In [mathematics](http://en.wikipedia.org/wiki/Mathematics) and [computer science](http://en.wikipedia.org/wiki/Computer_science), an **algorithm** ([Listen](http://upload.wikimedia.org/wikipedia/commons/7/7f/En-us-algorithm.ogg)[**i**](http://en.wikipedia.org/wiki/File:En-us-algorithm.ogg)[/](http://en.wikipedia.org/wiki/Help:IPA_for_English)[ˈælɡərɪðəm](http://en.wikipedia.org/wiki/Help:IPA_for_English#Key)[/](http://en.wikipedia.org/wiki/Help:IPA_for_English) [***al****-gə-ri-dhəm*](http://en.wikipedia.org/wiki/Wikipedia:Pronunciation_respelling_key)) is a step-by-step procedure for calculations. Algorithms are used for [calculation](http://en.wikipedia.org/wiki/Calculation), [data processing](http://en.wikipedia.org/wiki/Data_processing), and [automated reasoning](http://en.wikipedia.org/wiki/Automated_reasoning).

An algorithm is an [effective method](http://en.wikipedia.org/wiki/Effective_method) expressed as a [finite](http://en.wiktionary.org/wiki/finite) list[[1]](http://en.wikipedia.org/wiki/Algorithm#cite_note-1) of well-defined instructions[[2]](http://en.wikipedia.org/wiki/Algorithm#cite_note-2) for calculating a [function](http://en.wikipedia.org/wiki/Function_(mathematics)).

1. An **algorithm** (pronounced AL-go-rith-um) is a procedure or formula for solving a problem. The word derives from the name of the mathematician, Mohammed ibn-Musa al-Khwarizmi, who was part of the royal court in Baghdad and who lived from about 780 to 850.
2. In computer science, a **data structure** is a particular way of storing and organizing **data** in a computer so that it can be used efficiently. Different kinds of **data structures** are suited to different kinds of applications, and some are highly specialized to specific tasks.

[Interrelationship](http://www.businessdictionary.com/definition/interrelationship.html) among [data elements](http://www.businessdictionary.com/definition/data-element.html) that determine how [data](http://www.businessdictionary.com/definition/data.html) is recorded, manipulated, stored, and presented by a [database](http://www.businessdictionary.com/definition/database.html).

**What is an Abstract Data Type (ADT)? Explain with an example.**

Abstract data types or ADTs are a mathematical specification of a set of data and the set of operations that can be performed on the data. They are abstract in the sense that the focus is on the definitions of the constructor that returns an abstract handle that represents the data, and the various operations with their arguments. The actual implementation is not defined, and does not affect the use of the ADT.  
For example, rational numbers (numbers that can be written in the form a/b where a and b are integers) cannot be represented natively in a computer. A Rational ADT could be defined as shown below.  
**Construction:** Create an instance of a rational number ADT using two integers, a and b, where a represents the numerator and b represents the denominator.  
**Operations:** addition, subtraction, multiplication, division, exponentiation, comparison, simplify, conversion to a real (floating point) number.  
To be a complete specification, each operation should be defined in terms of the data. For example, when multiplying two rational numbers a/b and c/d, the result is defined as ac/bd. Typically, inputs, outputs, preconditions, postconditions, and assumptions to the ADT are specified as well.  
  
When realized in a computer program, the ADT is represented by an interface, which shields a corresponding implementation. Users of an ADT are concerned with the interface, but not the implementation, as the implementation can change in the future. ADTs typically seen in textbooks and implemented in programming languages (or their libraries) include:  
• String ADT  
• List ADT  
• Stack (last-in, first-out) ADT  
• Queue (first-in, first-out) ADT  
• Binary Search Tree ADT  
• Priority Queue ADT  
• Complex Number ADT (imaginary numbers)  
There is a distinction, although sometimes subtle, between  
There is a distinction, although sometimes subtle, between the abstract data type and the data structure used in its implementation. For example, a List ADT can be represented using an array-based implementation or a linked-list implementation. A List is an abstract data type with well-defined operations (add element, remove element, etc.) while a linked-list is a pointer-based data structure that can be used to create a representation of a List. The linked-list implementation is so commonly used to represent a List ADT that the terms are interchanged and understood in common use.  
Similarly, a Binary Search Tree ADT can be represented in several ways: binary  
tree, AVL tree, red-black tree, array, etc. Regardless of the implementation, the Binary Search Tree always has the same operations (insert, remove, find, etc.)