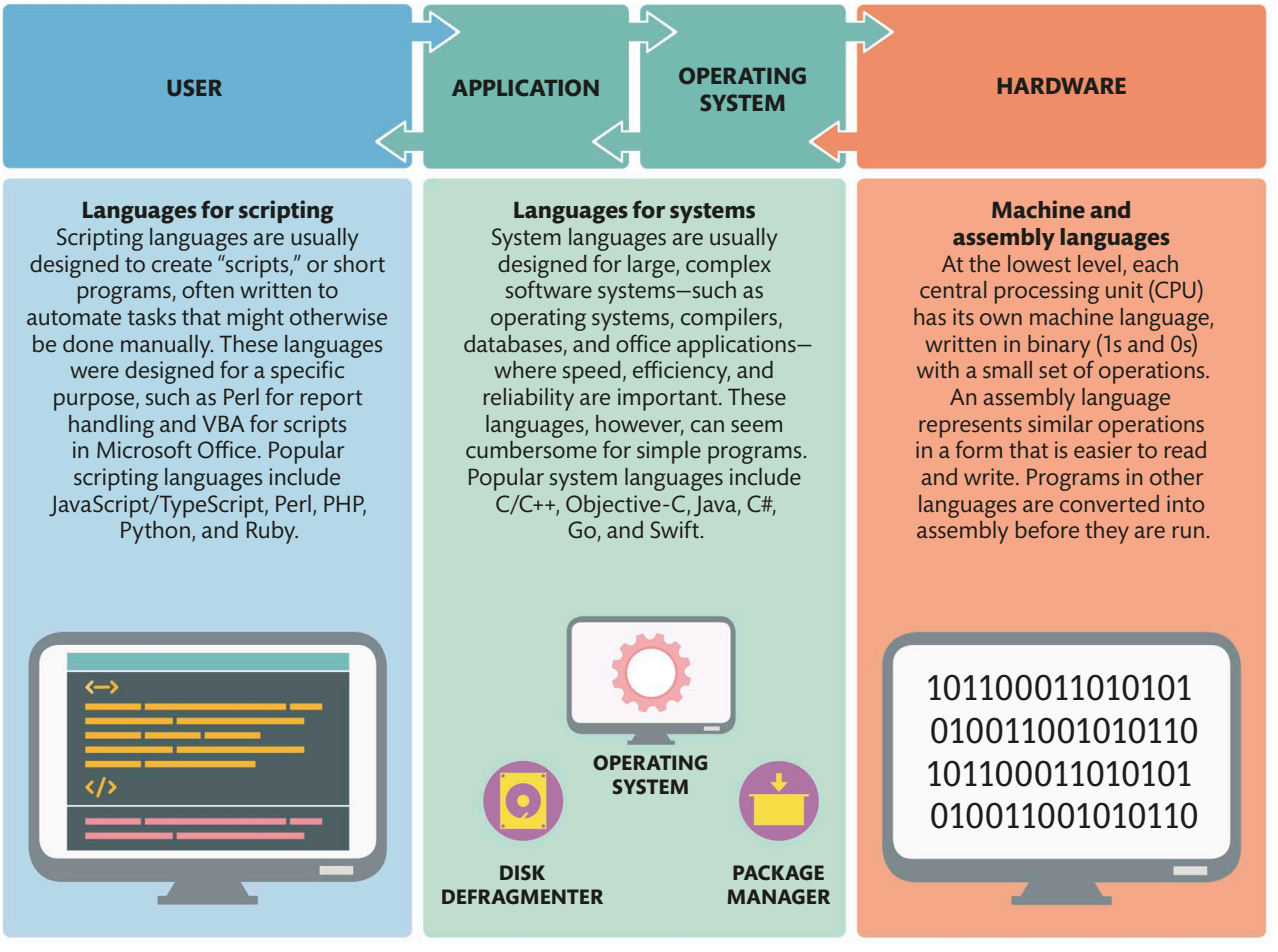
Other programming languages

Every profession has its own vocabulary and ways to describe common problems and solutions. Programming languages were developed to help humans communicate with computers. Most languages are designed for a specific task or domain but are often adapted for other purposes.

Grouping programming languages

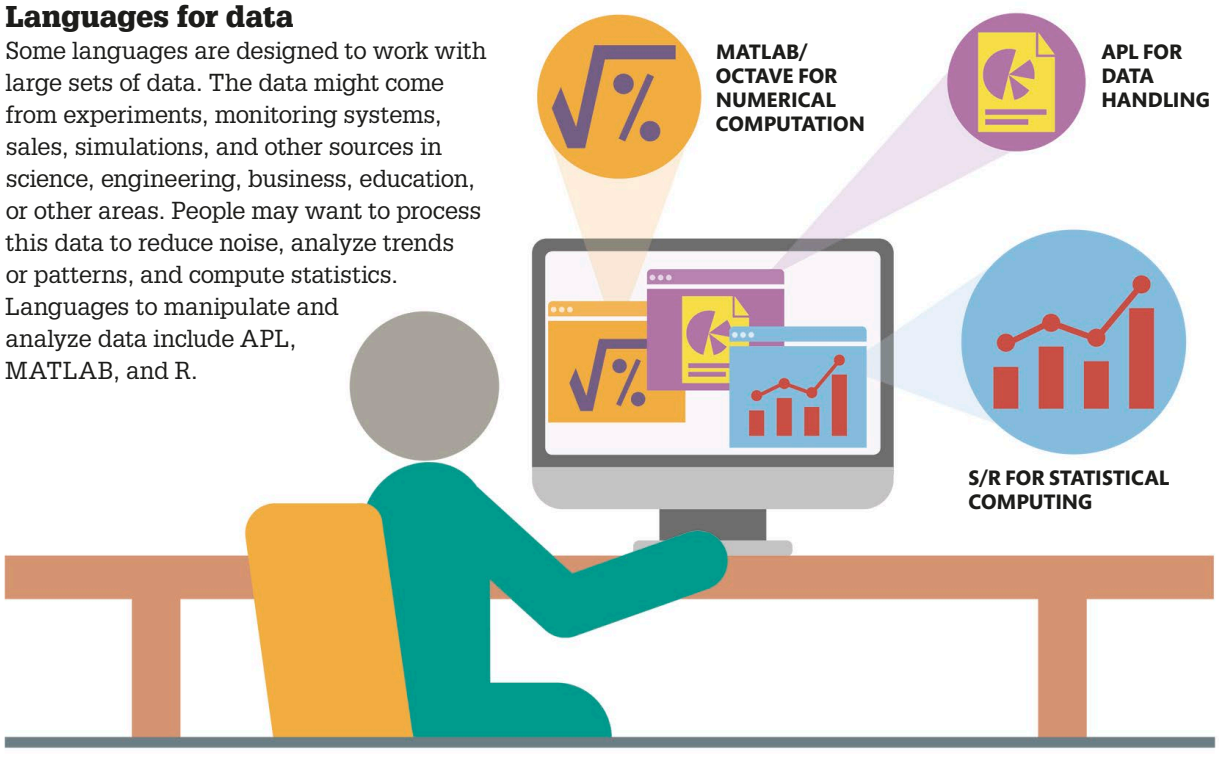
Human languages are grouped into families (such as Germanic or Dravidian) that use similar alphabets, vocabulary, and structures. If you know one language in a family, it is easier to learn others.

Programming languages are also grouped into families and often borrow words and structures from each other. For example, C, C++, Objective-C, Java, C#, Go, and Swift are all related, so developers who know one of them can learn the other languages more easily.



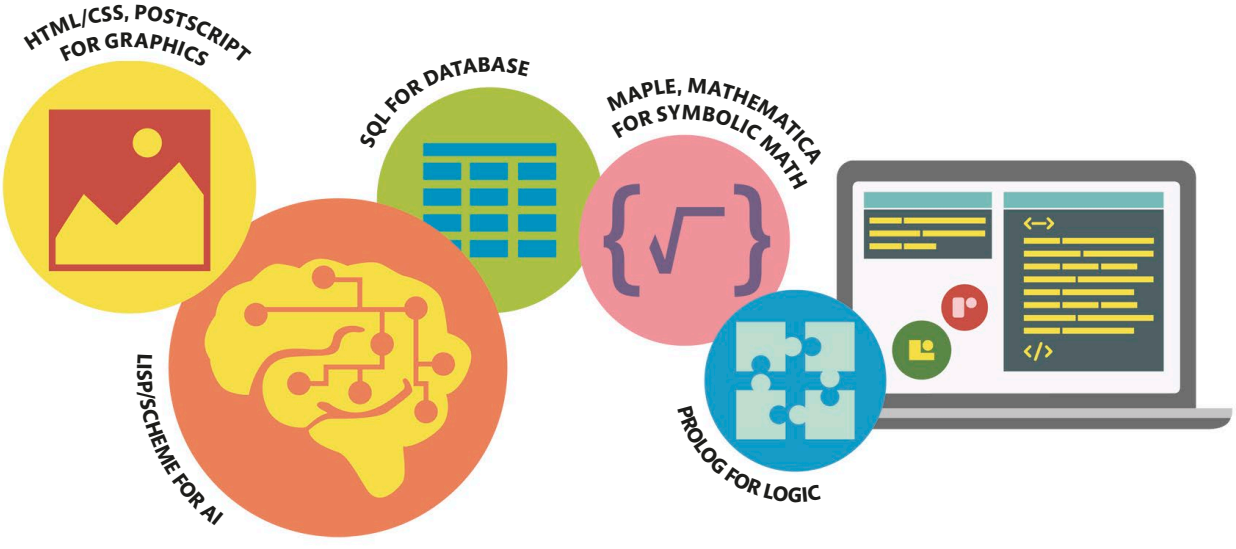
Languages for data

Some languages are designed to work with large sets of data. The data might come from experiments, monitoring systems, sales, simulations, and other sources in science, engineering, business, education, or other areas. People may want to process this data to reduce noise, analyze trends or patterns, and compute statistics. Languages to manipulate and analyze data include APL, MATLAB, and R.



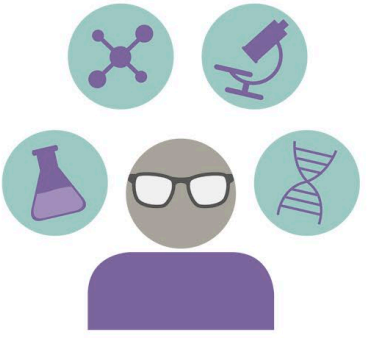
Languages for special purposes

Some programming languages are designed to solve specific problems and might not be useful in other areas. PostScript, TeX, and HTML describe the content and layout of pages with text, images, and other information. SQL is used to manage databases. Maple and Mathematica are used for symbolic mathematics. LISP and Scheme are useful for AI (artificial intelligence). Prolog is used for logic programming.



Early programming languages

There are other widely used languages, some with a long history. Fortran was created in the 1950s for scientific and engineering applications. COBOL was created around 1960 for business applications. BASIC was an easy language created in the mid-1960s for students. Pascal was created around 1970 to encourage structured programming practices and was widely used in education. Ada was created around 1980 to reduce the number of different languages used across the US Department of Defense (DoD).



FORTRAN FOR SCIENCE

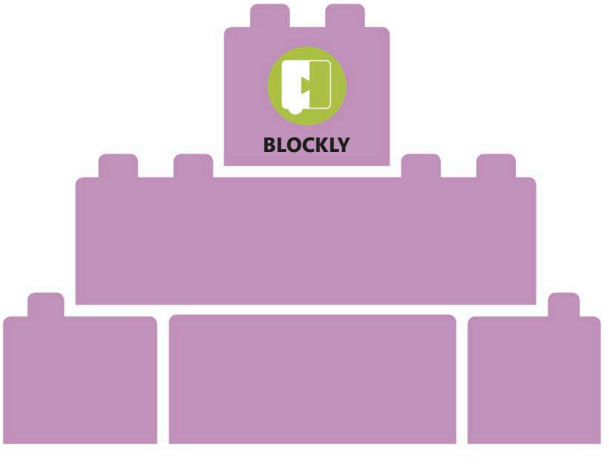
COBOL FOR BUSINESS

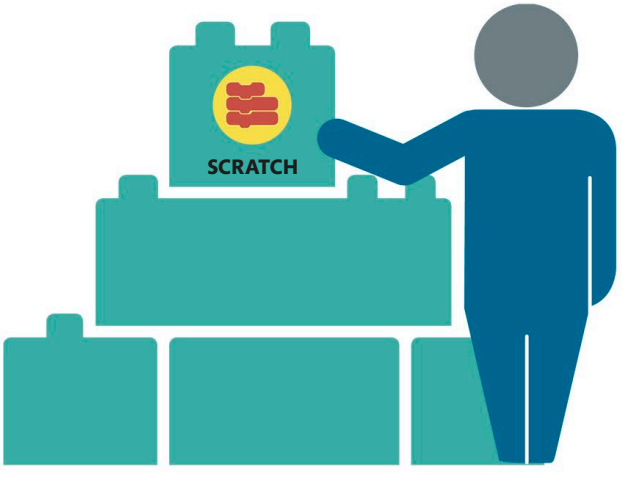
PASCAL FOR EDUCATION

ADA FOR MILITARY

Visual languages

In visual (or block-based) languages, programs are created graphically rather than with text. For example, a user might drag elements into position, connect them, and then type in number values or text messages. Such languages are often designed for nonprogrammers in specific areas, such as education, multimedia, and simulation. Popular visual languages include Blockly, Alice, and Scratch in education, and Kyma, Max, and SynthEdit for music.





Other programming languages

There are hundreds of programming languages, and most developers are proficient in a few, familiar with more, and expect to learn new languages throughout their careers. The table below lists some of these programming languages, with information on when they were first developed, the lead creator, and a brief summary of key ideas and major uses.

POPULAR PROGRAMMING LANGUAGES	
Language, date, and creator	**Key ideas and major uses**
C (1972) Dennis Ritchie	Designed to be concise and portable and to generate efficient machine code. It is widely used for operating systems, compilers, interpreters, and large applications. Many other languages have adapted features and syntax from C.
C++ (1983) Bjarne Stroustrup	Designed to add object-oriented features to C (in C, "C++" adds 1 to the variable c). Widely used for operating systems, compilers, interpreters, and large applications.
Java (1995) James Gosling	Designed to be an object-oriented language based on C and C++. Java was meant to be a "write once, run anywhere" programming language—code written on one type of computer can be run on other types. Widely used for desktop applications and browser-server applications.
Python (1991) Guido van Rossum	Designed for readability and to support multiple programming styles. Uses a small core language with libraries that add more specialized functions. Widely used in web applications, in scientific computing, and for scripting in other software products.
PHP (1994) Rasmus Lerdorf	Designed for web development ("PHP" originally meant "Personal Home Page"), where it is widely used.
JavaScript (1995) Brendan Eich	Designed to create interactive web pages and applications, where is it widely used. JavaScript is also used in some web servers so that a web application can use the same language in the browser and server.
Fortran (1950s) John Backus	Designed at IBM (International Business Machines Corporation) for scientific and engineering applications, which often involve many numeric calculations. Named from "FORmula TRANslation."
COBOL (1959)	Designed for data processing, COBOL was based on the earlier work of Grace Hopper. It was supported by the US Department of Defense, which led to its wider adoption. Named from "COmmon Business-Oriented Language."
BASIC (1964) John Kemeny and Thomas Kurtz	Designed to be easy to use for students in many fields, not just science and mathematics. It expanded into Microsoft Basic (1975) and Visual Basic (1991). Named from "Beginner's All-purpose Symbolic Instruction Code."
Ada (1980s) Jean Ichbiah	Designed for embedded and real-time systems and to reduce the number of languages used across the US Department of Defense (DoD). Named after Ada Lovelace, often described as the first computer programmer.
SQL (1970s) Donald Chamberlin and Raymond Boyce	Designed to edit and search databases, especially "relational databases" (when data is stored in tables that are related to each other in various ways). SQL is short for "Structured Query Language."

Glossary

algorithm
A sequence of steps or instructions that complete a task or solve a problem. In programming, an algorithm often includes repeated steps, decisions between two or subsequences of steps, and steps that refer to other algorithms to do subtasks or solve subproblems.

API (Application Programming Interface)
A set of definitions that programmers can use to access another system without having to understand all of its details. The definitions might include functions, classes, data structures, and data values. Originally named because it defines an interface for programmers to develop applications using an underlying system. *See also* library.

array
A collection of items stored in adjacent locations in the system's memory, using a single name and a numeric index. The index usually starts at 0. Often, all elements in the array have the same type. For example, all integers or all strings of characters. An array is one way to store a list. *See also* list.

attribute
A specific piece of information associated with a data object. For example, an image would have attributes for height and width, and a sound would have attributes for length and sampling rate.

binary
A numbering system used by computers that has only two digits (0 and 1), not the usual decimal system with ten digits (0 to 9). In binary, each position is two times the position to its right, rather than ten times in decimal. For example, 101101 = 1*32 + 0*16 + 1*8 + 1*4 + 0*2 + 1*1 = 45.

bit
Shortened from "binary digit," it is the basic unit for information or communication. The value of a bit can be either 0 or 1. Thus, an eight bit device mostly uses storage elements with 8 bits, which can store 2^8 = (256) different values.

block element
An HTML element that breaks the flow of text and changes the layout of the page. For example, paragraphs (<p>), lists (<ul>, <ol>, <li>), and tables are all block elements. *See also* inline element.

branching statement
A program statement that chooses one of several possible paths or sets of steps, usually based on the value of an expression. For example, an "if-then-else" statement takes the "then" path if an expression is true, and the "else" path if an expression is false. Also called a conditional statement.

Boolean
A value that can be either true or false. Named after George Boole, who defined a logic system based on such values.

bug
A defect or an error in a program or other system that prevents it from working correctly. The term was used in engineering long before computers, but is often associated with a story told by Grace Hopper about a moth stuck in an early computer, causing wrong results.

call
A program statement that causes the computer to run another function, and return to the original function when done.

carousel
A software component in Bootstrap that cycles through a set of elements, like a slideshow.

CDN (Content Delivery Network)
A network of servers spread across different places that can deliver the same content (data or services). For example, when a web browser loads content for a page, the CDN can deliver content from nearby servers, which reduces the wait time and the network traffic.

child object
An object created from a prototype in a parent object. The child shares (inherits) all functions and properties of the parent, but can override them. For example, the parent might define functions and properties for any book, and each child would define the author, title, publisher, and date for a specific book.

class
(1) A definition or description of a category, which usually includes data and functions, and is used to create (instantiate) objects in that category. For example, the class for employees might specify that every employee has a name, phone, and email address, and provide functions to set or display them. (2) In CSS, a style definition that can be added to any number of elements.

cloud
A set of internet servers that can be used instead of a local computer. Cloud storage stores files and other data, and cloud computing does computation.

compiler
A program that analyzes a computer program and converts (compiles) it into machine code so it runs faster. *See also* interpreter.

composite data
Data that is created by combining other simpler data. For example, a string of characters, an array of numbers, or an object. *See also* primitive data.

concatenate
To combine items, usually character strings, one after another. For example, concatenate "snow" and "ball" to get "snowball."

conditional statement
See branching statement

constructor
A special function used to create new objects of a class. Typically, the constructor allocates memory, initializes variables, and does other setup.

data
Any information stored in or used by a computer.

data binding
Connecting (binding) the data values in two or more objects or systems so that changing one also changes the others. For example, binding a GUI element to a data object ensures that changes to the object appear in the GUI, and GUI changes also change the object.

debug
To remove bugs in a program. This might involve running the program with different inputs, adding statements to print or store values

as the program runs, or watching memory values and how they change. *See also* bug.

directory
(1) Also called a folder; a structure to store files, and subdirectories with other files. (2) A list of resources and how to access them.

ECMAScript
The official definition for the scripting language used in browsers and servers, to provide a standard that could be used by JavaScript, JScript, ActionScript, and other web languages.

element
(1) A single value in a larger set, such as an array. (2) In HTML, a part of a document, often with a start tag, content, and a stop tag. For example, "DANGER" is an element that shows "DANGER" as emphasized text.

event
A description of something that has happened, often used as a signal to trigger responses in a program. For example, a mouse-click event could submit a form or display a menu.

execute
Also called run; the command to start a program.

file object
An object that describes or gives access to a file stored in the system's memory.

float
A number with a decimal point in it. It allows a computer to store very large and very small numbers more efficiently. Also called a floating point number.

flowchart
A graphical way to show the steps, branches, and loops in an algorithm.

framework
A collection of software elements that can be combined and customized in various ways, usually to create user applications. For example, Angular, Django, Express, jQuery, React, and Ruby on Rails are all frameworks used for websites and web applications.

function
Code that carries out a specific task, working like a program within a program. Often, a function has a name, a set of input parameters to give information to the function, and a result for the code that called the function. Also called procedures, subroutines, and methods (especially in object-oriented languages).

git
A popular version control system, used to track changes in a set of files, so that users can easily collaborate and access different versions of the same file. *See also* version control system.

global variable
A variable that can be used anywhere in a program. *See also* local variable; variable.

GUI (Graphical User Interface)
Often pronounced "gooey," a user interface is the name for graphical elements, such as buttons, menus, text fields, and checkboxes that make up the part of the program that a user can see and interact with. It is different from a command line interface where everything is displayed as text.

hardware
The physical parts of a computer such as the processor, memory, network connectors, and display. *See also* software.

hack
(1) An ingenious change to code that makes it do something new or simplifies it. (2) To break into other computer systems.

hosting
Also called web hosting; providing server and internet access to clients for their own websites. In dedicated hosting, each client gets their own server; in shared hosting, many clients share a single server.

hover state
The appearance of a GUI element when the cursor or pointer hovers above it. For example, a button or text field might have a different color or border when the mouse hovers above it, to indicate that it is active or ready to use. Also called "mouse-over" state. *See also* normal state.

hyperlink
A text or graphical element that can be clicked, tapped, or otherwise selected to access other information, often using a URL. The other information can be in the same document, another document, or on another website.

index number
A number indicating the position (index) of an element in an array. Many programming languages use square brackets with arrays, so "myArr[3]" means the element in position "3" of array "myArr."

inline element
An HTML element that does not break the flow of text or change the layout of the page. *See also* block element.

input control
A part of a user interface, such as a button, checkbox, or text field, that allows a user to provide input to a program.

instantiate
To create a new object, usually using its class definition.

integer
A number without a fractional part, also called a whole number. Usually, a computer can represent a large, but not infinite, set of integer values.

interface
A boundary between two parts of a system. Thus, a user interface (UI) is how a user interacts with the system, and an API (Application Programming Interface) is a set of definitions to help programmers develop applications using an underlying system.

internet
The global computer network, which is actually a network of networks. Shortened from "interconnected network."

interpreter
A program that executes computer programs one statement at a time, without first converting (or compiling) the program to machine code.

iterate
To execute a task or set of statements repeatedly. Most programming languages have special syntax to make it easier for programs to iterate, either a set number of times or until some condition is met. For example, a program might iterate through an array to perform the same actions on each element.

iteration
The general process of iterating, or the process of repeatedly going through a set of statements in the code.

library
A set of resources that can be reused in other projects. These resources might include functions, classes, data structures, and data values. A library is similar to an API. For example, a math library might have a constant value for pi and functions to compute the sine, cosine, and tangent of an angle. *See also* API.

literal
A fixed value written in source code. In most programming languages, integer and real number literals are written normally, and strings of characters are written between quotation marks.

list
A set of data values, where each value has a specific position in the list. One way to store a list is as an array. *See also* array.

local variable
A variable that can only be used with a particular function or other limited part of a program. *See also* global variable; variable.

loop counter
A variable that counts (tracks) the number of times a loop has been repeated.

machine code
The set of instructions that is used by a computer processor. It is difficult for users to read or write machine code, so other programming languages are used with a compiler or interpreter to convert them to machine code.

malware
Short for "malicious software"; any software designed to gain illegal access to a computer or system. Malware includes viruses, worms, spyware, and ransomware.

memory
Storage used by a computer, using a wide range of technologies, including ROM (read-only memory), RAM (random access memory), solid states drives (SSDs), hard disk drives, and optical drives (e.g., CDs or DVDs). In general, faster technologies are much more expensive, so most computers use smaller amounts of expensive memory (RAM) and larger amounts of cheaper memory (hard disk drives).

metadata
Data that describes other bits of data. For example, web pages use metadata to specify the page title, language, and HTML version, while music files use metadata to specify the composer, performer, title, date recorded, style of music, copyright status, and other information.

module
A package of premade code that can be imported into a program, making lots of useful functions available.

network
A set of computers connected together to allow the sharing of data and resources.

normal state
The way a GUI element (for example, a button) appears normally. *See also* hover state.

object
In object-oriented programming, an object is a component that consists of data and code to manipulate the data.

object-oriented
An approach to coding where programs are organized into classes and objects, which typically contain data values and functions that use or change those values.

opcode
Part of a machine code instruction that specifies the operation rather than other information (such as the memory locations) to use. Shortened from "operation code." *See also* operand.

operand
Parts of a machine code instruction that do not specify the operation, but other information such as the memory locations to use. More generally, a parameter passed to a function. See also *opcode*.

OS (operating system)
The underlying software system that manages resources (both hardware and software) and provides services used by other software. For example, Microsoft Windows, Apple's macOS, and Linux.

output
The result of a program, which might be displayed on a screen, stored in a file, or sent to another program or computer.

parameter
An input for a function. In most languages, a function definition includes a name for each input. For example, the function "sum(x,y)" has two formal parameters "x" and "y".

parent object
An object used to create child objects. The parent has a prototype with functions and properties that can be used by each child. *See also* child object.

parse
To take text or other input data and create a more useful structure out of it. For example, a browser parses a file of characters and creates a data structure (called the Document Object Model, or DOM) that shows which elements contain which other elements.

payload
The actual message within a larger communication. For example, when a browser loads a web page, the payload is the actual HTML that will be displayed.

port
(1) A virtual connection point used to contact a specific service or process. (2) To adapt software to run on another operating system or on other hardware.

primitive data
The basic data type that is used to build more complex data types. For example, characters, integers, and real numbers. *See also* composite data.

primitive variable
A variable that contains primitive data. *See also* reference variable.

processor
The hardware that actually executes a program. Also called the central processing unit or CPU.

protocol
A set of rules that define how something works. For example, HTTP is a high level protocol that describes how a browser and a web server communicate using lower level protocols that handle other details.

prototype
In JavaScript, a built-in variable with functions and properties that can be used by each child object.

pseudo-class
In CSS, a way to define a special state of an element. For example, the pseudo-class ":hover" defines an element's hover state.

reference variable
A variable that does not contain primitive data, but refers to a location in the system's memory where the data is stored. Typically used for arrays, strings, and other composite data. *See also* primitive variable.

run
See execute

run time
(1) The period of time during which a program runs. (2) Software that is used to help programs run.

SASS variable
A variable defined using SASS (Syntactically Awesome Stylesheet). SASS is an extension to CSS and adds features, including variables, that make it easier to develop style sheets.

scope
The parts of a program in which the specific name of a variable, function, or class has meaning. For example, a global variable's scope is the entire program, while a local variable's scope is a single function.

screen reader
A program that finds text on the screen and reads it aloud, to assist users with limited vision.

script
A program written in a scripting language, usually intended for an interpreter rather than a compiler. Originally, scripts were short programs that performed very specific tasks, but over time scripting languages have been used for many other purposes.

semantic
The part of code that is focused on the underlying meaning of text, rather than the rules it follows (the syntax). Most HTML tags focus on the meaning and role of the data, not the appearance. For example, <h1> marks a heading and marks emphasized text, but neither describes how the text should be displayed.

server
A hardware or software system that provides services to other systems or clients. Software servers include database servers, mail servers, and web servers. A hardware server can run more than one software server.

software
A set of instructions or data that tells a computer what to do, including the operating system, libraries, server software, and user applications. *See also* hardware.

source code
The set of instructions that is read and written by users. Source code can also be intended for an interpreter or a compiler.

state
The way a GUI element (a button or a text field) looks, which may change over time. For example, a button might be in its normal state most of the time, but may switch to its hover state when a cursor or pointer moves over it.

string
A sequence of characters that are stored together. This includes letters, numbers, and punctuation. In most languages, literal strings are written within quotation marks.

style definition
In CSS, the definition of a specific style for a category of text. For example, the style definition for a list might include what type of bullets to use and how much to indent.

subset
A group of items taken from another set.

syntax
The part of code that is focused on the rules followed by text rather than its underlying meaning (the semantics). For example, the syntax for emphasized text requires an tag, the text, and an .

tag
In HTML, the text marking the start and end of an element, usually using angle brackets. For example, and are tags used for emphasizing a piece of text.

template literal
A way to write a string that can span multiple lines and insert the values of other variables.

tuple
A short list of items or values; a 2-tuple has two items, and an n-tuple has n items.

URL (Uniform Resource Locator)
A consistent way to refer to the location of some resource and how to access it via the internet.

variable
The name associated with a value stored in the system's memory. In computing, a variable can have different values at different times.

version control system
A system that keeps track of files so that users can easily collaborate and access different versions of the same file. Often, but not necessarily, used in software development.

view
In the Angular framework, a set of screen elements that control what the users can see.

virus
A type of malware that inserts its code into other programs, creating more copies of itself.

web page
A document that can be accessed over the internet. It is displayed in a web browser.

website
A set of related resources, such as web pages, images, sounds, and videos that are stored and accessed together over the internet using a web browser.

Index

Page numbers in **bold** refer to main entries.

C

D

E

F

G

S

T

U

V

W

XYZ

Acknowledgments

DK would like to thank the following people for help in preparing this book: Anjali Sachar, Mridushmita Bose, and George Thomas for design assistance; Deepak Negi for picture research assistance; Nayan Keshan and Kanika Praharaj for code testing; Helen Peters for indexing; Jamie Ambrose for proofreading; and Harish Aggarwal (Senior DTP Designer), Surabhi Wadhwa-Gandhi (Jacket Designer), Priyanka Sharma (Jackets Editorial Coordinator), and Saloni Singh (Managing Jackets Editor).