. A subdivision rule starts with a collection of polygons labelled by finitely many labels, and then each polygon is subdivided into smaller labelled polygons in a way that depends only on the labels of the original polygon. This process can be iterated. The standard `middle thirds' technique for creating the Cantor set is a subdivision rule, as is barycentric subdivision.

Functional recursion

A function may be recursively defined in terms of itself. A familiar example is the Fibonacci number sequence: $F(n) = F(n - 1) + F(n - 2)$. For such a definition to be useful, it must be reducible to non-recursively defined values: in this case $F(0) = 0$ and $F(1) = 1$.

A famous recursive function is the Ackermann function, which, unlike the Fibonacci sequence, cannot be expressed without recursion.[*citation needed*]

Proofs involving recursive definitions

Applying the standard technique of proof by cases to recursively defined sets or functions, as in the preceding sections, yields structural induction — a powerful generalization of mathematical induction widely used to derive proofs in mathematical logic and computer science.

Recursive optimization

Dynamic programming is an approach to optimization that restates a multiperiod or multistep optimization problem in recursive form. The key result in dynamic programming is the Bellman equation, which writes the value of the optimization problem at an earlier time (or earlier step) in terms of its value at a later time (or later step).

The recursion theorem

In set theory, this is a theorem guaranteeing that recursively defined functions exist. Given a set *X*, an element *a* of *X* and a function $f : X \to X$, the theorem states that there is a unique function $F : \mathbb{N} \to X$ (where $\mathbb{N}$ denotes the set of natural numbers including zero) such that

$$F(0) = a$$
$$F(n+1) = f(F(n))$$

for any natural number *n*.

Proof of uniqueness

Take two functions $F : \mathbb{N} \to X$ and $G : \mathbb{N} \to X$ such that:

$$F(0) = a$$
$$G(0) = a$$
$$F(n+1) = f(F(n))$$
$$G(n+1) = f(G(n))$$

where *a* is an element of *X*.

It can be proved by mathematical induction that $F(n) = G(n)$ for all natural numbers *n*:

Base Case: $F(0) = a = G(0)$ so the equality holds for $n = 0$.

Inductive Step: Suppose $F(k) = G(k)$ for some $k \in \mathbb{N}$. Then $F(k+1) = f(F(k)) = f(G(k)) = G(k+1)$.

Hence F(k) = G(k) implies F(k+1) = G(k+1).

By induction, $F(n) = G(n)$ for all $n \in \mathbb{N}$.

In computer science

Main article: Recursion (computer science)

A common method of simplification is to divide a problem into subproblems of the same type. As a computer programming technique, this is called divide and conquer and is key to the design of many important algorithms. Divide and conquer serves as a top-down approach to problem solving, where problems are solved by solving smaller and smaller instances. A contrary approach is dynamic programming. This approach serves as a bottom-up approach, where problems are solved by solving larger and larger instances, until the desired size is reached.

A classic example of recursion is the definition of the factorial function, given here in C code:

```c
unsigned int factorial(unsigned int n) {
    if (n == 0) {
        return 1;
    } else {
        return n * factorial(n - 1);
    }
}
```

The function calls itself recursively on a smaller version of the input `(n - 1)` and multiplies the result of the recursive call by `n`, until reaching the base case, analogously to the mathematical definition of factorial.

Recursion in computer programming is exemplified when a function is defined in terms of simpler, often smaller versions of itself. The solution to the problem is then devised by combining the solutions obtained from the simpler versions of the problem. One example application of recursion is in parsers for programming languages. The great advantage of recursion is that an infinite set of possible sentences, designs or other data can be defined, parsed or produced by a finite computer program.

Recurrence relations are equations which define one or more sequences recursively. Some specific kinds of recurrence relation can be "solved" to obtain a non-recursive definition (e.g., a closed-form expression).

Use of recursion in an algorithm has both advantages and disadvantages. The main advantage is usually the simplicity of instructions. The main disadvantage is that the memory usage of recursive algorithms may grow very quickly, rendering them impractical for larger instances.

In biology

Shapes that seem to have been created by recursive processes sometimes appear in plants and animals, e.g. in branching structures where one large part branches out to two or more similar smaller parts. One example is Romanesco broccoli.[13]

In art

See also: Mathematics and art and Infinity mirror

Front face of Giotto's *Stefaneschi Triptych*, 1320, recursively contains an image of itself (held up by the kneeling figure in the central panel).

The Russian Doll or Matryoshka doll is a physical artistic example of the recursive concept.[14]

Recursion has been used in paintings since Giotto's *Stefaneschi Triptych*, made in 1320. Its central panel contains the kneeling figure of Cardinal Stefaneschi, holding up the triptych itself as an offering.[15][*failed verification*]

M. C. Escher's *Print Gallery* (1956) is a print which depicts a distorted city containing a gallery which recursively contains the picture, and so *ad infinitum*.[16]

Recursive dolls: the original set of Matryoshka dolls by Zvyozdochkin and Malyutin, 1892

See also

- Corecursion
- Course-of-values recursion
- Digital infinity
- A Dream Within a Dream (poem)
- Droste effect
- False awakening
- Fixed point combinator
- Infinite compositions of analytic functions
- Infinite loop
- Infinitism
- Infinity mirror
- Iterated function
- Mathematical induction
- Mise en abyme
- Reentrant (subroutine)
- Self-reference
- Spiegel im Spiegel
- Strange loop

- Tail recursion
- Tupper's self-referential formula
- Turtles all the way down

References

1. ^ "Peano axioms | mathematics". *Encyclopedia Britannica*. Retrieved 2019-10-24.
2. ^ "The Definitive Glossary of Higher Mathematical Jargon — Recursion". *Math Vault*. 2019-08-01. Retrieved 2019-10-24.
3. ^ "Definition of RECURSIVE". *www.merriam-webster.com*. Retrieved 2019-10-24.
4. ^ Pinker, Steven (1994). *The Language Instinct*. William Morrow.
5. ^ Pinker, Steven; Jackendoff, Ray (2005). "The faculty of language: What's so special about it?". *Cognition*. **95** (2): 201–236. CiteSeerX 10.1.1.116.7784. doi:10.1016/j.cognition.2004.08.004. PMID 15694646.
6. ^ Nordquist, Richard. "What Is Recursion in English Grammar?". *ThoughtCo*. Retrieved 2019-10-24.
7. ^ Nevins, Andrew; Pesetsky, David; Rodrigues, Cilene (2009). "Evidence and argumentation: A reply to Everett (2009)" (PDF). *Language*. **85** (3): 671–681. doi:10.1353/lan.0.0140. Archived from the original (PDF) on 2012-01-06.
8. ^ Drucker, Thomas (4 January 2008). *Perspectives on the History of Mathematical Logic*. Springer Science & Business Media. p. 110. ISBN 978-0-8176-4768-1.
9. ^ Barbara Partee and Mats Rooth. 1983. In Rainer Bäuerle et al., *Meaning, Use, and Interpretation of Language*. Reprinted in Paul Portner and Barbara Partee, eds. 2002. *Formal Semantics: The Essential Readings*. Blackwell.
10. ^ Nederhof, Mark-Jan; Satta, Giorgio (2002), "Parsing Non-recursive Context-free Grammars", *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL '02)*, Stroudsburg, PA, USA: Association for Computational Linguistics, pp. 112–119, doi:10.3115/1073083.1073104.
11. ^ *a* *b* Hunter, David (2011). *Essentials of Discrete Mathematics*. Jones and Bartlett. p. 494. ISBN 9781449604424.
12. ^ "recursion - Google Search". *www.google.com*. Retrieved 2019-10-24.
13. ^ "Picture of the Day: Fractal Cauliflower". Retrieved 19 April 2020.
14. ^ Tang, Daisy. "Recursion". Retrieved 24 September 2015. "More examples of recursion: Russian Matryoshka dolls. Each doll is made of solid wood or is hollow and contains another Matryoshka doll inside it."
15. ^ "Giotto di Bondone and assistants: Stefaneschi triptych". The Vatican. Retrieved 16 September 2015.
16. ^ Cooper, Jonathan (5 September 2007). "Art and Mathematics". Retrieved 5 July 2020.

Bibliography

- Dijkstra, Edsger W. (1960). "Recursive Programming". *Numerische Mathematik*. **2** (1): 312–318. doi:10.1007/BF01386232.

- Johnsonbaugh, Richard (2004). *Discrete Mathematics*. Prentice Hall. ISBN 978-0-13-117686-7.
- Hofstadter, Douglas (1999). *Gödel, Escher, Bach: an Eternal Golden Braid*. Basic Books. ISBN 978-0-465-02656-2.
- Shoenfield, Joseph R. (2000). *Recursion Theory*. A K Peters Ltd. ISBN 978-1-56881-149-9.
- Causey, Robert L. (2001). *Logic, Sets, and Recursion*. Jones & Bartlett. ISBN 978-0-7637-1695-0.
- Cori, Rene; Lascar, Daniel; Pelletier, Donald H. (2001). *Recursion Theory, Gödel's Theorems, Set Theory, Model Theory*. Oxford University Press. ISBN 978-0-19-850050-6.
- Barwise, Jon; Moss, Lawrence S. (1996). *Vicious Circles*. Stanford Univ Center for the Study of Language and Information. ISBN 978-0-19-850050-6. - offers a treatment of corecursion.
- Rosen, Kenneth H. (2002). *Discrete Mathematics and Its Applications*. McGraw-Hill College. ISBN 978-0-07-293033-7.
- Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (2001). *Introduction to Algorithms*. Mit Pr. ISBN 978-0-262-03293-3.
- Kernighan, B.; Ritchie, D. (1988). *The C programming Language*. Prentice Hall. ISBN 978-0-13-110362-7.
- Stokey, Nancy; Robert Lucas; Edward Prescott (1989). *Recursive Methods in Economic Dynamics*. Harvard University Press. ISBN 978-0-674-75096-8.
- Hungerford (1980). *Algebra*. Springer. ISBN 978-0-387-90518-1., first chapter on set theory.

External links

- Recursion - tutorial by Alan Gauld
- Zip Files All The Way Down
- Nevins, Andrew and David Pesetsky and Cilene Rodrigues. Evidence and Argumentation: A Reply to Everett (2009). Language 85.3: 671--681 (2009)

Wikimedia Commons has media related to ***Recursion***.

Look up ***recursion*** or ***recursivity*** in Wiktionary, the free dictionary.

V · T · E **Fractals** [show]

V · T · E **Mathematical logic** [show]

Categories: Recursion | Theory of computation | Self-reference | Feedback