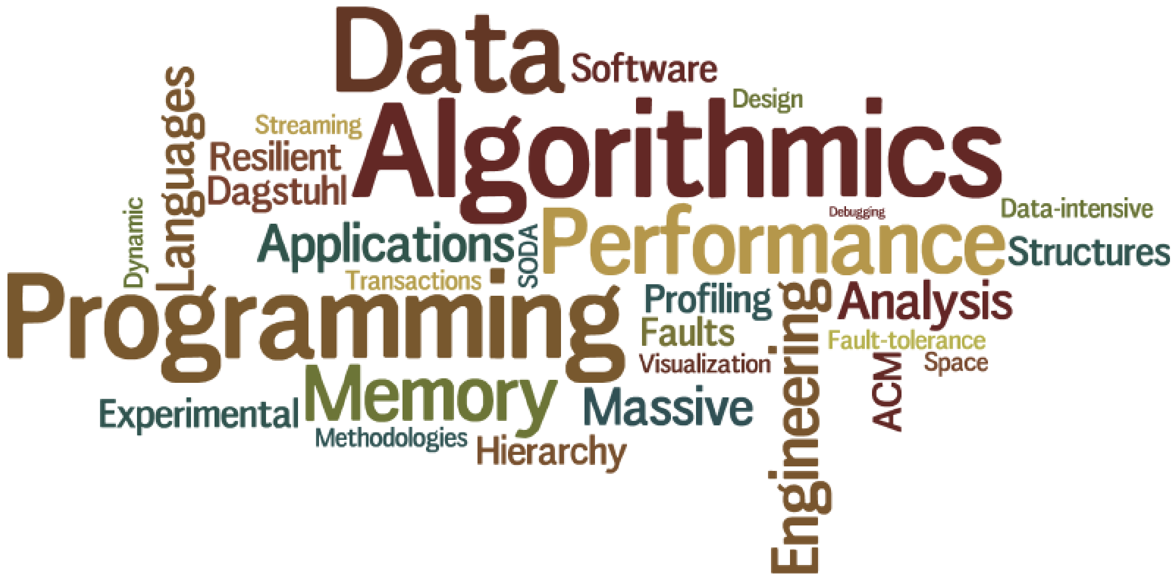
# Algorithm Analysis

Algorithms Defined | Problem Solving | Search
Algorithms | Pseudocode | Abstraction | Review

---

## Computational Fairy Tales: The Ant and the Grasshopper: A Fable of Algorithms

One summer day a grasshopper came upon an ant who was collecting grain. The grasshopper watched as the ant struggled to remove a kernel from a fallen stalk. After a few minutes, the grasshopper spoke....click here to finish the story.



## Algorithms Defined

## What is an algorithm and why should I care?

An **algorithm** is a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer. Algorithms are especially important to computers because computers are really general purpose machines for solving problems. But in order for a computer to be useful, we must give it a problem to solve and a technique for solving the problem. Through the use of algorithms, we can make computers "intelligent" by programming them with various algorithms to solve problems. Because of their speed and accuracy, computers are well-suited for solving tedious problems such as searching for a name in a large telephone directory or adding a long column of numbers. However, the usefulness of computers as problem solving machines is limited because the solutions to some problems cannot be stated in an algorithm. Much of the study of computer science is dedicated to discovering efficient algorithms and representing them so that they can be understood by computers.

Computer Scientists use 'Big O notation' to more accurately describe the performance or complexity of an algorithm, and you are likely to come across this notation very quickly when investigating the performance of algorithms. It characterizes the resources needed by an algorithm and is usually applied to the execution time required, or sometimes the space used by the algorithm.

An algorithm should have the following five characteristics:

- Algorithms are well-ordered.
- Algorithms have unambiguous operations.
- Algorithms have effectively computable operations.
- Algorithms produce a result.
- Algorithms halt in a finite amount of time.

Okay. Now we know what an algorithm is but why should we care?

Kevin Slavin argues that we're living in a world designed for -- and increasingly controlled by -- algorithms. In this riveting talk from TEDGlobal, he shows how these complex computer programs determine espionage tactics, stock prices, movie scripts, and architecture. Slavin also warns that we are writing code we can't understand with implications we can't control.

## Assignment: Algorithms in Your Life Discussion

**Directions:** How are algorithms a part of your life? Share your experiences with algorithms by answering the following questions and posting your answers in a thoughtful paragraph (~100 words) to the *Algorithms in Your Life* discussion board in itsLearning. Then continue the collaborative discussion by responding (meaningfully) to at least two classmates.

## Practice: Rewriting Directions

**Directions:** Read the directions to get to John's house. Now rewrite the directions to John's House using structured English.

**Directions to John's House**

From the Quik Mart, you should follow Saddle road for four miles until you reach a stoplight. Then make a left-hand turn at the stop light. Now you will be on Hollow street. Continue driving on Hollow street for one mile. You should drive past four blocks until you reach the post office. Once you are at the post office, turn right onto Jackson road. Then stay on Jackson for about 10 miles. Eventually you will pass the Happy Meadow farm on your right. Just after Happy Meadow, you should turn left onto Brickland drive. My house is the first house on your left.

---

# Problem Solving

Problem Solving in general consist of multiple steps:

- Understanding the problem
- Breaking the problem into manageable pieces
- Designing a solution
- Considering alternatives to the solution and refining the solution
- Implementing the solution

- Testing the solution and fixing any problems

Christopher Steiner is the author of Automate This (2012) and $20 Per Gallon, a New York Times Bestseller (2009). He is a cofounder at Aisle50, a Y Combinator company that sells grocery deals through the Web. Before starting Aisle50 in 2011, Steiner was a senior writer covering technology at Forbes magazine for seven years.

## Advantages of Combining Algorithms

- Common patterns are recognized
- Consistency
- No need to repeat work, thus less work
- Reusability
- Easier to change

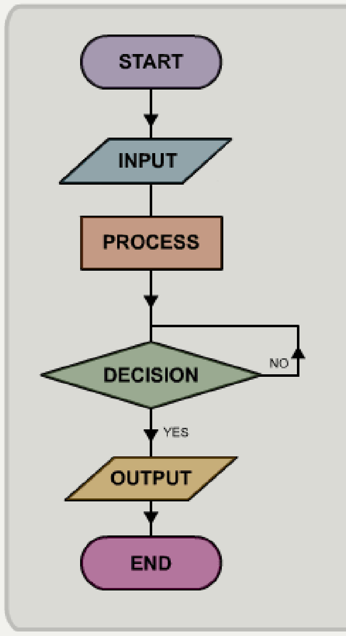
## Assignment: Origami Algorithms

**Directions:** Using one sheet of the origami paper you have, create the animal from the pictorial instructions you were given. Write instructions (steps) in words, to create your origami animal. You cannot use any pictures or diagrams. Be exact and detailed.

---

# Flow Charts

**System flowchart symbols**



All flowcharts begin with the **START** symbol. This shape is called a terminator.

**INPUTS**, such as materials or components.
eg Printed Circuit Board (PCB)

**PROCESSES**, such as activities or tasks, are sometimes used to link to a subroutine (another flowchart) with more detailed steps, egdrill Printed Circuit Board(PCB)

The **DECISION** symbol checks a condition before carrying on, eg is the drilling accurate?

**OUTPUTS**, egPrinted Circuit Board(PCB) with holes drilled.

All flowcharts end with the **END** symbol. This shape is called a terminator.

What is a flow chart? A **flow chart** is a graphical or symbolic representation of a process. Each step in the process is represented by a different symbol and contains a short description of the process step. The flow chart symbols are linked together with arrows showing the process flow direction.

Sometimes it's more effective to visualize something graphically that it is to describe it with words. That is the essence of what flowcharts do for you. Flowcharts explain a process clearly through symbols and text. Moreover, flowcharts give you the gist of the process flow in a single glance. The following are some of the more salient reasons to use flowcharts.

The top five reasons to use flowcharts include:

- Process Documentation/Training Materials
- Workflow Management and Continuous Improvement
- Programming
- Troubleshooting Guides
- Regulatory and Quality Management Requirements

*The Top 5 Reasons To Use Flowcharts*

## Common Flowchart Symbols

Different flow chart symbols have different meanings. The most common flow chart symbols are:

- **Terminator**: An oval flow chart shape indicating the start or end of the process.
- **Process**: A rectangular flow chart shape indicating a normal process flow step.
- **Decision**: A diamond flow chart shape indication a branch in the process flow.
- **Connector**: A small, labeled, circular flow chart shape used to indicate a jump in the process flow. (Shown as the circle with the letter "A", below.)
- **Data**: A parallelogram that indicates data input or output (I/O) for a process.
- **Document**: Used to indicate a document or report (see image in sample flow chart below).

## Assignment: Flowcharting with Board Games

**Directions:** Select a board game such as Sorry, Trouble, CandyLand or Chutes and Ladders. Create a flow chart, with appropriate charting symbols, illustrating the decision making process required while playing the game.

---

## Search Algorithms

Have you ever thought how amazing a search algorithm is? Okay, most of us take for granted that when we type a keyword into a search engine and hit the enter key, what follows is going to the answer to our query.

In computer science, a search algorithm is an algorithm for finding an item with specified properties among a collection of items. When you use a search engine, like Google, you want the answer to your query, not trillions of webpages. Algorithms are computer programs that look for clues to give you back exactly what you want.

For a typical query, there are thousands, if not millions, of webpages with helpful information. Algorithms are the computer processes and formulas that take your questions and turn them into answers. Today Google's algorithms rely on more than 200 unique signals or "clues" that make it possible to guess what you might really be looking for. These signals include things like the terms on websites, the freshness of content, your region and PageRank. *http://www.google.com/insidesearch/howsearchworks/algorithms.html*

Google is always working to improve the quality, accuracy, and efficiency of their search engine.

### Assignment: 10 Algorithms That Dominate Our World

**Directions:** Read the article "The 10 Algorithms That Dominate Our World" and answer the following questions.

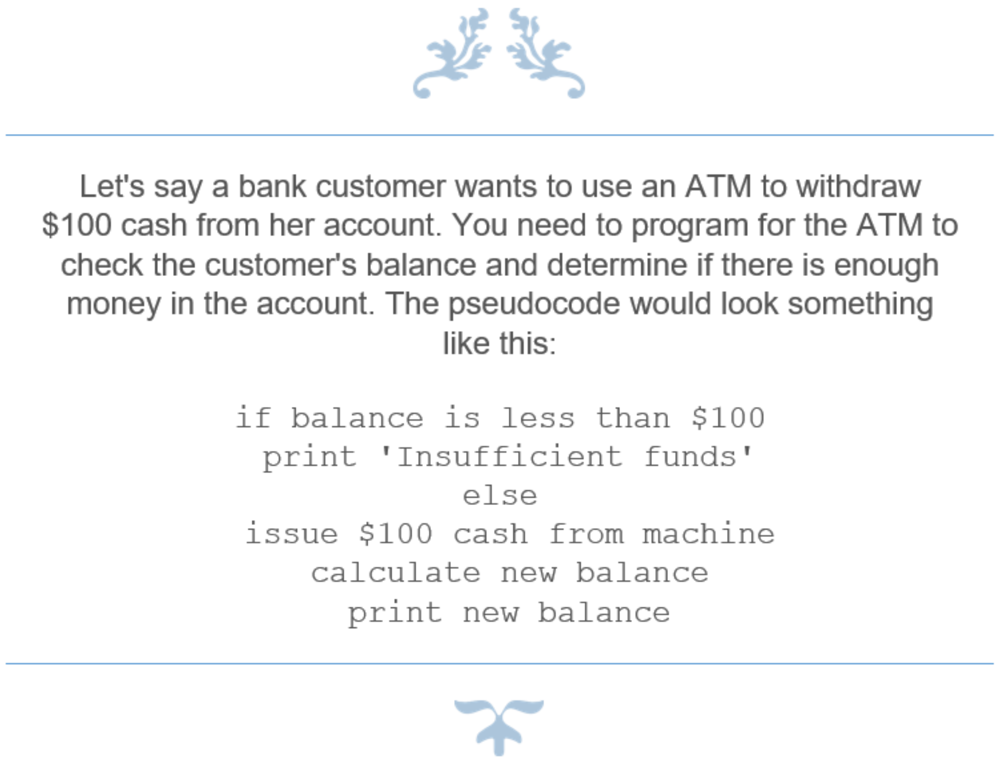
### Assignment: Question for Thought 1

**Directions:** Explain the building blocks of algorithms: sequencing, selection, iteration, and recursion. Your essay should be about 500 words total with about 100 words per building block. Be sure you run spell check before submitting your essay.

---

## Pseudocode

Algorithms can be expressed in many different languages - natural language (English), code, mathematics - all ranging from formal to informal syntax. When algorithms are being designed, they can be expressed in a notation that is independent of any programming language. We call this **pseudocode**. Pseudocode is, as the name implies, not real code, but it looks like code. It helps people to understand a problem domain or solution better without having to add all the baggage necessary when using a real language. In short: it is used only for illustrational purposes. Pseudocode is not an actual programming language. It contains well-defined structures that resemble programming languages, but these are not unique to one particular language.



Let's say a bank customer wants to use an ATM to withdraw $100 cash from her account. You need to program for the ATM to check the customer's balance and determine if there is enough money in the account. The pseudocode would look something like this:

```
if balance is less than $100
print 'Insufficient funds'
else
issue $100 cash from machine
calculate new balance
print new balance
```

You do not need to be a programmer to understand this code. If you were to read out this code, it almost sounds like regular English. On the other hand, the logic and structure is starting to look a lot like code.

The example illustrates some of the basic aspects of pseudocode. First, every instruction is printed on a new line. The logic of the problem needs to be broken down into small steps that are very specific.

Second, pseudocode uses some key programming terms, such as 'if' and 'else.' The use of if-else is a fundamental construct in programming, known as a selection or decision statement. Based on a condition, only one of multiple options is carried out. Every programming language uses these types of if-else constructs. However, the details on exactly how this logic is implemented will vary among languages. When you are writing pseudocode, you don't worry about those differences and focus on the general logic.

Third, individual lines of pseudocode often start with action words that tell you to do something, as you might expect for a series of instructions. Verbs like 'print', 'read,' 'calculate' are common. These verbs correspond closely to commands in programming language, although the actual term may be different. *Using Pseudocode to Map Code @ Study.com*

---

## What is Abstraction?

Modern systems are growing ever more complex, and the ability to use **abstraction** to reduce the complexity of interacting with these systems is critical to successful operation.

An abstraction is a general representation of something -- of some person or place or event or process. Abstractions are formed by including only those details needed to make the abstraction useful to us in some way.

You use abstractions all the time, although you probably don't call them that. You couldn't think or speak without them. Words, ideas, concepts, maps, models, symbols -- these are all examples of abstractions that we use every day to think about and talk about the world.

- Abstraction: An map contains less information that the world that it represents.
    - Abstracting: We create a map by including only those details that will help us achieve the purpose of drawing the map.
- Abstraction: An model airplane contains less information than the real airplane it represents.
    - Abstracting: We create a model airplane by including only those features of the plane that make the model useful for our purposes.
- Abstraction: A floor plan contains less information than the real space it represents.

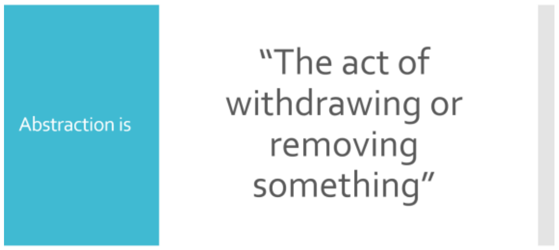
    - Abstracting: We create a floor plan by including only those spatial features that are relevant to why we are making the plan.

Our ideas and concepts are abstractions are formed through a process of leaving out details of the specific things that the idea represents. For example, our concept of chair is general enough to represent any and all chairs.

- Abstraction: An idea or concept is a general representation of something
    - Abstracting: Thinking in abstractions is a fundamental part of human behavior.

Some of the most abstract examples of abstractions are the words we use in our everyday language. Unlike maps and models and floor plans, which have some visible similarities with the objects they represent, a word is completely abstract.

- Abstraction: A word contains almost no specific information about the objects it represents other than their definition.
    - Abstracting: Learning that words represent things is a fundamental skill in learning to speak and read.



Abstraction is

"The act of withdrawing or removing something"

In computer science, the process of simplifying, condensing, and encapsulating is an important problem solving skill. Abstractions in hardware and software help to reduce complexity and make computer systems easier to use and understand. Computer languages contain constants and variables, both of which are abstractions. A constant, such as the numeral 5 represents only one thing, the value 5. A variable, such as the symbol 'X', can represent any number.

- Abstracting: We use variables to make our programs more general and more useful.

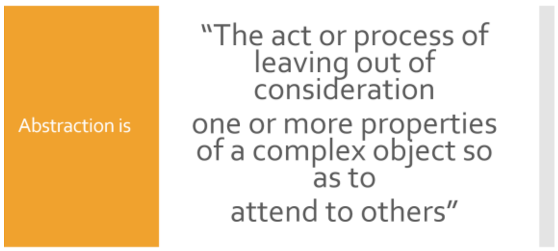
Computer languages have procedures that perform certain calculations for us. The sqrt(x) procedure calculates the square root of any number, x. For example, the sqrt(5) is 2.236.

- Abstraction: The sqrt() procedure simplifies the process of calculating a square root. We don’t need to know exactly how it is done.
    - Abstracting: We will create procedures to make our programs easier to use and understand.

In conclusion, an abstraction is a general and simplified representation of something. Words, symbols, maps, and models are examples. Abstracting is the process of removing details and condensing information in order to focus on relevant aspects of the object or objects we are representing.

What is abstraction? Well, according to WhatIs.com, abstraction is (from the Latin abs, meaning away from and trahere, meaning to draw) is the process of taking away or removing characteristics from something in order to reduce it to a set of essential characteristics. According to the dictionary, abstraction is freedom from representational qualities in art.

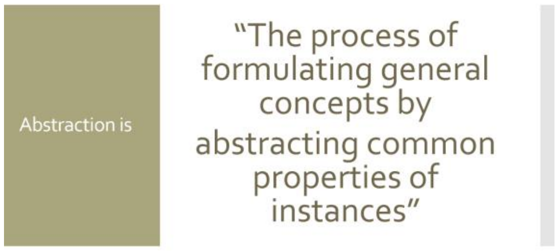
While these are excellent definitions of abstraction, what we really want to know is what is abstraction to a computer programmer. Abstraction can be used to simplify a problem through defining the level of granularity at which data is used.



Abstraction is

“The act or process of leaving out of consideration one or more properties of a complex object so as to attend to others”

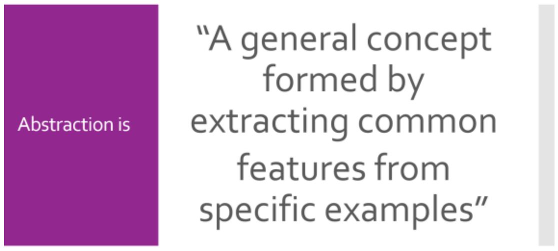
According to Wikipedia, in computer science, abstraction is a technique for managing complexity of computer systems. It works by establishing a level of complexity on which a person interacts with the system, suppressing the more complex details below the current level. The programmer works with an idealized interface (usually well defined) and can add additional levels of functionality that would otherwise be too complex to handle. For example, a programmer writing code that involves numerical operations may not be interested in the way numbers are represented in the underlying hardware (e.g. whether they're 16 bit or 32 bit integers), and where those details have been suppressed it can be said that they were abstracted away, leaving simply numbers with which the programmer can work. In addition, a task of sending an email message across continents would be extremely complex if you start with a piece of optic cable and basic hardware components. By using layers of complexity that have been created to abstract away the physical cables, network layout and presenting the programmer with a virtual data channel, this task is manageable.



Abstraction is

“The process of formulating general concepts by abstracting common properties of instances”

Simple examples of using an interface to simplify a process are available in many software products. Document and image building programs (e.g., Microsoft Word or Adobe Photoshop) often provide features that automatically convert a file to a new format, without the user having to know any of the details about individual formats. Banks provide simple features to transfer funds from one account to another, while hiding the underlying communication and security needed to ensure the privacy of such transactions. Video editing software provides features to combine video files, allowing the user to focus only on the task of transforming the video or photo, rather than on the lower level concepts of bit representation. In all of these examples, a modular code block performs a dedicated function.



Abstraction is

“A general concept formed by extracting common features from specific examples”

In addition to the concept of modular functions, abstraction is also utilized in the problem solving technique of reduction and helps with the concept of "separation of concerns." For example, consider developing a software-based "physics engine" that could be used to simulate the gravity and other physics laws in a simulation for a movie or video game. In reality, objects moving through space are affected by resistance from wind/water in a very complex process. However, a physics engine might simplify this down to a simple coefficient affecting your movement through water (e.g., move half speed in water moving against you and double speed in water moving with you). As another example, wind has a calculable effect on the trajectory of projectiles (e.g., a strong crosswind will push a golf ball of course) and when wind changes this effect becomes increasingly difficult to calculate. Many games disregard this effect when calculating paths. Note that reduction eliminates a number of complex calculations, but introduces a level of inaccuracy. However, in many situations these inaccuracies are acceptable. Jeff Kramer discusses a similar reduction made by Harry Beck when removing the exact geographical dimensions from the London underground map to create a simplified representation. This reduction made the map an inaccurate representation of the underground system, but produced a much more easily understood representation.

An abstraction hides (or suppresses) the right details at the right time. An abstraction hides (or suppresses) the right details at the right time. An object is abstract in that we don't have to think about its internal details in order to use it. For example, we don't have to know how the println method (Java) works in order to invoke it.

A human being can manage only seven (plus or minus 2) pieces of information at one time. But if we group information into chunks (such as objects) we can manage many complicated pieces at once. Java Classes and objects help us write complex software.

## Assignment: Question for Thought 2

**Directions:** Describe how software is built using low and high level abstraction. Post your response (~500 words) directly to the itsLearning textbox. Do not attach a separate document and be sure to proofread before posting.

## Assignment: Question for Thought 3

**Directions:** What are other examples of reducing a complex problem to produce a more manageable problem? Consider the ramifications (consequences) of your proposed reductions (e.g., the loss of accuracy in simulating movement through water as discussed above). Post your response (~100 words) directly to the itsLearning textbox. Do not attach a separate document and be sure to proofread before posting.

## Review

Flash Card Deck created by sarush with GoConqr

## Resources

If you are having problems viewing this page, opening videos, or accessing the URLs, the direct links are posted below. All assignments are submitted in itsLearning. If you have having problems, contact Mrs. Rush through the itsLearning email client.

What is an algorithm and why should you care?: https://www.youtube.com/watch?v=CvSOaYi89B4

The Big Bang Theory - The Friendship Algorithm: https://www.youtube.com/watch?v=k0xgjUhEG3U

How algorithm shape our world: https://www.youtube.com/watch?v=ENWVRcMGDoU

Algorithms are taking over the world: https://www.youtube.com/watch?v=H_aLU-NOdHM

Improving the world's videos with algorithms: https://www.youtube.com/watch?v=lTjjV9xKYhk

Findable photos using data and algorithms: https://www.youtube.com/watch?v=o1mNf6_YqZM

The evolution of search: https://www.youtube.com/watch?v=mTBShTwCnD4

How Google makes improvements to its search algorithm: https://www.youtube.com/watch?v=J5RZOU6vK4Q

Algorithms in pseudocode and flow charts: https://www.youtube.com/watch?v=XDWw4Ltfy5w

**Credits**

Flow Chart Example picture: www.abstractingcs.html

Abstract flower picture: http://www.bhmpics.com/view-flower_and_petals_abstraction-wide.html

Common Flowchart Symbols: http://www.breezetree.com/articles/what-is-a-flow-chart.htm