Difference Between Recursion And Iteration

== Mathematics == d17c37ea3a == Data and codata == d17c37ea3a The power of recursion evidently lies in the possibility of defining an infinite set of objects by a finite statement. In the same manner, an infinite number of computations can be described by a finite recursive program, even if this program contains no explicit repetitions. d17c37ea3a

Kleene's recursion theorem One difference between the first and In computability theory, Kleene's recursion theorems are a pair of fundamental results about the application of computable functions to their own descriptions. A related theorem, which constructs fixed points of a computable function, is known as Rogers's theorem and is due to Hartley Rogers, Jr. The theorems were first proved by Stephen Kleene in 1938 and appear in his 1952 book Introduction to Metamathematics. recursion theorem for the first recursion theorem and strong recursion

theorem for the second recursion theorem d17c37ea3a

Fringe search One list now, stores the current iteration, and the other list later stores In computer science, fringe search is a graph search algorithm that finds the least-cost path from a given initial node to one goal node. more or less two lists to iterate over the frontier or fringe of the search tree d17c37ea3a

Both corecursion and recursion can be thought of as operating on trees, which include data structures like lists and streams as special cases. Since recursion must terminate, it only works on trees that are well-founded, i.e. not infinitely deep, which are called data or initial data types; on the other hand, corecursion produces codata or final data types, which includes infinitely deep trees. Codata cannot be represented in memory directly, so is often implemented using self-referential data structures or lazy evaluation. d17c37ea3a Not all programming languages require tail... d17c37ea3a x d17c37ea3a

Iteration related technique of recursion) is a standard element of algorithms. Each repetition of the process is a single iteration, and the outcome of each iteration is the starting point of the next iteration. In mathematics, iteration may refer to the process of iterating a function, i.

applying Iteration means repeating a process to generate a (possibly unbounded) sequence of outcomes. e d17c37ea3a

Recursion (computer science) sort/insertion sort. Recursion and iteration are equally expressive: recursion can be replaced by iteration with an explicit call stack, while iteration can be replaced In computer science, recursion is a method of solving a computational problem where the solution depends on solutions to smaller instances of the same problem. The approach can be applied to many types of problems, and recursion is one of the central ideas of computer science. Recursion solves such recursive problems by using functions that call themselves from within their own code d17c37ea3a

Corecursion corecursion is a type of operation that is dual to (structural) recursion. Whereas recursion consumes a data structure by first handling the topmost layer before descending into its inner parts, corecursion produces a data structure by first defining the topmost layer before defining its inner parts. Corecursion is a particularly important in total languages, as it allows encoding potentially non-terminating computations in a context where every function must terminate. It is supported by theorem provers Agda and Rocq. Whereas recursion consumes a data structure by first handling the topmost layer In computer science, corecursion is a type of

operation that is dual to (structural) recursion d17c37ea3a

In essence, fringe search is a middle ground between A* and the iterative deepening A* variant (IDA*). d17c37ea3a

Total functional programming using the upper bound as an extra argument decremented on each iteration or recursion. For example, quicksort is not trivially shown to be substructural Total functional programming (also known as strong functional programming, to be contrasted with ordinary, or weak functional programming) is a programming paradigm that restricts the range of programs to those that are provably terminating d17c37ea3a

If g(x) is the cost of the search path from the first node to the current, and h(x) is the heuristic estimate of the cost from the current node to the goal, then ƒ(x) = g(x) + h(x), and h* is the actual path cost to the goal. Consider IDA*, which does a recursive left-to-right depth-first search from the root node, stopping the recursion once the goal has been found or the nodes have reached a maximum value ƒ. If no goal is found in the first threshold ƒ, the threshold is then increased and the algorithm searches again. I.E. It iterates on the threshold. d17c37ea3a Most computer programming languages support recursion by allowing a function to call itself from

within its own code. Some functional programming languages (for instance, Clojure) do not define any built-in looping constructs, and instead rely solely on recursion. It is proved in computability theory that these recursion-only languages are Turing complete; this means that they are as powerful (they can be used to solve the same problems) as imperative languages based on control structures such as while and for. d17c37ea3a In mathematics and computer science, iteration (along with the related technique of recursion) is a standard element of algorithms. d17c37ea3a == Notation == d17c37ea3a If the target of a tail is the same subroutine, the subroutine is said to be tail recursive, which is a special case of direct recursion. Tail recursion (or tail-end recursion) is particularly useful, and is often easy to optimize in implementations. d17c37ea3a The statement of the theorems refers to an admissible numbering d17c37ea3a

ID3 algorithm On each iteration of the algorithm, it iterates through every unused attribute of the set S {\displaystyle S} and calculates the entropy In decision tree learning, ID3 (Iterative Dichotomiser 3) is a greedy algorithm invented by Ross Quinlan used to generate a decision tree from a dataset. 5 algorithm. ID3 is the precursor to the C4. the root node. The 3 in the name is meant to signify that this was Quinlan's third attempt at a model based on entropy-based splitting, and the term dichotomiser is a misnomer as it implies a binary split, but the ID3 algorithm can split on multi-valued attributes d17c37ea3a

There are three major inefficiencies with IDA*. First, IDA* will repeat states when there are multiple (sometimes non-optimal) paths to a goal node - this is often solved by keeping a cache of visited states. IDA* thus altered is denoted as memory-enhanced IDA* (ME-IDA*), since it uses some storage. Furthermore, IDA* repeats all previous operations in a search when it iterates in a new threshold... d17c37ea3a Tail calls can be implemented without adding a new stack frame to the call stack. Most of the frame of the current procedure is no longer needed, and can be replaced by the frame of the tail call, modified as appropriate (similar to overlay for processes, but for function calls). The program can then jump to the called subroutine. Producing such code instead of a standard call sequence is called tail-call elimination or tail-call optimization. Tail-call elimination allows procedure calls in tail position to be implemented as efficiently as goto statements, thus allowing efficient structured programming. In the words of Guy L. Steele, "in general, procedure calls may be usefully thought of as GOTO statements which also pass parameters, and can be uniformly coded as [machine code] JUMP instructions". d17c37ea3a The recursion theorems can be applied to construct fixed points of certain operations on computable functions, to generate quines, and to construct functions defined via recursive definitions. d17c37ea3a

Tail call Tail recursion (or tail-end recursion) is particularly useful, and is often easy to optimize in

implementations In computer science, a tail call is a subroutine call performed as the final action of a procedure. which is a special case of direct recursion d17c37ea3a

The ID3 algorithm begins with the original set d17c37ea3a Termination is guaranteed by the following restrictions: d17c37ea3a == Restrictions == d17c37ea3a == Algorithm == d17c37ea3a In total programming languages, the natural numbers may be defined as follows (using Haskell syntax):... d17c37ea3a Repeatedly calling... d17c37ea3a

Mutual recursion Mutual recursion is very common in functional programming and in some problem domains, such as recursive descent parsers, where the datatypes are naturally mutually recursive. In mathematics and computer science, mutual recursion is a form of recursion where two or more mathematical or computational objects, such as functions In mathematics and computer science, mutual recursion is a form of recursion where two or more mathematical or computational objects, such as functions or datatypes, are defined in terms of each other d17c37ea3a

Newton's method the iteration can be determined directly from the difference between the location of the iteration from the left and the location of the iteration from In numerical analysis,

the Newton–Raphson method, also known simply as Newton's method, named after Isaac Newton and Joseph Raphson, is a root-finding algorithm which produces successively better approximations to the roots (or zeroes) of a real-valued function. The most basic version starts with a real-valued function f, its derivative f′, and an initial guess x0 for a root of f. If f satisfies certain assumptions and the initial guess is close, then d17c37ea3a

== Examples == d17c37ea3a