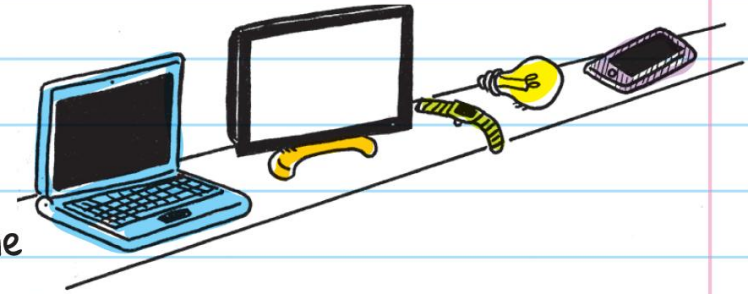
THE FIVE CONCEPT AREAS OF COMPUTER SCIENCE

Computer science can be divided into five concept areas (main parts) of study:

1. Computing Systems



Computing systems are the machines that run programs and process information. Examples are desktop computers, laptops, and phones. Many electronic devices are run by **PROGRAMS** and have small built-in computers. For example, dishwashers, TVs, smart watches, and even some light bulbs.

2. Algorithms and Programming

ALGORITHMS and programming involve writing the programs (code) that tell computers what to do. Programs can be very complex—like iTunes, Safari, Chrome, or Roblox—following many different instructions. They can also be simple, like a program that only prints out the message "Hello, World!"

PROGRAM
A set of instructions (or an algorithm) that has been translated into commands a computer can understand (**code**).

ALGORITHM
A list of steps or instructions written in human language that tells a person how to complete a task.

Writing a program is sometimes broken up into two steps:

STEP 1: The computer scientist makes a list of step-by-step instructions for what she wants her program to do.

Algorithm

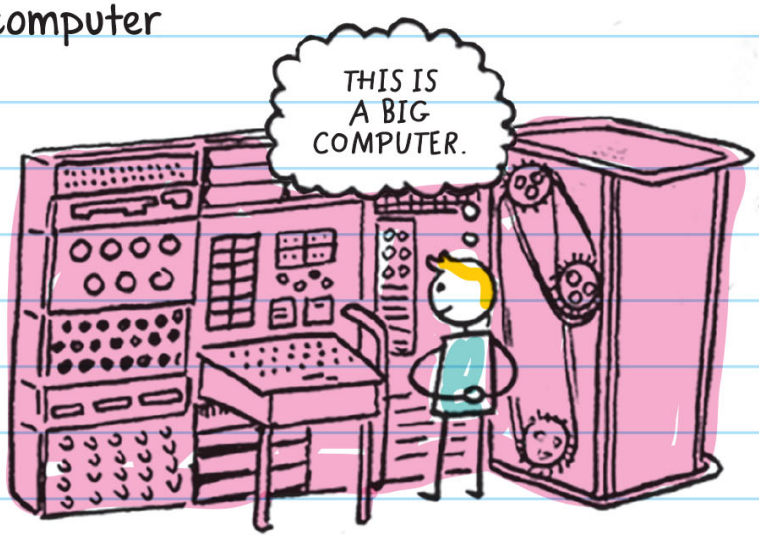
STEP 2: The computer scientist translates her instructions into a language (code) the computer can understand. Now the algorithm is a program.

Programming

Computers are powerful, but they really aren't that smart. They can only do what they're told. Programs give computers the instructions they need to work.

In the 1600s, computing became a profession and the word "computer" meant "a person who makes calculations." Computers (the people) used a tool called the **SLIDE RULE** to help them calculate.

In 1944, during World War II, the first electronic computer was made. It was called COLOSSUS MARK 1 and was used by the British government to crack secret German codes.

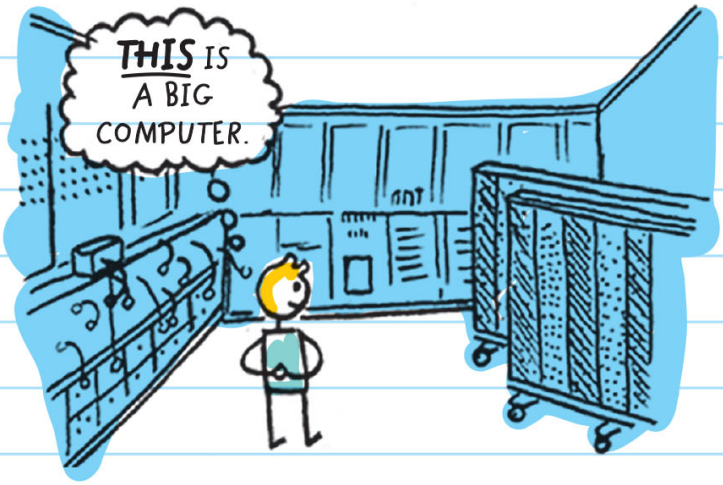


In 1945, Americans created a faster computer called ENIAC.

ENIAC

Stands for "Electronic Numerical Integrator and Computer," which was the first programmable, general-purpose computer.

Early computers were massive—they took up entire rooms! These computers were used by large businesses, governments, or researchers.



The first mass-produced personal computer, the APPLE II, was introduced in 1977. Owners could use the computer to run simple programs or play games.

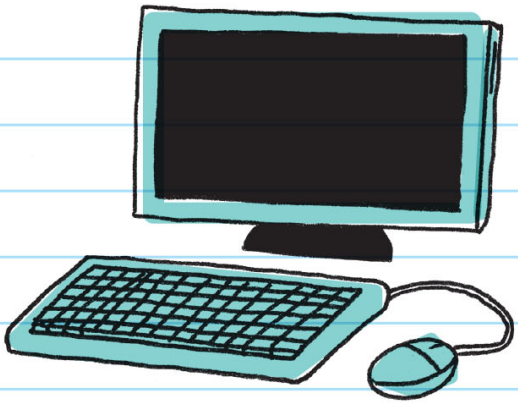
Today, computers are tiny and everywhere, including phones. Smartphones are considered computers because:

- they run programs (like messaging, photo editing, and maps).
- they store information (like photos, contact information, and documents).
- they perform calculations (with a calculator, with the clock/stopwatch, with maps).

Smartphones are more powerful computers than the computer used on the rocket that took astronauts to the moon in 1969.

PARTS OF A WHOLE

Computers are made up of two parts: **HARDWARE** and **SOFTWARE**.



Hardware

Hardware is the physical parts of a computer, like the keyboard, mouse, and screen. An easy way to determine whether part of a computer system is hardware is to look: If you can see it with your eyes, it's hardware.

Software

Software is the set of programs (instructions) that tell a computer what to do, like phone applications, editing programs, and entertainment services. There are all kinds of software: from games for a gaming system, to the program that runs your microwave, to the web browser on your phone. Software is stored on storage devices (hardware) like hard drives, flash drives, and CDs.

An **application** (app) is a program designed for the user to perform a task—like take a photo.

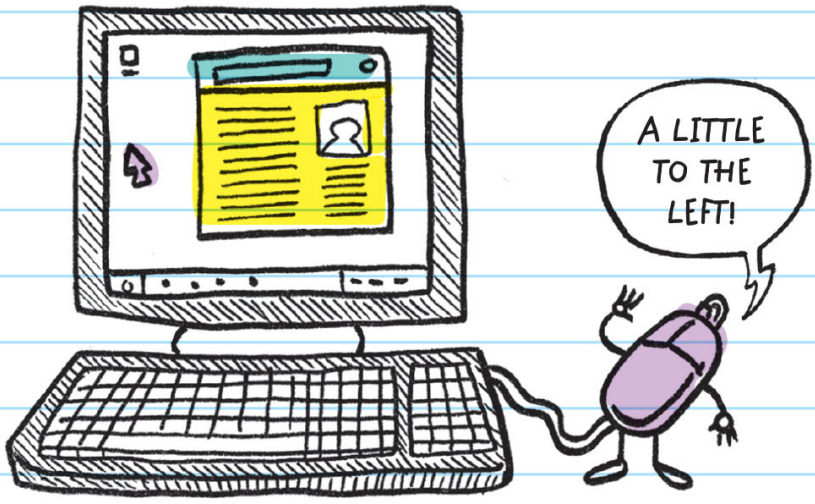
All applications are programs, but not all programs are applications. Some programs are for the computer's use only—not the person using the computer.

A CLOSER LOOK AT HARDWARE

Hardware can be separated into different categories:

Input

Input refers to the parts used to send information to a computer, including a keyboard, mouse, game controller, and webcam.

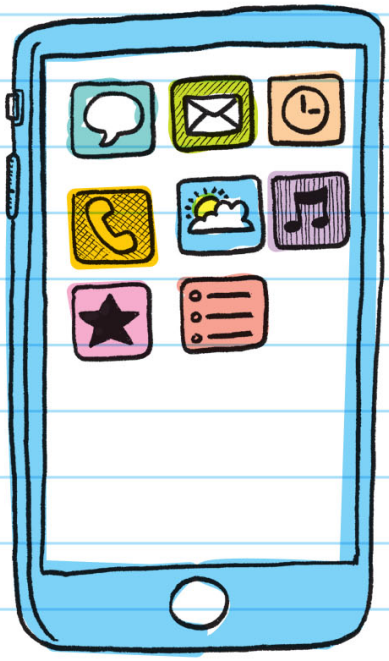


System Software

System software are the programs that make sure the individual hardware devices work together and properly with other programs. **OPERATING SYSTEMS** like Windows on a laptop or iOS (the mobile operating system) on an iPhone are examples of system software. Operating systems run in the background; you cannot see what they're doing on-screen. The operating system makes it possible to install games, social media, and other apps on the phone.

Applications cannot work without the system software.

OPERATING SYSTEMS
Programs that communicate with hardware, allowing other programs to run in the background.



Hardware and software are pretty much useless on their own. Only when they are put together in a complete system do they become tools.

For example:

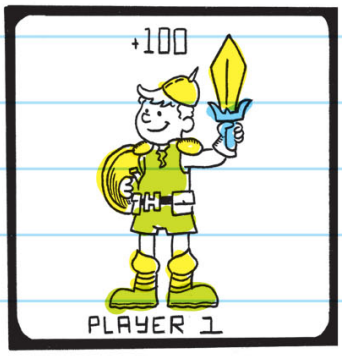
HARDWARE		SOFTWARE		FINAL RESULT
CAMERA	+	PHOTO APP	=	PHONE TO TAKE SELFIES
KEYBOARD + PRINTER	+	WORD PROCESSING PROGRAM	=	PRINTED PARTY INVITATION
CAR + SENSORS + CAMERAS	+	STREET NAVIGATION PROGRAM	=	SELF-DRIVING CAR

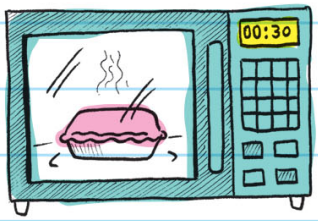
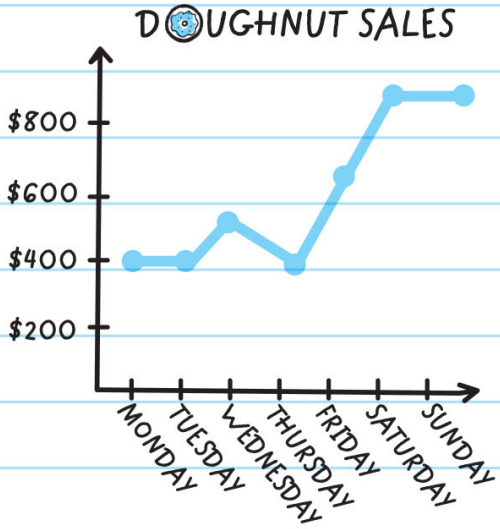
the answers to a survey are data, and the average of the results is information.

Input data is the unorganized information entered into the computer.

Output data is the information after the computer has processed it.

the computer processes

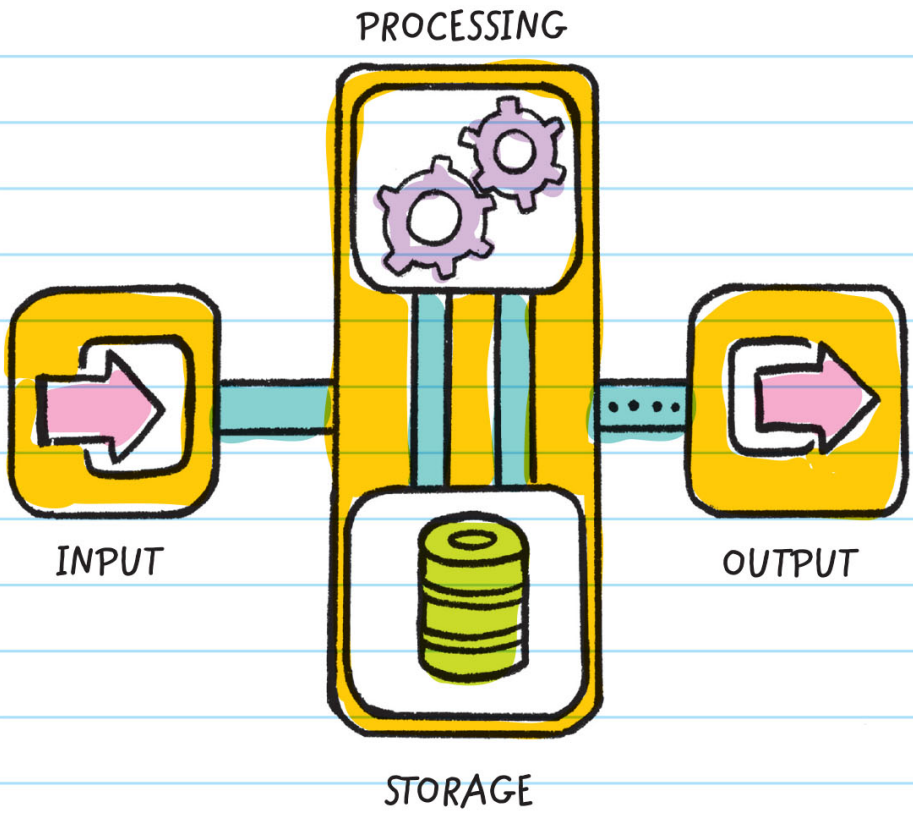
INPUT DATA	OUTPUT DATA
In a video game: pressing keys or buttons on a game console	The character moving around the screen 
In social media: typing comments or uploading photos and videos	Edited videos and photos along with your profile 

INPUT DATA	OUTPUT DATA
On a microwave oven: selecting numbers or buttons on the keypad	The food cooking in the oven for a given amount of time 
In a graphing program: entering numbers for different categories	A graph of the numbers 

Computers take input data like answers to questions, numbers, and uploaded images and turn the data into a format they can understand and process. For example, pictures are broken up into millions of tiny chunks by a computer, and then the color of each chunk is recorded

in a long sequence of code. The computer then processes the code and sends it back to the user as output data. To show the picture, the computer reads the long sequence of code and reconstructs the image on your screen by displaying the corresponding color in each tiny chunk, kind of like a super-detailed paint by number.

CODE
A system of symbols, letters, and numbers used to represent something else



ENCODING DATA

MORSE CODE
Uses dashes and dots to represent the letters of the alphabet

Data has to be written in a format the computer will understand. This means you have to **ENCODE** the data so that the computer can process it. When you encode information, you change an image, video, words, etc. into code. For example, MORSE CODE OPERATORS in World War II would encode written messages into dashes and dots that were sent over telegraph wires.

An outdated communication system that used electrical signals.

When you **DECODE** information, you convert code into an understandable form of communication. For example, when the Morse code operators would receive coded messages in the form of dots and dashes, they would decode the dots and dashes back into English.

Morse code operator

There are several **DATA ENCODING SCHEMES** we can use to help a computer understand input data. Data encoding schemes are ways that we can represent all types of information so that computers can understand it. Computers only understand sequences of the digits 1 and 0.

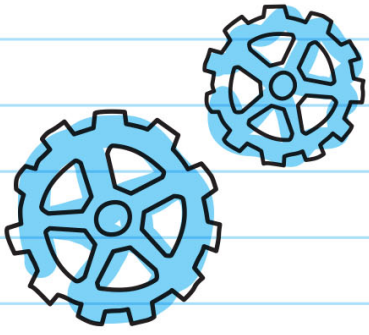
Chapter 6

DESIGNING COMPUTER SYSTEMS

THE ENGINEERING DESIGN PROCESS

When designing computers or programs, computer scientists follow an **ENGINEERING DESIGN PROCESS** to help them understand and improve their designs. Their goal is to make the program as efficient and user-friendly as possible.

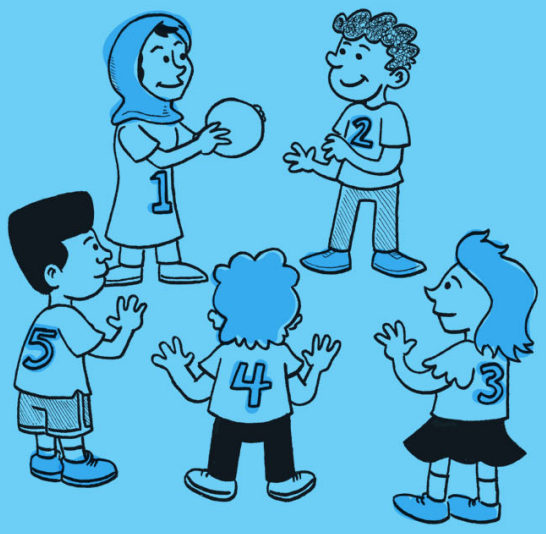
ENGINEERING is a branch of science that studies the design, building, and use of machines and structures to solve problems. **Computer engineering** is the application of study of computer science and the practice of engineering to



There are many slightly different **engineering design processes**, but a basic version is

1. identify the problem
2. plan a solution
3. build/make the plan
4. test
5. improve

and then the cycle starts again.

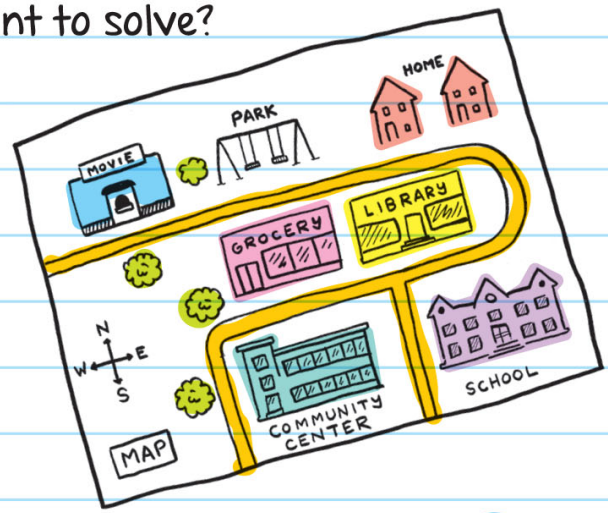


computer software and hardware. **Software engineering** is an application of computer science.

Step 1: Identify the Problem

Software engineers identify the problem and what they need to find a solution. They ask themselves:

- What is the problem I want to solve?
  - Example: find the shortest way home from school that includes stopping at the library and grocery store.

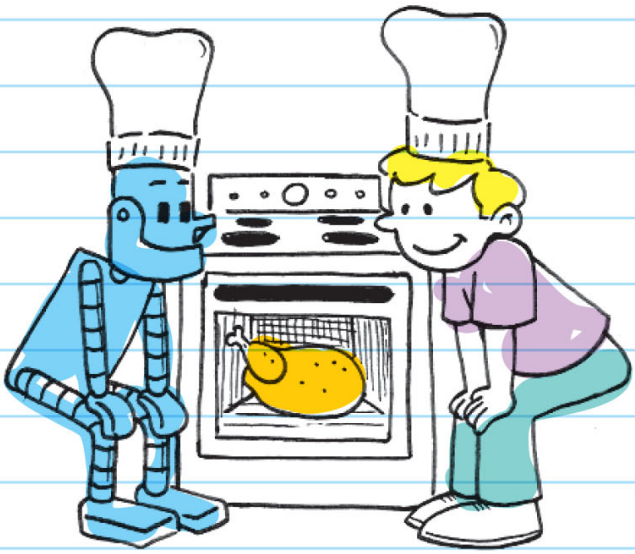


Chapter 11

USING ALGORITHMS

ALGORITHMS

Both humans and computers use algorithms. Algorithms are processes or steps that can be followed. A recipe is an example of an algorithm that both humans and some computers can use.

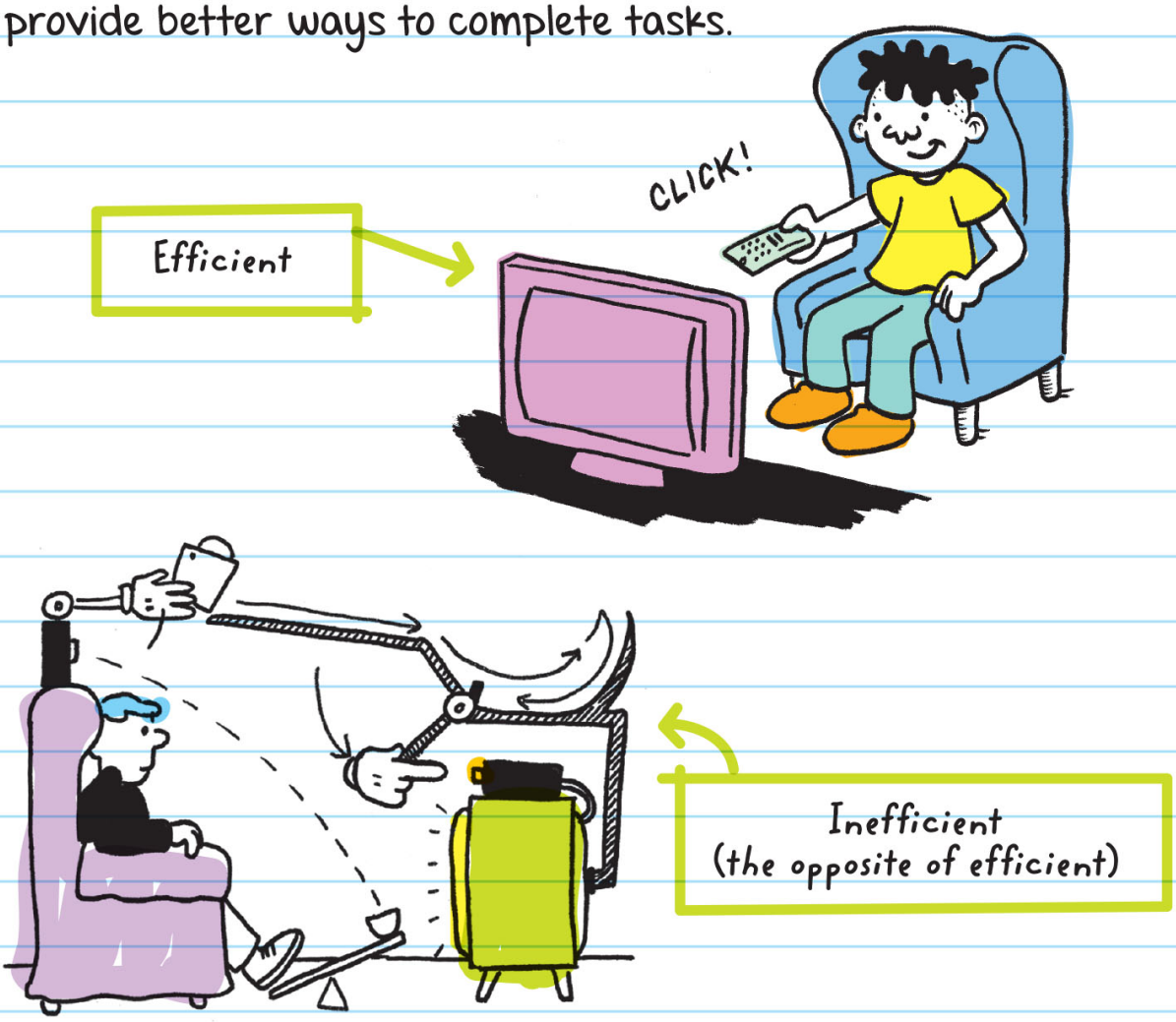


Algorithms give clear instructions for repeating tasks. Different algorithms could be used to complete the same task. For example, there are different ways to divide two numbers using long division or by doing repeated subtraction. Both ways give you the same result,

but one way may be better. In computer science, we are always looking for better algorithms.

Better algorithms are faster, simpler, or more efficient.

Algorithms are useful because they can provide better ways to complete tasks.



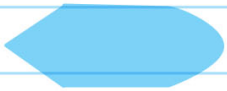
Studying Algorithms

Developing algorithms is its own branch of computer science and is very similar to the study of mathematics. For example, mathematicians and computer scientists have been studying

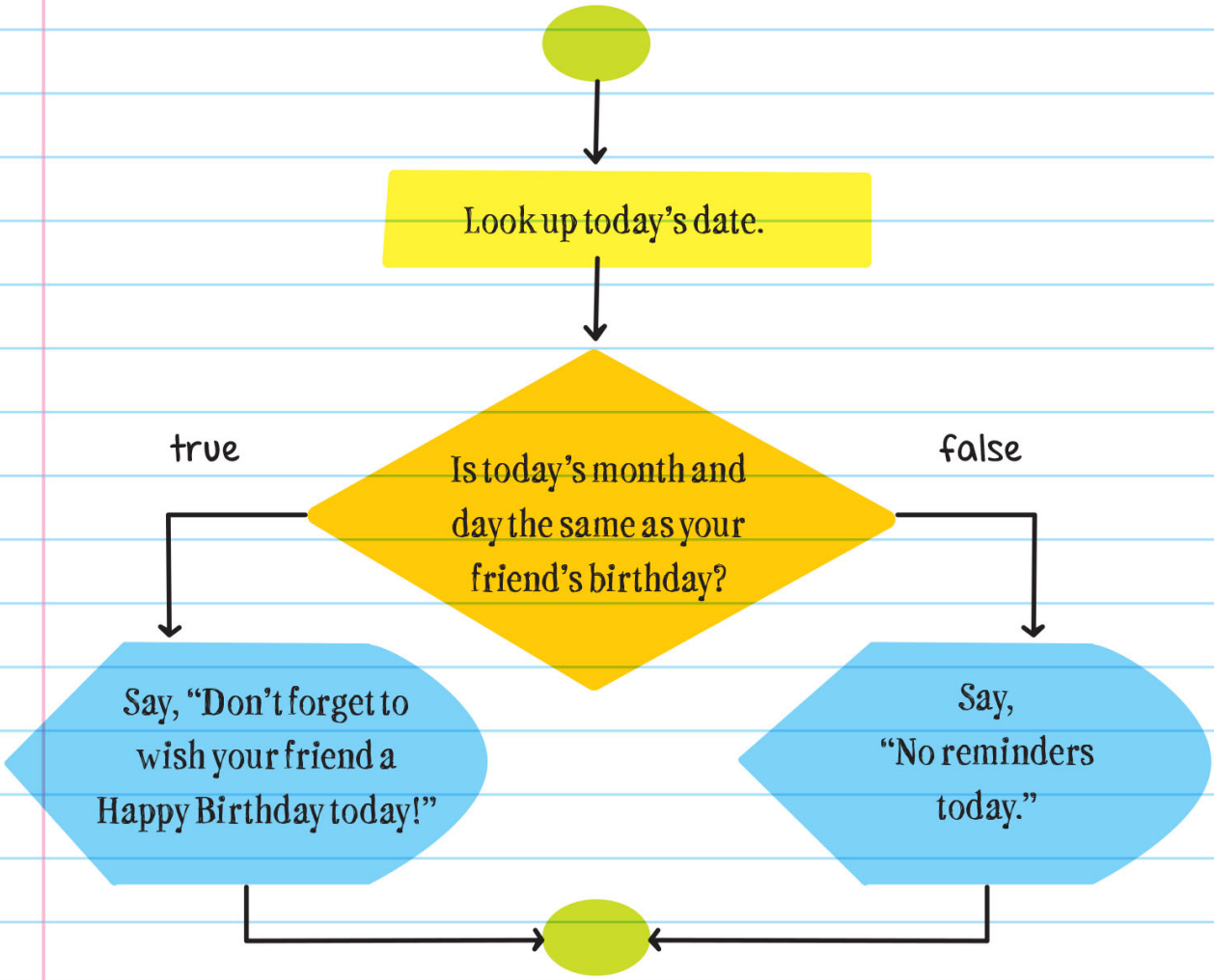
Flowcharts

Flowcharts help programmers visualize the steps in an algorithm. Just like making an outline before writing an essay, flowcharts help organize ideas, and they use specific symbols to represent different parts of an algorithm.

Each symbol in a flowchart has a meaning.

NAME	SYMBOL	MEANING IN FLOWCHART
Oval		The beginning or end of the program
Parallelogram		An input operation
Rectangle		A process to be carried out (addition, subtraction, division, etc.)
Diamond		A decision (or branch) to be made; the program should continue along one of the two routes
Hybrid		An output operation
Flow line		The direction of flow in the program

If you wanted to write an algorithm to check if today is your friend's birthday, the flowchart might look like this:



It might look like this in pseudocode:

Look up today's date.

If the month and day of your friend's birthdate are the same as today's date:

    Then, say "Don't forget to wish your friend a Happy Birthday today!"

    Else, say "No reminders today."