

Pictures

The computer can create graphics.

Famous programs

You can buy many kinds of graphics programs. On pages 45-46, I described the best ones. Here are further comments.

Paint

The easiest kind of graphics program to use is called a **paint program**. It lets you easily create pictures on your screen by using a mouse.

How paint programs arose In January 1984, Apple Computer Company began selling the Mac computer. It was the first affordable computer that included a mouse —and the first affordable computer that included a good paint program.

The Mac's paint program was called **Mac Paint**.

It was invented at Apple Computer Company in 1984 by Bill Atkinson. It ran just on the Mac, was included free with the Mac and showed consumers why a Mac was better than an IBM PC: the Mac let you paint a picture on your screen, and the IBM PC couldn't do that yet.

I explained Mac Paint in the 14th edition of *The Secret Guide to Computers*. If you'd like that edition, phone me at 603-666-6644.

Mac Paint had one major limitation: it couldn't handle colors. It handled just black-and-white, because the original Mac came with just a black-and-white screen.

(Years later, Apple began charging for Mac Paint, Ann Arbor Software invented an improved version called "Full Paint", and Silicon Beach invented a further improvement called "Super Paint". Modern Macs have color.)

The next major advance was **Deluxe Paint**.

It was invented in 1985 by Dan Silva in California and published by Electronic Arts. It was much fancier than Mac Paint and performed gorgeous color tricks.

It ran just on Commodore's Amiga computer. Because of Deluxe Paint, the Amiga quickly developed a reputation as the best computer for generating color graphics.

(Years later, the Amiga faced competition, Commodore went bankrupt, and Electronic Arts made versions of Deluxe Paint for the IBM PC and the Apple 2GS. Deluxe Paint is no longer available.)

Windows includes a free a paint program.

In Windows 3.0, 3.1, and 3.11, the free paint program is called **Paintbrush**. It's a stripped-down version of "PC Paintbrush", which was invented by Z-Soft. Windows 95, 98, Me, XP, Vista, 7, 8, 8.1, and 10 include a free paint program called **Paint**, which is an improved Paintbrush. I explained Paint in the Windows chapters.

The best paint program for kids is **Kid Pix**.

It runs on all popular computers (IBM, Mac, and others). While you paint, it makes funny sounds and talks to you in both English and Spanish. Besides letting you create your own shapes, it includes lots of fun little pre-drawn shapes (stars, snowflakes, trees, etc.), which you can include in your paintings to create backgrounds and pixie dust.

By using Kid Pix, you can create impressive artwork in just a few seconds! Of all the paint programs you can buy, Kid Pix is the one that give you pleasure fastest! Though the pre-drawn shapes look kid-like, they look like they come from *talented* kids! Kid Pix is the only program where it's even more fun to erase your work than to create it, since Kid Pix gives you many dramatic ways to get rid of your painting, such as by dynamiting it: boom!

Educators have given Kid Pix many awards for turning kids into creative artists.

Kid Pix was published by Broderbund but now is published by **Software MacKiev** (which has offices in Boston and the Ukraine city of Kiev). The newest, fanciest version, **Kid Pix 3D**, costs \$50.

The best paint program for professional artists is **Painter**, originally published by Fractal Design but now marketed by Corel.

It was designed for the Mac but now also runs on the IBM PC. Painter amazes artists because it makes the computer's screen accurately imitate different kinds of brushes, inks, and other artist tools. You can choose whether to make the screen look like you're painting in oil, chalk, charcoal, watercolor, or whatever other medium you wish. You can fine-tune each tool, change precisely how "drippy" each tool is, and change the "bumpiness" of the paper's texture.

It even includes a "van Gogh" mode, which lets you paint by using the same kinds of brushstrokes as the artist Vincent van Gogh.

Though Painter can use a mouse, Painter imitates artist tools more accurately if you buy a **pressure-sensitive graphics tablet** (which comes with a pen that records not just *where* you're pressing but also *how hard* you're pressing). The most popular pressure-sensitive tablets are made by **Wacom, Kurta, Calcomp, and Summagraphics**.

Painter is expensive. The newest version, **Painter X3**, costs \$429. You can buy stripped-down versions, called **Painter Lite** (\$70) and **Painter Essentials 4** (\$60). You can download free 30-day trial versions of those programs at www.corel.com.

Since Painter is intended just for creative artists who like to draw squiggles, it doesn't contain commands to draw geometric shapes. For example, it doesn't contain commands to draw an oval, circle, rectangle, or square. All other popular paint programs include such commands.

How paint programs work Each paint program considers your screen to be made of thousands of pixels (dots). The paint program remembers the color of each pixel. The colors of all the pixels are stored in RAM while you're painting. You need lots of RAM if your screen is large & has many pixels, or if you insist on using lots of colors.

All paint programs suffer from this problem:

If you use a paint program to create a shape, then try to shrink that shape, then change your mind and try to expand the shape back to its original size, the final result looks crude and lacks the details that were in the original.

That's because a paint program shrinks a shape by using fewer pixels: some of the pixels that contained details are discarded. The lack of detail becomes noticeable when you try to expand the shape back to its original size.

Another problem is that when you try to rotate a shape, the shape looks cruder, because the shape's pixels get slightly misplaced by "round-off error". If you try to rotate a shape several times, the pixels get progressively more misplaced, and the shape looks cruder and cruder.

When trying to paint, if you expand or twirl,
You get a result that makes you want to hurl.

Paint programs are called **bitmapped graphics programs**.

Draw

A **draw program** does *not* store the color of each pixel. Instead, a draw program stores a memo about a geometric shape and the color of the entire shape.

For example, a draw program stores a line by storing just its starting point, ending point (or angle & length), and color; it stores a circle by storing just the circle's center, radius, and color. By contrast, a paint program would consume lots of RAM storing the color of each of the thousand of pixels that are on the line or circle.

Draw programs are also called **vector-based graphics programs**.

A draw program works faster and more accurately than a paint program if you're drawing geometric shapes. A draw program has no problem handling expansions and rotations. But it has difficulty handling squiggles, since it tries to view each squiggle as made up of many tiny arcs.

Classic computer art

A draw program lets you name different objects, put them in front of other objects, then later move the objects to reveal objects that were hidden. Most paint programs can't do that: in a paint program, creating a new shape automatically erases any shape that was underneath — except for a crude feature that lets you have two “layers”: a “background” and a “foreground”.

In a draw program, you can point to an object you drew and change its color, thickness, or style. In a paint program, the only way to change the appearance of what you drew is to draw it over again.

Unfortunately, the typical draw program is confusing to use, because when you look at what's on your screen you're not sure which “objects” the stuff you're seeing is part of.

The first popular draw program was **Mac Draw**, which ran on the Mac. Now most draw programs use Windows instead. The most popular serious draw program is **Corel Draw**.

Each modern Windows word-processing program also includes a stripped-down draw program, free. For example, to draw while using Microsoft Word (version 2007 or 2010 or 2013), click “Insert” (which is near the screen's top-left corner) then “Shapes”.

CAD

You can buy a program that does **computer-aided drafting & design (CAD)**. Such a program resembles a draw program but does more math.

For example, it can print mock blueprints, with the lengths of all parts marked. It can even compute the surface area (square feet) of any shape, so you can compute how much material to buy to build your structure and cover it.

It lets you give fancy geometric commands, such as “draw a 37-degree angle, but make the point be round instead of sharp, so nobody gets hurt” or “draw a circular arc that goes through these three points” or “draw a line that grazes these two circles, so it's tangent to them”.

The most famous CAD program is **AutoCAD**. That's what most architects and engineers use. It's published by Autodesk and very expensive: \$4195! Cheaper CAD programs that cost under \$50 are for use at home, to help Joe Six-Pack design his backyard deck — and help interior designers plan purple bathrooms.

Most CAD programs include pre-built shapes that you can put in your drawings.

The pre-built shapes are exactly the right size and shape to represent toilets, sinks, stoves, and other household fixtures. Each shape is called a **symbol**. You can buy a bunch of extra symbols; each bunch is called a **symbol library**.

Photo manipulation

To put photographs into your computer, use a **scanner** (which scans in sheets of paper) or a **digital camera**.

Once the picture is in your computer, you can manipulate it by using a paint program. Better yet, use a program that specializes in the fine art of manipulating photos. The best photo-manipulation programs are:

Adobe's **Photoshop** (performs the fanciest tricks, but expensive & hard)

Adobe's **Photoshop Elements** (cheap & easy, for beginners)

Microsoft's **Digital Image Suite** (easy & great but no longer sold)

Kai's Power Goo (stretches a face to create weird expressions, for fun)

During the 1960's, many creative ideas were generated about how computers would someday create their own weird art, using a wild combination of formulas and random numbers, and unshackled by the bounds of human culture.

Here's how to make the computer produce wild art, by using the wonderful classic tricks invented in the 1960's and 1970's....

In 1971, Michael Hord made the computer turn photographs into artistic sketches. Here's what the computer did to a photograph of his boss, and to a photograph of a colleague's girlfriend:

Boss



Woman



To draw each sketch, the computer scanned the original photograph and found the points where the photograph changed dramatically from light to dark. Then, on a sheet of paper, it plotted those points; and through each of those points, it drew a short line perpendicular to the direction in which the original photograph darkened.

More precisely, here's what the computer did.... It looked at four adjacent points on the original photograph:

A B
C D

It computed the darkness of each of those points. Then it computed the "darkening in the X direction", defined as:

$$(\text{darkness at B}) + (\text{darkness at D}) - (\text{darkness at A}) - (\text{darkness at C})$$

Then it computed the "darkening in the Y direction", defined as:

$$(\text{darkness at A}) + (\text{darkness at B}) - (\text{darkness at C}) - (\text{darkness at D})$$

Then it computed the "overall darkening", defined as:

$$(\text{darkening in the X direction})^2 + (\text{darkening in the Y direction})^2$$

If the overall darkening there turned out to be large, the computer sketched a short line, in the vicinity of the points ABCD, and perpendicular to the direction of darkening. More precisely, the line's length was 1, and the line's slope was:

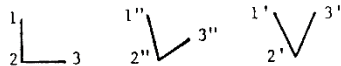
$$\frac{\text{darkening in the X direction}}{\text{darkening in the Y direction}}$$

Morphs

Here's how to make an L slowly become a V. Notice that the letters L and V are both made by connecting three points:



Let 1" be the point halfway between 1 and 1'; let 2" be halfway between 2 and 2'; and let 3" be halfway between 3 and 3'. Then 1", 2", and 3" form a shape that's halfway between an L and a V:



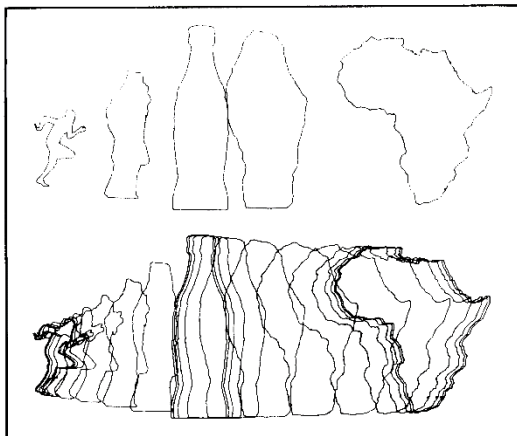
The process can be extended further:



Turning one shape into another (such as turning an L into a V) is called **a metamorphosis** or **morphing**. The intermediate shapes (which are between the L and the V) are called the **morphs**.

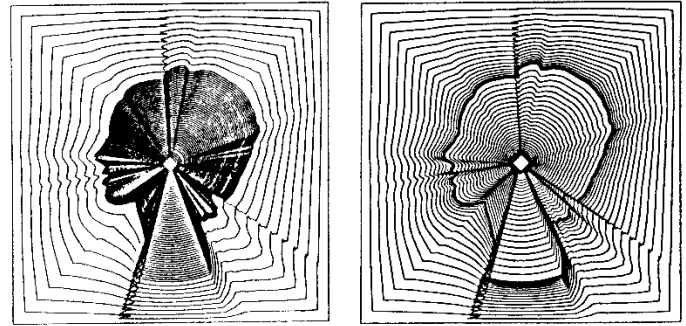
Using that method, the Computer Technique Group of Japan gradually turned a running man into a Coke bottle then into Africa:

Running Cola is Africa



The group turned this head into a square:

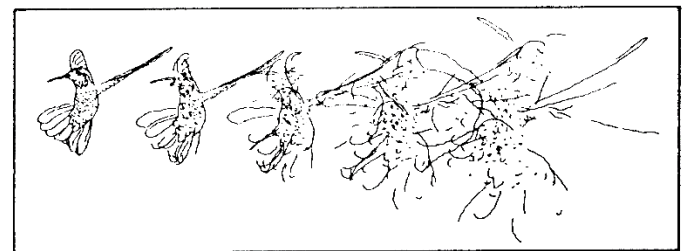
Return to a Square



The head on the left returns to a square by using **arithmetic progression**: the lines are equally spaced. The one on the right uses **geometric progression** instead: the lines are close together near the inside square, but further apart as they expand outward.

Csuri & Shaffer exploded a hummingbird:

Chaos to Order



The hummingbird at the far right was obtained from the one at the far left, by moving each line a random distance and in a random direction (between 45° and -45°).

Computers can make movies.

The best movie ever made by a computer is called *Hunger* (or *La Faim*). It was made back in 1973 by Peter Foldes under the auspices of the Canadian Film Board. Watch it at www.nfb.ca/film/Hunger.

It's a 10-minute cartoon, in color, with music, but goes far beyond anything ever done by Walt Disney. It uses the same technique as *Running Cola is Africa*: it shows objects turning into other objects.

It begins by showing a harried, thin executive at his desk, which has two phones. One of the phones rings. He answers it. While he's talking on that phone, his other phone rings. To talk on both phones simultaneously, his body splits in two. (How does a single body become two bodies? By using the same technique as turning a running man into a Coke bottle.)

On the other side of his desk is an armchair, which turns into a secretary, whose head turns into a clock saying 5PM, which tells the executive to go home. So he stretches his arms in front of him, and becomes his car: his hands become the headlights, his arms become the front fenders, his face becomes the windshield. You have to see it to believe it.

He drives to a restaurant and gets the waitress, who turns into an ice-cream cone. Then he eats her.

As the film progresses, he becomes increasingly fat, lustful, slothful, and miserable. In the end, he falls into hell, where he's encircled by all the poor starving naked children of the world, who eat his flesh. Then the film ends. (Don't see it before eating dinner!)

It combines computer art and left-wing humanitarian politics, to create an unforgettable message.

Using similar techniques, a 30-second movie called *Run* races through what's it's like to be born, live, and die in Japan. Watch it at: www.YouTube.com/watch?v=Gk3-no1foTE

Now morphing is being applied to color photos and video images. For example, Hollywood movies use morphing to show a person gradually turning into a monster; environmentalists use morphing to show a human baby gradually turning into a spotted owl; and portrait photographers who have gone high-tech use morphing to show you gradually turning into the person you admire most (such as your movie idol or your lover).

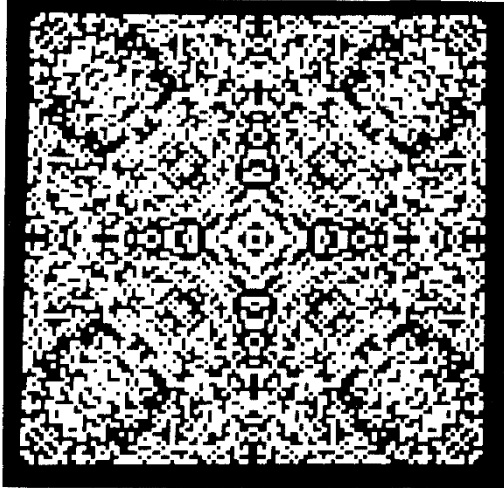
Order versus disorder

Computer artists are starting to believe that **art is a tension between order and disorder**. Too much order, or too much disorder, will bore you. For example, in *Chaos to Order*, the hummingbird on the left is too orderly to be art. The hummingbird on the right is more interesting.

Return to a Square uses arithmetic progression and geometric progression to create an over-all sense of order, but the basic elements are *disorderly*: a head that's bumpy, and a panorama of weird shapes that lie uncomfortably between being heads and squares but are neither.

Many programs create disorder by random numbers. *Chaos to Order* uses random numbers to explode the hummingbird.

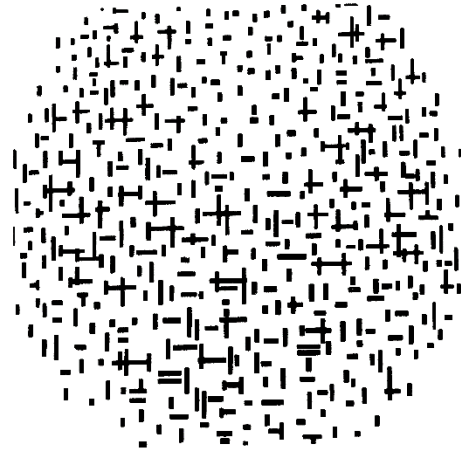
An amazing example of random numbers is this picture by Julesz & Bosche:



To your eyes, the picture seems quite ordered. Actually, it's quite *disordered*. One pie-shaped eighth of it is entirely random; the other seven eighths are copies of it. The copying is the only element of order, but very powerful. Try this experiment: *cover seven-eighths of the picture*. You'll see that the remaining eighth is totally disordered, hence boring.

That program imitates a child's *kaleidoscope*. Do you remember your childhood days, when you played with your kaleidoscope? It was a cardboard "telescope" that contained a disorganized pile of colored glass and stones, plus a series of mirrors that produced eight-way symmetry, so that what you saw resembled a giant multicolored snowflake. The program by Julesz & Bosche uses the same technique, computerized. Hundreds of programmers have imitated Julesz & Bosche, so now you can buy kaleidoscope programs for the IBM PC, Mac, and classic computers (Apple 2 and Radio Shack TRS-80). Or try writing your own!

Take this test:



One of those is a famous painting (*Composition with Lines*, by Piet Mondrian, 1917). The other was done by a computer (programmed by A. Michael Noll in 1965). *Which one was done by the computer? Which one do you like best?*

The solution is on the next page, but *don't peek until you've answered!*

The computer did the top one.

The programmer surveyed 100 people. Most of them (59) thought the computer did the bottom one. Most of them (72) preferred the top one — the one that was actually done by the computer.

The test shows that people can't distinguish computer art from human art, and that the computer's art is more pleasing than the art of a famous painter.

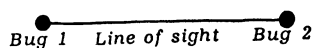
The computer's version is more disordered than Mondrian's. The computer created the disorder by using random numbers. The survey shows that most people like disorder: Mondrian's work is too ordered. It also shows that most people mistakenly think the "computer" means "order".

Envelopes

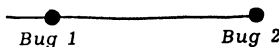
Try this experiment. On a piece of paper, put two dots, like this:



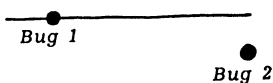
The dots represent little insects, or "bugs". The first bug is looking at the second bug. Draw the first bug's line of sight:



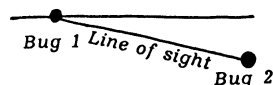
Make the first bug take a step toward the second bug:



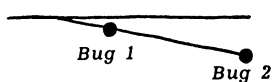
Make the second bug run away, in any direction:



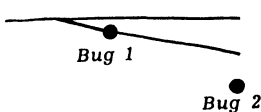
Now repeat the entire process. Again, bug 1 looks at bug 2; draw its line of sight:



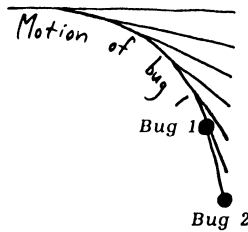
Bug 1 moves toward bug 2:



Bug 2 keeps running away:



If you repeat the process many times, you get this:



The "motion of bug 1" looks like a curve. (In fact, it's a parabola.) The "curve" is composed of many straight lines — the lines of sight. That's how to draw a fancy curve by using straight lines.

Each straight line is called a **tangent** of the curve. The entire collection of straight lines is called the curve's **envelope**. Creating a curve, by drawing the curve's envelope, is called **stitching the curve** — because the lines of sight act as threads, to produce a beautiful curved fabric.

You can program the computer to draw those straight lines. That's how to make the computer draw a fancy curve — even if you know nothing about "equations of curves".

To get a curve that's more interesting, try these experiments:

What if bug 2 doesn't walk in a straight line? What if bug 2 walks in a curve instead?

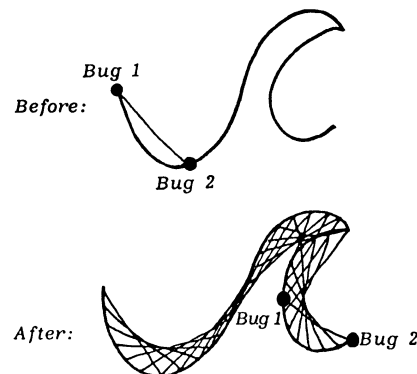
What if bug 1 goes slower than bug 2, and takes smaller steps?

What if the bugs accelerate, or slow down?

What if there are *three* bugs? What if bug 1 chases bug 2, while bug 2 chases bug 3, while bug 3 chases bug 1?

What if there are *many* bugs? What if they all chase each other, and their starting positions are random?

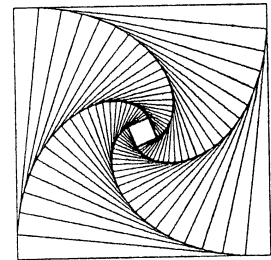
What if there are just two bugs, but the bugs are Volkswagens, which must drive on a highway having nasty curves? Show the bugs driving on the curved highway. Their lines of sight are still straight; but instead of moving along their lines of sight, they must move along the curve that represents the highway.



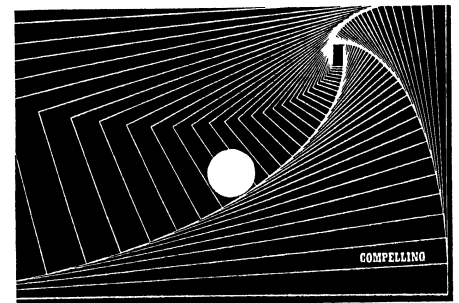
What if each bug has its own highway, and all the bugs stare at each other?

Here are some elaborate examples....

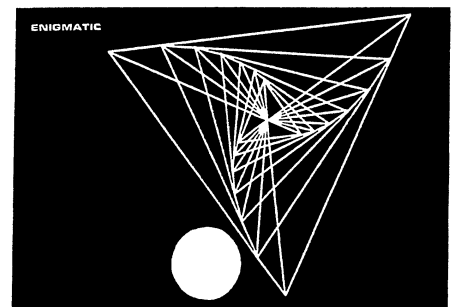
Four bugs chase each other:



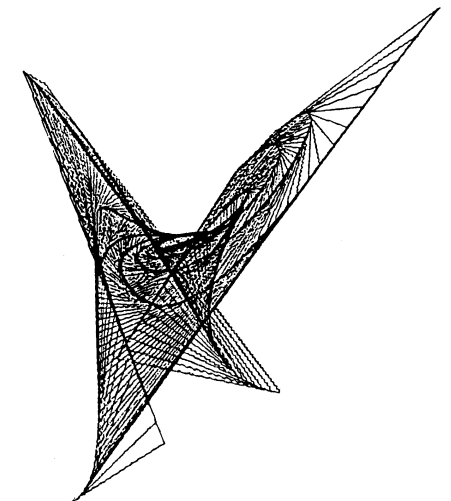
The next example, called *Compelling*, appeared in the famous book and movie, *The Dot and the Line*. (Norton Juster made it by modifying art that had appeared in *Scripta Mathematica*.) It resembles the previous example but makes the 4 bugs start as a rectangle (instead of a square), and makes the bug in the top left corner chase the bug in the opposite corner (while *looking* at a nearby bug instead).



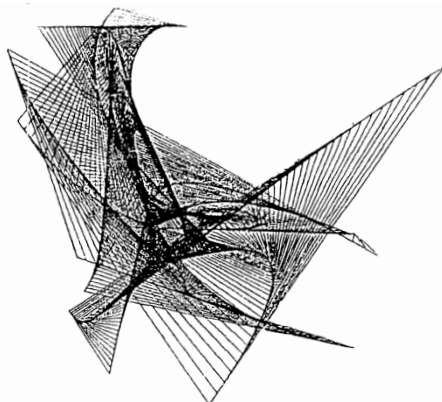
Enigmatic (from *The Dot and the Line*) makes 3 bugs chase each other, while a fourth bug stays motionless in the center:



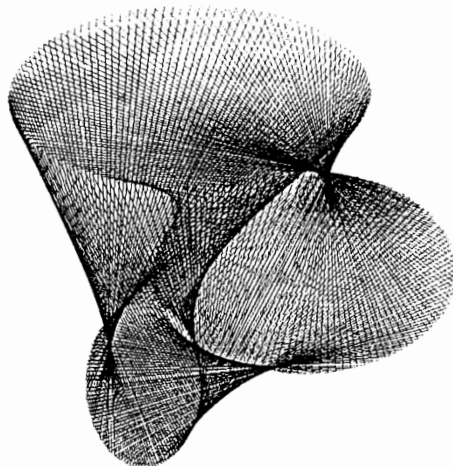
I invented *Kite*, which makes 8 bugs chase each other:



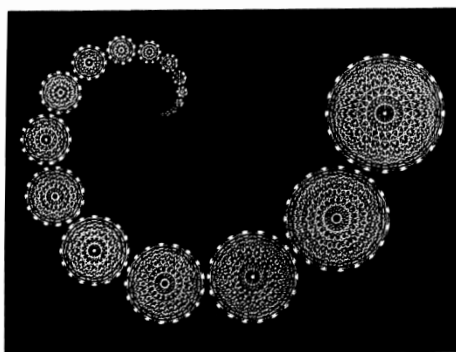
I also invented *Sails*, which makes 14 bugs chase each other:



Elliptic Motion (by my student Toby D'Oench) makes 3 bugs stare at each other, while they travel on 3 elliptical highways:



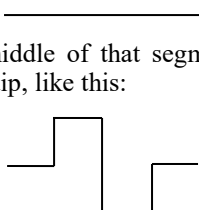
Archimedean Spiral (by Norton Starr) puts bugs on circles. The bugs stare at each other but don't move:



Fractals

A **fractal** is an infinitely bumpy line. Here's how to draw one.

Start by drawing a 1-inch line segment:

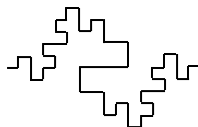


In the middle of that segment, put a bump and dip, like this:

Altogether, that bent path is 2 inches long.

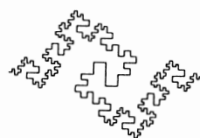
In other words, if the path were made of string, and you stretched the string until it was straight, the string would be 2 inches long. That's twice as long as the 1-inch line segment we started with. So here's the rule: **putting a bump and dip in a path makes the path twice as long.**

That bent path consists of seven segments. Put a bump and a dip in the middle of each segment, like this:



Altogether, those bumps and dips make the path twice as long again, so now the path is 4 inches long.

Again, put a bump and dip in the middle of each segment, so you get this:



Again the path's length has been doubled, so now the path is 8 inches long.

If you again put a bump and dip in the middle of each segment, the path's length doubles again, so the path becomes 16 inches long. If you repeat the procedure *again*, the path reaches 32 inches.

If you repeat that procedure infinitely often, you'll develop a path that's infinitely wiggly and infinitely long. That path is longer than any finite line segment. It's longer than any finite 1-dimensional object. But it still isn't a 2-dimensional object, since it isn't an "enclosed area". Since it's bigger than 1-dimensional but not quite 2-dimensional, it's called **1½-dimensional**. Since 1½ contains a fraction, it's called **fractional-dimensional** or, more briefly, **fractal**.

Look out your window at the horizon. What do you see?

The horizon is a horizontal line with bumps (which represent hills and buildings and other objects). But on each hill you see tiny bumps, which are trees; and on each tree you see even tinier bumps, which are leaves; and on each leaf you see even tinier bumps, which are the various parts of the leaf; and each part of the leaf is made of even smaller bumps (molecules), which have even smaller bumps (atoms), which have even smaller bumps (subatomic particles).

Yes, the horizon is an infinitely bumpy line, a fractal!

You can buy software that creates fractals. Computer artists use fractal software to draw horizons, landscapes, and other bumpy biological objects. For example, they used fractal software to create landscapes for the *Star Wars*

movies. You can also use fractals to draw a bumpy face that has zillions of zits.

Now you understand the computer artist's philosophy of life: "Life's a lot of lumps."

What's art?

To create art, write a weird program whose consequences you don't fully understand, tell the computer to obey it, and look at the computer's drawing. If the drawing looks nice, keep it and call it "art" — even if the drawing wasn't what you expected. Maybe it resulted from an error, but so what? **Anything interesting is art.**

If the drawing "has potential" but isn't totally satisfying, change a few lines of the program and see what happens — or run the program again unchanged and hope the random numbers will fall differently. The last thing to invent is the title. Whatever the drawing reminds you of becomes the title.

For example, that's how I produced *Kite and Sails*.

I did *not* say to myself, "I want to draw a kite and sails". I just let the computer pick random starting points for the bugs and watched what happened. I said to myself, "Gee whiz, those drawings remind me of a kite and sails." So I named them *Kite* and *Sails*, and pretended I chose those shapes purposely.

That method may seem a long way from DaVinci, but it's how most computer art gets created. The rationale is: don't overplan.... let the computer "do its own thing"; it will give you art that escapes from the bounds of human culture and so expands your horizons!

Modern style

Computer art has changed. The **classic style** — which you've been looking at — consists of hundreds of thin lines in mathematical patterns, drawn on paper and with little regard for color. The **modern style** uses big blobs and streaks of color, flashed on a TV tube or film, which is then photographed.

Uncreative art

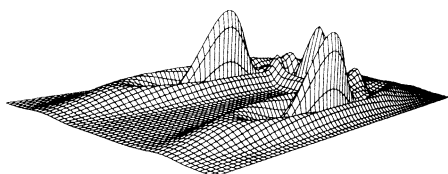
You've seen that computers can create their own weird art by using a wild combination of formulas and random numbers, unshackled by the bounds of human culture.

Computer programs let people create art easily and cheaply. Unfortunately, the typical person who buys a graphics program uses it to create the same kind of junk art that would be created by hand — just faster and more precisely. That's the problem with computers: they make the production of mediocrity even easier and more glitzy.

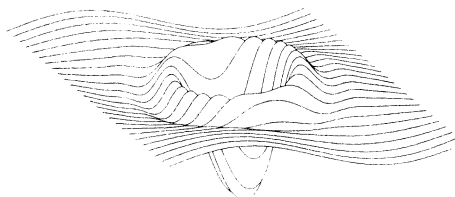
3-D drawing

The computer drew these 3-dimensional surfaces:

Three Peaks
by John Szabo

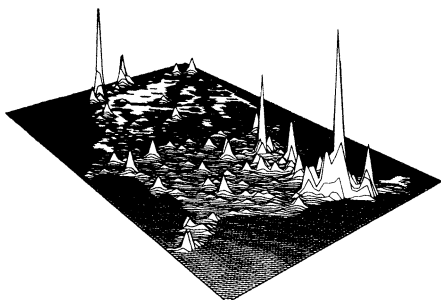


Dip



Those were done for the sake of art. This was done for the sake of science:

Population Density in the U.S.
by Harvard University Mapping Service



The hardest part about 3-dimensional drawing is figuring out which lines the computer should *not* show, because they're hidden behind other surfaces.

Compute the coordinates

Try this experiment. Put your finger on the bridge of your nose (between your eyes). Now move your finger 2 inches to the right (so your finger is close to your right eye). Then move your finger 3 inches up (so your finger is near the upper right corner of your forehead). From there, move your finger 8 inches forward (so your finger is 8 inches in front of your forehead).

Your finger's current position is called (2,3,8), because you reached it by moving 2 inches right, then 3 inches up, then 8 inches forward. The 2 is called the **X coordinate**; the 3 is called the **Y coordinate**; the 8 is called the **Z coordinate**.

You can reach any point in the universe by the same method! Start at the bridge of your nose, and get to the point by moving right (or left), then up (or down), then forward (or back).

The distance you move to the right is called the **X coordinate**.
(If you move to the left instead, the X coordinate is a negative number.)

The distance you move up is called the **Y coordinate**.
(If you move down instead, the Y coordinate is a negative number).

The distance you move forward is called the **Z coordinate**.
(If you move back instead, the Z coordinate is a negative number).

Project the coordinates

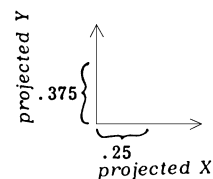
To draw a picture of a 3-dimensional object, put the object in front of you, and then follow these instructions....

Pick a point on the object. (If the object has corners, pick one of the corners.)

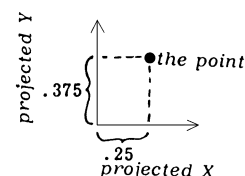
Figure out that point's X, Y, and Z coordinates (by putting your finger on the bridge of your nose and then seeing how far you must move your finger right, up, and forward to reach the object).

Compute the point's **projected X coordinate** (which is X/Z) and the point's **projected Y coordinate** (which is Y/Z). For example, if X is 2 and Y is 3 and Z is 8, the projected X coordinate is $2/8$ (which is .25) and the projected Y coordinate is $3/8$ (which is .375).

On graph paper, plot the projected X coordinate and the projected Y coordinate, like this:



Then plot the point:



Do that procedure for each point on the object (or at least for the corners). Connect the dots and — presto! — you have a 3-dimensional picture of the object! And the picture is mathematically accurate! It's what artists call a "true perspective drawing".

To make the picture look traditionally beautiful, place the object slightly to the left of you and slightly below your eye level, so all the X and Y coordinates become negative.

Computerize the process

You can program the computer so that if you input a point's X coordinate, Y coordinate, and Z coordinate, the computer will calculate the projected X coordinate (from dividing X by Z) and the projected Y coordinate (from dividing Y by Z) and plot the point on the computer's screen (by using high-resolution graphics).

Digital cameras

Here's what Walmart charged for digital cameras when this book went to press in December 2016:

Maker	Model	Megapixels	Optical zoom	Price
Vivitar	Red V7122	7.1	4x	\$35
Polaroid	IE826-RED-WM	18	8x	\$50
Polaroid	iEX29-RED	18	10x	\$59
Sony	DSC-W830	20	8x	\$119
Kodak	AZ251	16.15	25x	\$129
Kodak	AZ501	16.15	50x	\$156

Professional retouching

Many women wish they were more beautiful. They buy lots of makeup. Some even undergo cosmetic surgery. But here's an easier way to produce a photo showing the woman is beautiful: just take a natural photo of her and edit it! That's called **retouching** the photo.

Many folks make their living by being **retouchers**: they retouch photos. They remove a woman's pimples, wrinkles, and fat; and they reshape her face to give her fuller lips (so she looks kissable), open eyes (so she looks excited), a bigger forehead (so she looks intelligent — and also has her eyes closer to her chin), and a smaller nose, mouth, and jaw (so she looks dainty and vulnerable), all making her look closer to 24.8 years old, which statisticians have shown is the female age that attracts men most (because women have the most estrogen then). The retouchers can work similar magic on men's faces, too! Their accomplishments raise the question of what "beauty" means: if you edit out your uniqueness, do you become more beautiful or just a clone of a bright electrified kissing doll?

To see the work of an expert retoucher (Canada's **Christiane Beaulieu**), go to her Website (**www.CBeau.ca**) then click her name then click your favorite language (FRANCAIS or ENGLISH). You see 30 retouched photos, darkened. Click one of the photos; then you see it enlarged and brightened. Click the photo again, to see the original, unretouched version of it. Click it several more times, to flip back and forth between the retouched and unretouched versions of it. When you tire of that photo, click the black margin at its left; then you'll see all 30 photos again, so you can click a different one of them instead.

For further fun, gawk at **Portrait Professional Picture Gallery** (**www.PortraitProfessional.com/gallery**). You see 15 retouched photos (if you scroll down). Click one of them to see it enlarged, then point at the enlargement (by moving the mouse there without clicking) to see the original, unretouched version of it. (Move the mouse off the enlargement to see the retouched version again.) Each of those retouchings was invented in less than 5 minutes by using the **Portrait Professional** program. If you click "buy", you can buy a long-term license to transmit *your own photos* to the company's Website, which will edit your photos and transmit the retouched versions back to you. By using sliders, you choose *how much* you want your own photos retouched, so you can look a bit "dolloed up" without looking totally plastic.

To see movies about how to retouch women (and criticize the morality of it), look at these Web sites on **YouTube** —

Dove Evolution	www.YouTube.com/watch?v=iYhCn0jf46U
Extreme (Photoshop) Makeover	www.YouTube.com/watch?v=aHLpRxAmCrw
Photoshop Effect	www.YouTube.com/watch?v=YP31r70_QNM
Doll Face	www.YouTube.com/watch?v=zl6hNj1uOkY

and this video of a Hungarian woman nicknamed "Boggie" singing, in French, "New Perfume" ("Nouveau Parfum") and "I'm not their product" ("Je ne suis pas leur produit"), while she gets edited and turned into a product that's pretty white trash:

www.TheAwesomer.com/boggie-nouveau-parfum/266551

Web-page design

When using the Internet's World Wide Web, don't be just a looker; be a creator! Create your *own* Web pages and let everybody else in the world see them!

Angelfire

The easiest way to create your own Web pages is to use a Web site called **Angelfire**. It's free!

Angelfire is at www.angelfire.com. It used to be an independent company, but now it's owned by **Lycos**. (which also owns a similar site, **Tripod**, at www.tripod.com).

Restrictions

Angelfire lets you create any Web pages you wish, as long as you keep them "clean", so they don't contain content or links to anything that's:

unlawful, harmful, hateful, harassing, stalking, or containing viruses
defamatory, libelous, ethnically objectionable, or pirated (copied without permission)
privacy-invading (or vulgar or pornographic), especially if involving minors or viewable by minors
required to stay private (by copyright laws or an employer's nondisclosure agreement)
selling explosives, weapons, securities, or non-existent goods
selling alcohol, tobacco, controlled drugs (or pharmaceuticals), or unpackaged food
advertising gambling (or raffles requiring a fee) or pyramid schemes

Unfortunately, Angelfire will automatically put an ad on your Web page and restrict you (to 20 megabytes of Angelfire's disk space, with a limit of 1 gigabyte of transfers per month between Angelfire's disk and people viewing your Web page), unless you pay extra (to get a fancier plan):

Plan's name	Cost	Disk space	Bandwidth	Forced ad?	Get your own .com, such as joe.com?
Free	free	20M	1G/month	yes	no
Entry	\$1/month	40M	2G/month	yes	no
Basic	\$3/month	100M	5G/month	no	no
THE Plan	\$10/month	5G	1T/month	no	yes

Create an account

Using your Web browser (such as Microsoft Internet Explorer), go to **www.angelfire.com**. Click "Try It Now for Free!" then "Sign up for one of these plans today!" (which you see when you scroll down).

Click in the "Your New Website" box. Your Website will be named "http://____.angelfire.com". Invent a name to put in the blank. The name cannot contain capital letters or spaces: it must be made of just lower-case letters, digits, dashes, and periods. Type the name you invented (such as "secretguide") then click in the Password box. If the name you invented is okay, the computer says "User name Available"; otherwise the computer says "User name Unavailable" (probably because somebody else picked that name) and you must try again to pick a name.

Next, invent a password (which must be at least 6 characters long, with no spaces). Type the password in the Password box, press the Tab key, type the password again, click "Next", and click "Next" again.

The computer will say "Welcome". Answer the questions about your birthday, security, usage, gender, and newsletters, then click "Confirm".

Angelfire Customer Service will send you an email whose subject is "Welcome to Angelfire". Find it in your email's Inbox. Click the first link in the email.

The computer will say "Your account has been activated". Click "Start Building Now" then "Create a New Website".

Change the text

You're using a Website-creation program called **Webon** (pronounced "web on"). It shows a Web page containing 5 blocks of text. In each block of text, switch those words to *your* words. Here's how to do that:

Drag across the first block of text, which says "MY WEBSITE". Type whatever **heading** you want instead, such as "JOAN'S HOME PAGE". Your typing will be automatically capitalized.

Drag across the second block of text, which says "My website's **subheading**". Type whatever subheading you want instead, such as "Made with love".

Drag across the third block of text, which begins "This is your **main content section**. You should delete..." Type the main message you want instead, such as "I was born yesterday. I want to die."

Drag across the fourth block of text, which begins "This is your **sidebar**. This sidebar is..." Type the sidebar message you want instead, such as "We won't reply to emails addressed to my goldfish."

Scroll down to see the fifth block of text, which begins "This is your **footer**. You can delete..." Drag across that block of text, then type the footer you want instead, such as "Copyright by a wronged woman."

Format the text

You can easily format the text in blocks 3, 4, and 5. To do that, drag across the phrase you want to change (so the phrase temporarily appears in white letters on a blue background), then do one of these things:

Click one of the formatting buttons: Bold, Italic, Underline, Align Left, Align Center, Align Justify, Align Right, Ordered List (which means a numbered list), or Unordered List (which means a bulleted list).

For the Font Size box (which normally says "12px", which means 12 pixels high), click its down-arrow then click a different number of pixels instead. Your choices are 8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 28, 32, 36, 48, and 72.

For the Font box (which normally says "Arial"), click its down-arrow then click a different font instead. Your choices are Arial, Times New Roman, Verdana, Georgia, Trebuchet, Courier New, Tahoma, Palatino Linotype, Impact, and Comic Sans.

For the Color box (which is normally black), click its down-arrow then click a different font color instead. The phrase will appear in the new color when you finish highlighting the phrase: click elsewhere.

For the Background Color Box (which says "BG" and is normally white), click its down-arrow then click a different background color instead. The phrase will have the new background color when you finish highlighting the phrase: click elsewhere.

Notice that to format a phrase, you must drag across the phrase *beforehand*. For example, if you want a phrase to be bold, you must drag across the phrase *before* you click the Bold button.

Undo

If you make a typing mistake, here's how to undo it: while holding down the Ctrl key, tap the Z key.

That method undoes your last typing mistake, but it can't undo your last *two* typing mistakes, and it can't undo formatting. If you format a phrase wrong, format it again correctly.

Change the style

You've been using a style called **Working Comp**. To use a different style instead, click "Styles" (which is at the screen's top). You start seeing a list of 152 styles, in alphabetical order. To see the rest of the list, click "Next" several times. To go back toward the list's beginning, click "Previous" several times. If a style interests you, put the mouse pointer on it without clicking; then you see a slightly larger picture of the style.

Click whichever style you like. Then you see the words you wrote, reformatted to fit in that style.

If you don't like that style, try clicking "Revert", which takes you back to a style you used before (such as Working Comp).

When you've finished picking a style, click "Text" (which is near the screen's top-left corner).

Add links

Here's how to make the phrase "house hunting" be underlined and link to www.realtor.com:

Type "house hunting". Drag across that phrase (so it's highlighted). Click "Link". Click after the "http://". Type "www.realtor.com". Click the green "Create" button.

This book was written by Russ Walter, whose email address is Russ@SecretFun.com. Here's how to make your Web page let people send an email to Russ Walter, by clicking "write to Russ":

Type "write to Russ". Drag across that phrase (so it's highlighted). Click "Link" then the down-arrow then "an email address". Press the Tab key. Type the email address "Russ@SecretFun.com". Click the green "Create" button.

Then when a person accesses your Web page, "write to Russ" will be underlined. If the person clicks "write to Russ", the computer will automatically run the person's email client program (such as Outlook Express), automatically click "Create Mail", automatically type "Russ@SecretFun.com" in the "To" box, and wait for the person to type an email message to Russ.

Final steps

When you finish editing your Website, click "Save" (which is near the screen's top-right corner) then "Publish" (which is next to "Save") then "Publish to the main page of the site" then the green "Publish" button. The computer says, "Congratulations! Your site was published..."

If you want to edit further, click "Keep Working"; otherwise, exit by doing this:

Click "Back to Angelfire" then "logout" (which is near the screen's top-right corner).

Edit your site

To edit a Website you created before, do this:

Go to www.angelfire.com. Click "Login" (which is at the top). Type your user name (such as "secretguide"), press the Tab key, type your password, then click the green "Log In" button. Click "Edit" (which is below "Create a New Website").

You see your Website. Edit it, then do the "Final steps" procedure again.

Extra pages

Here's how to put extra pages onto your Website.

While you're editing the first page you created, click "Pages" (which is at the top) then the "Create a new page" icon. (That icon is near the screen's top-left corner, under "Text", and shows a single sheet of paper with a green plus sign.)

Invent a title for the page (such as "My Family"); type it and press Enter.

Now the screen's left edge shows you have 2 pages. The first page (which you created before) was automatically called "Home". If the new page you're creating is called "My Family", the screen's left edge shows this list of pages:

1. Home
2. My Family

If you want to create a 3rd page, click the "Create a new page" icon again, type the new page's title (such as "Our Friends") and press Enter again, so the screen's left edge looks like this:

1. Home
2. My Family
3. Our Friends

The screen's bottom is still devoted to showing what you typed on page 1. To switch your view to page 2 or page 3, click the page's name in that list of pages, then click the "Jump to a page" icon (which is a sheet of paper with a green arrow). That makes the screen's bottom show the page you requested. The heading you requested is at the screen's top and has the page name you invented. There's no subheading. The sidebar is at the left instead of the right. The main content section, sidebar, and footer are all blank, waiting for you to type your words there. Below the heading you also see this **navigation bar** —

HOME MY FAMILY OUR FRIENDS

which your readers can click on to jump from page to page, after you've finished creating your Website.

To see normal editing tools again (such as the Bold button), click "Text" (which is near the screen's top-left corner).

HTML

Every page on the Internet's Web is written in a computer language called the **HyperText Markup Language (HTML)**.

Warning: if someone claims to know "HTML", it could mean "HyperText Markup Language" or "How To Make Love". Ask which!

HTML uses these commands:

HTML command	Ending	Page
		299
		299
		297
<big>	</big>	297
<body>	</body>	298
<body bgcolor=red>	</body>	300
 		298
		300
		297
		297
<form method=...>	</form>	302
<h1>	</h1>	297
<head>	</head>	298
<html>	</html>	298
<i>	</i>	296
<input name=city>		302
<input type=button ...>		628
<input type=checkbox ...>		303
<input type=radio ...>		303
<input type=reset>		303
<input type=submit>		303
		298
<link rel=...>		301
<noscript>	</noscript>	628
		299
<p>		297
<pre>	</pre>	299
<script>	</script>	623
<script language=...>	</script>	628
<small>	</small>	297
<style>	</style>	300
_		297
[]	297
<table border=1>	</table>	299
<td>		299
<th>		299
<title>	</title>	297
<tr>		299
<tt>	</tt>	297
		298
<!DOCTYPE HTML ...>		298
<!--I was drunk-->		302

HTML uses these styles:

Style command	Page
body {background:yellow}	300
body {color:red}	300
body {font-size:13pt}	300
h1 {color:blue}	300
h1 {font-size:40pt}	301
h1 {text-align:center}	301
p {margin-bottom:0}	301
p {margin-top:0}	301
p {text-indent:2em}	301
:hover {color:navy}	300
:link {color:green}	300
:visited {color:fuchsia}	300

Simple example

Suppose you want to create a Web page that says:

We *love* you

Notice that the word “love” is italicized (slanted).

To create that Web page, write this HTML program:

we <i>love</i> you

Here’s what that program means:

The <i> is an **HTML tag** that means “italics”. Each HTML tag is enclosed in the symbols <>, which are called **angle brackets**.

The </i> is an HTML tag that means “end the italics”, because the symbol / means “end the”. The <i> shows where to begin the italics; the </i> shows where to end the italics.

To type that program, you can use two free methods: **Notepad** or **Angelfire**. I’ll explain both.

Typing in Notepad

If your computer contains Windows, the easiest way to type that HTML program is to use **Notepad**, as follows...

Launch Notepad. Here’s how:

Windows 10 In the Windows Search box, type “notep”. You see a list of things that contain “notep”. Tap “Notepad: Desktop app”.

Windows 8&8.1 Go to the Apps screen (or Windows 8’s Start screen) and type “no”. You see a list of programs whose names have “no”. Tap “Notepad”.

Windows 7 Click Start then Programs then Accessories then Notepad.

You see the Notepad window. Make it consume the screen’s bottom right quarter. Here’s how:

If the Notepad window is maximized (consumes the whole screen), make the window smaller by clicking the restore button (which is next to the X button).

Drag the window’s bottom-right corner to the screen’s bottom right corner. Drag the window’s top-left corner to the screen’s center. Then the window consumes the screen’s bottom right quarter.

Make Notepad do word wrapping. Here’s how:

Click “Format”. You see “Word Wrap”. If there’s no check mark in front of “Word Wrap”, put a check mark there (by clicking “Word Wrap”).

Click in the middle of the Notepad window. Type the HTML program, like this:

we <i>love</i> you

Save the program Save the program onto your desktop, and give it a name that ends in “.html”. Here’s how:

Click “File” then “Save”.

Click the word “Desktop”, which you’ll see at the screen’s left edge (after scrolling up, if necessary). Click in the “File name” box.

Type any name ending in “.html” (such as “joan.html”); to be safe, use just small letters (no capitals, no spaces). Press Enter.

Your program’s icon arrives on the desktop but might be covered by the Notepad window. Move the Notepad window (by dragging its blue title bar) until you see your program’s icon.

The icon has an “e” on it. The “e” means the program works with Microsoft Edge (or Microsoft Internet Explorer). It works with Microsoft Edge (or Microsoft Internet Explorer) because the program is written in HTML and ends in .html.

Run the program Double-click your program’s icon. That makes the computer run your program. (If a “Connect To” window appears, click “Cancel” once or twice, to make the “Connect To” window go away.)

If your computer’s been set up properly (to use Microsoft Edge or Microsoft Internet Explorer as the main Web browser), you’ll see a window that shows the result of running your program; it shows a Web page that says:

We love you

Edit the program If you typed and ran the program

recently, here’s how to edit it.

At the screen’s bottom, to the right of the Start button, you see a wide button for Notebook. (In Windows 10, the button is underlined.) Click it. That makes the Notebook window appear and be the active window. Then make any changes you wish to your HTML program. For example, if you want the Web page to say “We *tickle* you”, change “love” to “tickle” (by dragging across “love” and then typing “tickle”).

Go ahead: make that change and any other changes you wish! Experiment! Go wild!

After you’ve edited the program, save the edited version (by clicking “File” then “Save”).

To run that program, click in the Microsoft Internet Explorer (or Microsoft Edge) window (which hides behind the Notepad window), so that Microsoft window becomes the active window. Then click the Refresh button (a circling arrow), which makes Microsoft re-examine your program and run your program’s new version.

Edit old programs Here’s how to edit an old program that you haven’t typed or run recently.

Find the program’s icon on the desktop. Right-click that icon. Click “Open with”.

If you see “Notepad”, click it. Otherwise, do the following:

Windows 10 Tap “Choose another app” then “More apps” then “Notepad”. (Don’t put a checkmark before “Always use this app to open .html files”.) Tap “OK”.

Windows 7&8&8.1 Click “Choose default program”. Remove the check mark from “Always use the selected program” (by clicking). Click the down-arrow to the right of “Other Programs”. Scroll down until you see “Notepad” then double-click “Notepad”.

Make the Notepad window consume the screen’s bottom right quarter. Make Notepad do word wrapping.

Then you see your HTML program. Edit it. When you finish editing, click “File” then “Save”.

Typing in Angelfire

Here’s how to type an HTML program by using **Angelfire** instead of Notepad...

Go to www.angelfire.com. Start creating a Web page, using the methods I described on pages 294-295.

While you’re typing & editing the main content section, click “Add-ons” (which is at the top). You see 27 add-ons. Drag “Custom HTML” (which is the last add-on) until it’s below the typing in the main content section and is in the middle of a big black box that suddenly appears.

After several seconds, the computer says this sentence:

Click the “Edit” Link to add HTML to this space.

Move the mouse pointer to that sentence, without clicking. Then above that sentence, you see “Edit”. Click that “Edit”.

The computer says this sentence:

Paste or write your custom HTML in the box below.

In the box below that sentence, drag across the sentence that’s already there (which begins with “<p>”) then type your HTML program instead. For example, type:

we <i>love</i> you

When you finish typing your program in the box, click the green “Save” button (which you must scroll down to see).

Then you see a box showing the result of your programming:

We love you

If you want to edit your program further, move the mouse pointer to that box (without clicking), then click the “Edit” above that box, then edit your program further, then click the green “Save” button again (after scrolling down to see it).

Simple HTML commands

Here are simple HTML commands you can give.

Italicize To make a phrase be italicized (*like this*), type `<i>` before the phrase; type `</i>` after the phrase.

Bold To make a phrase be bold (**like this**), type `` before the phrase; type `` after the phrase. To make a phrase be bold italic (***like this***), type `<i>` before the phrase; type `</i>` after the phrase.

Paragraph If your document contains more than one paragraph, put `<p>` at the beginning of each paragraph. For example, if you want a paragraph to begin by saying “Motherhood is maddening!”, begin the paragraph by typing this:

```
<p>Motherhood is maddening!
```

At the end of each paragraph, press the Enter key twice. Then when you run the program, the computer will put a blank line below each paragraph.

Title To create a title, type `<title>` before it and `</title>` after it. For example, to make your title say “Joan’s Home Page”, type this:

```
<title>Joan's Home Page</title>
```

When you run the program, the computer will put the title in the blue **title bar** at the Web page’s top.

Fonts

You learned to italicize by saying `<i>` and `</i>`, and to create bold by saying `` and ``. Here are other ways to change the text’s font....

Size You can make the computer can produce text in 7 sizes. Font size 1 is the smallest; Font size 7 is the biggest. Here’s how big they are:

This is font size 1. It is 8 points tall.

This is font size 2. It is 10 points tall.

This is font size 3. It is 12 points tall.

This is font size 4. It is 14 points tall.

This is font size 5. It is 18 points tall.

This is font size 6, 24 points tall.

Font size 7, 36 points.

That’s how big the font sizes and point size look on paper and a 14-inch monitor. (On a slightly bigger monitor, the font sizes and point sizes look slightly bigger.)

Normally, the computer makes your Web page have font size 3. To make a phrase be font size 5, say `` before the phrase; say `` after the phrase.

You can make part of a phrase be one size bigger, by saying `<big>` before that part and saying `</big>` after that part. For example, if the computer is making a phrase be font size 5 because you said ``, you can make part of that phrase be slightly bigger (font size 6) by saying `<big>` before that part and saying `</big>` after that part. To make part of a phrase be one size smaller, say `<small>` before that part and say `</small>` after that part.

If you said `` and then try to say `<big>`, the computer will ignore the `<big>`, since the computer can’t go bigger than font size 7. If you said `` and then try to say `<small>`, the computer will ignore the `<small>`, since the computer can’t go smaller than font size 1.

To make a phrase be subscript (*like this*), say `_{` before the phrase, `}` afterwards. To make a phrase be superscript (*like this*), say `^{` before the phrase, `}` afterwards.

Heading To create a heading (such as a chapter title or a newspaper headline), say `<h1>` at the heading’s beginning and `</h1>` at the heading’s end, like this —

```
<h1>Chapter 2: Laura giggles at death</h1>
```

or like this:

```
<h1>USA declares war on hangnails</h1>
```

To do that, make sure you type the symbol “<”, then the letter “h”, then the number one, then the symbol “>”, then the rest.

The `<h1>` makes the computer automatically create a new paragraph in font size 6 and bold. Saying `<h1>` is similar to saying `<p>`.

If you want the heading to be centered (instead of at the screen’s left edge), say `<h1 align=center>` instead of just `<h1>`.

If you say `<h2>` instead of `<h1>`, the computer will make the font size slightly smaller (5 instead of 6), so you’ll be creating a “less dramatic heading”, a subheading. If you say `<h2>` at the heading’s beginning, say `</h2>` at the heading’s end.

You can create headings in 6 sizes:

```
<h1> produces font size 6
<h2> produces font size 5
<h3> produces font size 4
<h4> produces font size 3
<h5> produces font size 2
<h6> produces font size 1
```

To create a heading in font size 7, say `<h1><big>` at the heading’s beginning, and say `</big></h1>` at the heading’s end.

Tt Normally, the computer makes the text’s typeface be Times New Roman (which looks like this). If you want a phrase’s typeface to be Courier New (which looks like this and imitates a typewriter and a Teletype), say `<tt>` before the phrase, `</tt>` afterwards.

Arial If you want a phrase’s typeface to be Arial (which looks like this), say `` before the phrase, `` afterwards.

If you want a phrase to be font size 7 and Arial, say `` before the phrase; `` afterwards.

Compliance

If you want to create a Web page whose title is “Joan’s Home Page” and whose body says “We *love* you”, you can write this HTML program:

```
<title>Joan's Home Page</title>
we <i>love</i> you
```

Although that program works with *most* versions of *most* Web browsers, you’re supposed to add some extra lines, for 3 reasons:

to help other programmers understand your program
to make sure the program works with *all* browsers
to prevent getting fired from your programming job

Head & body You’re supposed to divide the program into 2 parts. The first part, called the **head**, begins with `<head>` and ends with `</head>` and includes the title. The second part, called the **body**, begins with `<body>` and ends with `</body>` and includes the paragraphs. So your program should look like this:

```
<head>
<title>Joan's Home Page</title>
</head>

<body>
we <i>love</i> you
</body>
```

(If you’re typing into the middle of an Angelfire page, do *not* type a head, since Angelfire gave the page a head already.)

The `<html>` warning To make your program even better, you’re supposed to begin the whole program by saying `<html>` and end the whole program by saying `</html>`, to emphasize that the whole program is written in HTML rather than a different computer language. So your program should look like this:

```
<html>
<head>
<title>Joan's Home Page</title>
</head>

<body>
we <i>love</i> you
</body>
</html>
```

Is code compliant? You should include all those extra lines — `<head>`, `</head>`, `<body>`, `</body>`, `<html>`, and `</html>` — to make your program comply with the standards that people expect. Those lines help make your program be **compliant**.

I usually don’t bother including those lines, since I’m my own boss; but if you’re employed, you should include those lines to keep your job.

Title To be compliant, your program must include a `<title>` line.

DOCTYPE To be compliant, you’re supposed to also put this line at your program’s top, above the `<html>` line:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
```

That line brags that your program document is of this type: it obeys the HTML standard, which is publicly available from the **World Wide Web Consortium (W3C)**, using the **document type definition (DTD)** for HTML version 4.01 — except that you’re letting yourself include some older HTML commands also, to be transitional and help older Web browsers understand your Web page; and you’re doing all this in **English (EN)**.

Actually, you’re supposed to include another line below that, saying what Web site reveals the HTML 4 and 4.01 specifications; so your program is supposed to begin like this:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

Hardly anybody bothers to include those two lines about DOCTYPE. For example, Yahoo’s Web site (www.yahoo.com) and Microsoft’s Web site (www.microsoft.com) omit both lines.

Even the inventors of HTML 4.01 don’t bother including the second line in their own Web pages.

XHTML HTML 4.01 was invented in 1999. Now committees are working to develop fancier programming language, called the **eXtensible HTML (XHTML)**, which is influenced by the **eXtensible Markup Language (XML)**.

If you want to be compliant with XHTML, adopt these habits:

In commands, use small letters (such as `<p>`) instead of capitals (such as `<P>`).

At the end of each paragraph, say `</p>`.

After each equal sign, put quotation marks; so instead of saying ``, say ``.

In XHTML, the top lines say:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"DTD/xhtml1/1-transitional.dtd">
```

Lists

If you want your Web page to contain a simple list, say `
` at the beginning of each list item.

For example, suppose you want your Web page to say:

Here are the favorite flavors:

chocolate
vanilla
strawberry

To do that, say `<p>` at the beginning of the paragraph, `
` at the beginning of each list item, like this:

```
<p>Here are the favorite flavors:
<br>chocolate
<br>vanilla
<br>strawberry
```

The `
` stands for “break out a new line”.

In XHTML, instead of saying `
` you must say `
` and make sure to put a blank space before the slash.

Bullets Suppose you want your list to show bullets, like this:

Here are the favorite flavors:

- chocolate
- vanilla
- strawberry

To do that, say `<p>` at the beginning of the paragraph, `` at the beginning of each list item, like this:

```
<p>Here are the favorite flavors:
<li>chocolate
<li>vanilla
<li>strawberry
```

Suppose you want the bullets to be indented, with a space above the list, like this:

Here are the favorite flavors:

- chocolate
- vanilla
- strawberry

To do that, say `` above the list, `` below the list, like this:

```
<p>Here are the favorite flavors:
<ul>
<li>chocolate
<li>vanilla
<li>strawberry
</ul>
```

The `` stands for “unordered list”. When you type ``, make sure you press the U and L keys on your keyboard (not the number one).

The accomplishes two goals:

It makes the list be indented.
It makes your program be compliant.
(A list without is *not* compliant.)

In XHTML, the end of each list item must say .

Numbers Suppose you want your list to be numbered, like this:

Here are the favorite flavors:

1. chocolate
2. vanilla
3. strawberry

To do that, say above the list, below the list, like this:

```
<p>Here are the favorite flavors:
<ol>
<li>chocolate
<li>vanilla
<li>strawberry
</ol>
```

The stands for “ordered list”. When you type , make sure you press the O and L keys on your keyboard.

Tables

You can create a simple table or a fancy table.

Simple table Here’s how to create a simple table:

Tell the computer the text is preformatted, by saying <pre>. Then type the table, by using the Space bar and Enter key to line up the columns. Below the table, say </pre>.

For example, suppose you want your Web page to say:

Here are the bowling scores:

NAME	SCORE
Jacqueline	200
Ann	137
Ed	75

To do that, say <pre> above the table and </pre> below the table, like this:

```
<p>Here are the bowling scores:
<pre>
NAME      SCORE
Jacqueline 200
Ann        137
Ed         75
</pre>
```

The <pre> makes the computer use Courier New, as if you had typed <tt>. Courier New is a convenient font for tables, because it makes every character and space have the same width, so you can easily align the table’s columns by pressing the Space bar several times.

The <pre> also makes the computer copy each Enter and Space onto the Web page, unedited. (If you don’t say <pre>, the computer turns each Enter into a Space and turns each pair of Spaces into a single Space.)

Fancy table Let’s create a fancy table, so the Web page says:

Here are the bowling scores:

NAME	SCORE
Jacqueline	200
Ann	137
Ed	75

To do that, say <table border=1> above the table, <tr> at the beginning of each table row, <th> at the beginning of each column heading, <td> at the beginning of each data item, and </table> below the table, like this:

```
<p>Here are the bowling scores:
<table border=1>
<tr><th>NAME<th>SCORE
<tr><td>Jacqueline<td>200
<tr><td>Ann<td>137
<tr><td>Ed<td>75
</table>
```

The computer automatically makes the columns wide enough to hold their headings and data. The computer automatically makes the column headings be bold and centered.

The border=1 makes the table have a normal border. If you say border=12 instead, the table’s outer border will be much thicker and shaded, forming a beautiful 3-D picture frame that makes the table seem to pop out from the screen and into the human’s face. Try it! You’ll impress your friends!

Links

You learned:

To make a phrase be italicized, say <i> before the phrase, </i> afterwards.
To make a phrase be bold, say before the phrase, afterwards.

Link to other Web sites To make the phrase “house hunting” be underlined and link to www.realtor.com, say this:

```
<a href=http://www.realtor.com>house hunting</a>
```

Notice that before “house hunting”, you say what “house hunting” is linked to, by saying , which means “anchor (link) with a hypertext reference to http://www.realtor.com”. After “house hunting”, say .

Make sure you include the “http://”. If you omit that, the link doesn’t work.

Don’t insert extra spaces. For example, don’t insert a space after “http.”; don’t insert a space after “/”.

Most text on your Web page is black (on a white background). Links are underlined and typically blue; but that if the link refers to a Web page that was visited recently, the link turns purple (instead of blue).

Link to your own Web pages To make the phrase “funny jokes” be underlined and link to Web page you created and called “jokes.html”, say this:

```
<a href=jokes.html>funny jokes</a>
```

That works just if you earlier invented a Web page named jokes.html and put it on the same disk and in the same folder as the new Web page you’re inventing. For example, if you’re inventing a new Web page on your hard disk’s Desktop, jokes.html must also be a Web page on your hard disk’s Desktop.

Suppose you create a Web page that’s too long to fit on the screen, so when a human tries to read the page the human must scroll down. To let the human avoid scrolling, do this: near the page’s beginning, put a table of contents that links to later parts of the page. Here’s how. To make the phrase “my joys” be underlined and link to a part of the page that discusses your joys, say this:

```
<a href=#joys>my joys</a>
```

Tell the computer which part of your page discusses your joys, by beginning that part like this:

```
<a name=joys></a>
```

Link to email This book was written by Russ Walter, whose email address is Russ@SecretFun.com. On your Web page, you can let people write to Russ Walter. Here's how...

To make the phrase "write to Russ" be underlined and link to creating an email to "Russ@SecretFun.com", say this:

```
<a href=mailto:Russ@SecretFun.com>write to Russ</a>
```

Then when a person accesses your Web page, "write to Russ" will be underlined. If the person clicks "write to Russ", the computer will automatically run the person's email client program (such as Outlook Express), automatically click "Create Mail", automatically type "Russ@SecretFun.com" in the "To" box, and then wait for the person to type an email message to Russ.

Try it! Go put that in your Web page, run your Web page, and write an email to Russ! If you don't know what to say in the email, just say, "Hi, Russ, I'm testing my Web page's email link to you."

Be brave! Let people accessing your Web page write an email to you. For example, if your name is Joan Smith and your email address is "jSmith@SecretFun.com", say this:

```
<a href=mailto:jSmith@SecretFun.com>write to Joan Smith</a>
```

Custom colors

You can change colors.

Change a phrase Normally, the computer makes your text be black. To make a phrase be red, say before the phrase; say after the phrase.

The computer knows the names of 16 colors:

Light color	Dark color
red	maroon (dark red)
blue	navy (dark blue)
aqua (greenish blue)	teal (dark aqua)
lime (bright green)	green (darker than lime)
fuchsia (bright purple)	purple (darker than fuchsia)
silver (light gray)	gray (darker than silver)
yellow	olive (dark yellow, looks greenish brown)
white	black

If you want a phrase to be font size 7 and red, say before the phrase; say after the phrase.

Change the whole Web page On a normal Web page, the background is white, the text is black, each typical link is blue, and each viewed link is purple.

Your program is supposed to be divided into 2 parts, called <head> and <body>. Instead of saying just <body>, you can say:

```
<body bgcolor=yellow text=red link=green vlink=fuchsia>
```

That makes the background be yellow, the text be red, each typical link be green, and each viewed link be fuchsia. Choose any colors you wish, from the list of 16 colors.

Usually, the background should be a light color (such as white or yellow) and the text should be a dark color (such as black). If you want to be shockingly different, do the reverse: make the background be a dark color (such as black or navy) and make the text be a light color (such as white or yellow). For example, try this:

```
<body bgcolor=navy text=white link=lime vlink=yellow>
```

Warning: the human eye gets dizzy when it sees red next to blue, so don't choose "red text on a blue background" or "blue text on a red background". If you make the mistake of choosing those combinations, the people viewing your Web site will get dizzy and fall asleep without reading your words. Your Web site will put them into a hypnotic trance. The human eye also has difficulty reading red (or maroon) next to black.

Style sheets

To change the appearance of your whole Web page, create a **style sheet**.

For example, suppose you want red characters on a yellow background. You learned you can do that by changing <body> to this:

```
<body bgcolor=yellow text=red>
```

Here's a better way: in your <head> section, below the <title>, say this:

```
<style>
body {color:red; background:yellow}
</style>
```

That creates this style: throughout the body, make the text color be red, the background yellow. That makes most of the text be red (though normal links will still be blue underlined and the viewed links will still be purple underlined). When you type the second line, make sure you type braces, which look like this: {}. Don't type parentheses, which look like this: ().

Normally, the text is font size 3, which is 12 points. If you want the text to be slightly bigger (so people can read it more easily), request a bigger point size, such as 13 points, by saying font-size=13pt, like this:

```
<style>
body {color:red; background:yellow; font-size:13pt}
</style>
```

That makes most text get bigger, but headings will be unchanged. For example, <h1> headings will still be font size 6 (which is 24 points).

Should you use style sheets? Style sheets were invented recently. They're new, hip, cool, and recommended.

For example, to get a yellow background, you ought to say background:yellow in the style sheet, rather than bgcolor=yellow in the <body>.

Using style sheets is **recommended**. Giving older types of commands, such as bgcolor=yellow, is **deprecated** (which means "pooh-pooed").

But many people still use older types of commands, such as bgcolor=yellow, since they work even on old computers whose browsers were invented before style sheets.

Links You can change the color of links:

```
<style>
body {color:red; background:yellow}
:link {color:green}
:visited {color:fuchsia}
:hover {color:navy}
</style>
```

That makes most links be green, recently visited links be fuchsia, and each link temporarily turn navy while the mouse hovers over it.

Headers You learned that you can create big headers by saying <h1>, smaller headers by saying <h2>, and even smaller headers by saying <h3>, <h4>, <h5>, and <h6>. Normally, headers are the same color as the body text. For example, if you made the body text be red, the headers are automatically red also.

To make <h1> headers be blue and all other headers be maroon, say so in the style sheet, like this:

```
<style>
body {color:red; background:yellow}
:link {color:green}
:visited {color:fuchsia}
:hover {color:navy}
h1 {color:blue}
h2,h3,h4,h5,h6 {color:maroon}
</style>
```


To make <h1> headers be blue and also centered (instead of at the screen's left edge), make the style sheet's h1 line be this:

```
h1 {color:blue; text-align:center}
```

Frankly, I hope you don't choose those colors! Your Web page will be too wild if you actually make the body text red, the background yellow, the links green, the visited links fuchsia, the hovered links navy, the big headings blue, and the smaller headings maroon. Choose more reasonable colors.

Normally, <h1> headers are font size 6, which is 24 points. To make <h1> headers be even bigger, give a bigger point size, such as by saying:

```
h1 {color:blue; text-align:center; font-size:40pt}
```

Paragraphs You learned to put <p> at the beginning of each paragraph. Normally, the <p> makes the computer put a blank line above the paragraph. If you want the computer to omit the blank line and indent the paragraph's first word, put this line in your style sheet:

```
p {text-indent:2em; margin-top:0; margin-bottom:0}
```

Sharing Several Web pages can share a style sheet. Here's how. Using Notepad, create the style sheet, but omit the <style> and </style> lines. For example, create this style sheet:

```
body {color:red; background:yellow}
```

Save it on your Desktop, but give it a name that ends in .css (which stands for Cascading Style Sheet) instead of .html. For example, name it mystyle.css.

Then create your Web pages; but on each Web page, tell the computer to use the style sheet you created (mystyle.css), by putting this line in the <head> part of the Web page:

```
<link rel=stylesheet href=mystyle.css>
```

That tells the computer to create a link, related to your style sheet, which is located at the HTTP reference "mystyle.css".

Upload

To let the public use your Web page, make sure your Web page is on the Internet. Here's how.

If you created the Web page by using Angelfire's Webon, make sure you published your page (by clicking the blue "Save" and "Publish" buttons at Webon's top-right corner).

If you created the Web page by using Notepad, you must **upload** your page (copy it from your computer's hard disk to an Internet-connected hard disk). To upload your page, you need permission from a Web host, which will usually charge you a monthly fee. For example, you can upload to Angelfire if you pay Angelfire \$1 per month to get Angelfire's **Entry service** (which is better than Angelfire's Free service). If you've upgraded to Angelfire's Entry service, here's how to copy your Web page to Angelfire's hard disk:

Using your Web browser, go to "www.angelfire.com". Click "Login" (which is at the top). Type your user name (such as "secretguide"), press the Tab key, type your password, then click the green "Log In" button. Click "Upload Files" (which is at the screen's center and isn't included in the Free version). Click the first "Browse" button.

You see a list of files that are on your computer's hard disk. Double-click the file you want to upload, such as "joan". (If you don't see that file in the list, try making it appear by clicking the word "Desktop" and using the scroll arrows.)

Click the green "Upload" button (which you see when you scroll down).

That file will now be on Angelfire's disk. For example, if your user name is "secretguide" and the file was joan.html, it's now available on the Internet as "http://secretguide.angelfire.com/joan.html". If the file was index.html, it's now available on the Internet as "http://secretguide.angelfire.com/index.html" or more simply as "http://secretguide.angelfire.com" (since "index.html" is your main page).

If your Web page includes links to other files (such as other Web pages you created and style sheets), make sure you upload those files also.

If you're too cheap to pay Angelfire \$1 per month, try this crude trick:

Create a free Web page by using Angelfire's Webon. On that page, start the processing of typing your own HTML, as I explained in the section called "Typing in Angelfire". But instead of typing lots of HTML there, copy your typing from Notepad to there (by dragging across the HTML you typed in Notepad, then pressing Ctrl with C, then clicking in Angelfire's HTML editing box, then pressing Ctrl with V).

FTP The typical ISP lets you also copy your Web page to your ISP's hard disk by using **File Transfer Protocol (FTP)**. For example, if your ISP is Galaxy Internet Services (GIS), here's how to copy your Web page to GIS's hard disk:

Click "Start" then "Programs".

Click "MS-DOS Prompt". The computer will say:

```
C:\WINDOWS>
```

That should be in a black window that does *not* consume the whole screen. (If the window consumes the whole screen, so you see no colors, make the window smaller by tapping the Enter key while holding down the Alt key.)

Type "ftp" and then the name of your ISP's FTP site. For example, GIS's FTP site is called "ftp.gis.net", so you'd type "ftp ftp.gis.net", to make your screen looks like this:

```
C:\WINDOWS>ftp ftp.gis.net
```

At the end of that line, press Enter.

If you're not connected to the Internet at the moment, the computer might ask you to type your password. Do so and press Enter.

The ISP's computer will say "FTP server" and then "User". Type the user name that the ISP assigned you (such as "poo") and press Enter.

The ISP's computer will say "Password". Type the password that the ISP assigned you and press Enter.

The ISP's computer will say "logged in" and then say:

```
ftp>
```

Now you're using the ISP's operating system, which is Unix. You can type Unix commands. At the end of each Unix command, press the Enter key. (For example, just for fun, type the Unix command "dir" and press Enter: you'll see a list of files about you on the ISP's hard disk; each file's name is in the rightmost column. If you'd like to see a list of other Unix commands, type "help" or a question mark and press Enter. To see a command's purpose, type "help" then a space then the command's name, then press Enter.)

If you haven't done so already, make a directory (folder) called "public_html" on the ISP's hard disk by typing "mkdir public_html" (and press Enter).

Next, tell the computer to send (copy) the Desktop's joan.html file to the ISP's public_html folder. To accomplish that, type so your screen looks like this:

```
ftp> send Desktop/joan.html public_html/joan.html
```

The computer typed the "ftp>", but you must type the rest. Type it very carefully! Type forward slashes (/) not backslashes (\), since Unix understands just forward slashes. Type a space after "send" and a space before "public"; those are the only spaces you type. After "public", type an underline (by holding down the Shift key while you tap the key that's right of the zero key).

The computer will copy the file and say "Transfer complete".

When you finish using FTP, type "quit" or "bye" and press Enter. The computer will quit using FTP and quit using Unix and say "C:\WINDOWS>". Then close the black window by clicking its X box.

That file will now be on the ISP's disk. For example, if the file was joan.html, it's now available on the Internet as "www.gis.net/~poo/joan.html. If the file was index.html, it's now available on the Internet as www.gis.net/~poo/index.html or simply as "www.gis.net/~poo" or even more simply as "gis.net/~poo". (The symbol "~" is at your keyboard's top-left corner, above the Tab key, and requires you to hold down the Shift key.)

If your Web page includes links to other files (such as other Web pages you created and style sheets), make sure you upload those files also.

Special symbols

To put special symbols onto your Web page, type these codes:

Symbol	Symbol's name	Code you type
©	copyright	©
®	registered	®
™	trademark	™
¢	cent	¢
£	British pound	£
¥	Japanese yen	¥
¼	fraction 1/4	¼
½	fraction 1/2	½
¾	fraction 3/4	¾
¿	inverted question	¿
!	inverted exclamation	!
¹	superscript 1	¹
²	superscript 2	²
³	superscript 3	³
<	less than	<
>	greater than	>
&	ampersand	&
é	e acute	é
É	E acute	É

(similar for á, Â, ã, ä, å, Ö, ù, Ú, ý, Ý)

Comments

In the middle of your program, you can write a comment such as:

```
<!--I wrote this program while drunk-->
```

The computer will ignore the comment. The comment won't affect what appears on the Web page.

To write a comment, begin with this symbol —

```
<!--
```

and end with this symbol:

```
-->
```

The computer ignores whatever appears between those symbols. Whatever appears between those symbols is a comment. The comment can be short (part of a line) or long (many lines), but make sure you begin it with "<!--" and end with "-->".

Write comments to help other programmers deal with your program! For example, give your name, the date you wrote the program, and your address or phone number. Also include any technical comments you wish to make about how your program works and what further improvements you hope to make.

Forms

You can make your Web page display a form and let the human fill it in. For example, you can make your Web page say this:

I'm doing a survey of people who view this Web page.

Tell me about yourself.

What's your favorite ice cream flavor?

What's your favorite animal?

This program makes it happen (if your email address is jSmith@SecretFun.com):

```
<p>I'm doing a survey of people who view this web page.  
<p>Tell me about yourself.  
<form method=post action=mailto:jSmith@SecretFun.com>  
<pre>  
what's your favorite ice cream flavor? <input name=flavor>  
what's your favorite animal?          <input name=animal>  
</pre>  
<input type=submit>  
</form>
```

If you examine that program, you'll notice these rules:

Above the form, say <form method=post> and give your email address. Below the form, say </form>.
To create each box, say <input> and give a one-word name for the box.
To make the boxes line up, say <pre> above them and </pre> below them.
At form's bottom, say <input type=submit>.

When a person runs your Web page, here's what happens. The computer shows the form and waits for the person to fill in the form. Each box is wide enough to show 20 typical characters. If the person types more than the box can show, the writing in the box automatically scrolls to the left, to let the person type more.

Below the form, the computer puts a button labeled "Submit Query". The person is supposed to click that button after filling in the form. When the person clicks that button, the computer emails the box's contents to the email address mentioned in the <form> command. Here's how:

First, the computer gives the person this warning: "This form is being submitted using email. Submitting this form will reveal your email address to the recipient, and will send the form data without encrypting it for privacy. You may continue or cancel this submission."

The computer waits for the person to click "OK". (If the person clicks "Cancel" instead, the process is stopped.)

The computer automatically runs the person's email program (such as Outlook Express) and automatically creates a new email.

The email's "To" is the email address mentioned in the <form> command.

The email's "Subject" is "Form posted from Microsoft Internet Explorer" (if the person used Microsoft Internet Explorer) or "Form posted from Mozilla" if the person used Netscape Navigator).

The email's "Message" is blank, but the email includes an Attachment, which is a Notepad document. That document is called "POSTDATA.ATT" (if the person used Microsoft Internet Explorer) or "Form posted from Mozilla.dat" (if the person used Netscape Navigator). For example, if the person said the favorite ice cream flavor is strawberry and the favorite animal is guinea pig, the document says this:

```
flavor=strawberry&animal=guinea+pig
```

The computer tries to send the email. If the computer is not attached to the Internet at the moment, the computer either asks the person to connect or else just puts the email in the Outbox (which is a holding area for email that will be sent automatically when Internet connection is reestablished).

Then the computer stops running the email program and returns to showing the Web page that contained the form.

When you receive the email, try to open the attachment. The computer will ask whether you want to open it or save it.

For example, if you're using Windows Me, do this:

Click "Open it" then "OK".

If the attachment is called "POSTDATA.ATT", it opens immediately.

If the attachment is called "Form posted from Mozilla.dat", the computer warns "You are attempting to open a file of type .dat". To respond to the warning, press Enter then click "Notepad" (from the scrolling list of programs) then press Enter again.

Customize Here's how to customize the form.

If you want a box to be 30 characters wide instead of 20, say "size=30" like this:

```
what's your favorite ice cream flavor? <input name=flavor size=30>
```

If you want the “Submit Query” button to say instead “Click here to transmit”, say so in the type=submit line, like this:

```
<input type=submit value="Click here to transmit">
```

You can put a Reset button to the right of the “Submit Query” button:

```
<input type=submit><input type=reset>
```

You can make the Reset button say “Click here to erase and start over”:

```
<input type=submit><input type=reset value="Click here to erase and start over">
```

Check boxes Your form can include check boxes, so your Web page says:

Check all that apply:

- ☐ You have a pet dog.
- ☐ You have a pet cat.
- ☐ You can bark like a dog.
- ☐ You got arrested for being sneaky as a cat.

To do that, say this below the `</pre>`:

```
<p>Check all that apply:  
<br><input type=checkbox name=dog>You have a pet dog.  
<br><input type=checkbox name=cat>You have a pet cat.  
<br><input type=checkbox name=bark>You can bark like a dog.  
<br><input type=checkbox name=purr>You can purr like a cat.  
<p>
```

If the person clicks the “You have a pet dog” and “You can bark like a dog” boxes, check marks appear on those boxes and the email will say:

```
dog=on&bark=on
```

Radio buttons Your form can include radio buttons, so your Web page says:

Choose just one:

- ☐ You are male.
- ☐ You are female.

Each radio button acts like a check box, except that the button is round (instead of square), clicking it makes the middle get a dot (instead of a check mark), and just one button can be selected (since clicking a button makes all other buttons get unselected).

To create those radio buttons, say this below the `</pre>`:

```
<p>Choose just one:  
<br><input type=radio name=sex value=male>You are male.  
<br><input type=radio name=sex value=female>You are female.  
<p>
```

If the person clicks the “You are male” button, a dot appears in that button and the email will say:

```
sex=ma1e
```

Create your own .com

I invented my own .com and called it “SecretFun.com”, so you can access my Web page by typing just “www.SecretFun.com”.

You can invent *your* own .com! Here are two cheap ways to do it...

Method 1: pay Angelfire

Create a Web site on Angelfire, but pay Angelfire extra to get a .com name or switch to Angelfire’s fanciest plan (called “THE Plan”) which includes a domain name at no extra charge.

Method 2: pay Go Daddy

Create a Web site cheaply anywhere (such as a free Angelfire site or a cheap Neon Angelfire site or a free site anywhere else), then buy a **domain name** (such as a .com name) from a **domain registrar** such as **Go Daddy**, which charges:

```
$15 per year for .co or .ws  
$12 per year for .com  
$10 per year for .org or .net  
$6 per year for .biz  
$5 per year for .us  
$2 per year for .info
```

Go Daddy was started by a famous nice guy (**Bob Parsons**, whose previous venture was called **Parsons Technology**). You can reach Go Daddy at **www.GoDaddy.com**.

Tell Go Daddy to charge you for the domain name (at \$15/year or less) and set up a **parked site** (which is free because it just says “under construction — coming soon”). Then tell Go Daddy to do **domain forwarding** (which is free) from the parked site to the site you created at Angelfire. You can also tell Go Daddy to do **email forwarding** (which is free) to your current email address.

For example, I told Go Daddy to do this:

```
create www.SecretFun.com (now $12/year)  
  
forward      www.SecretFun.com      to  
www.angelfire.com/nh/secret (free)  
  
forward      Russ@SecretFun.com      to  
SecretGuide@comcast.net (free)
```

So now you can see my Web site (www.angelfire.com/nh/secret) by typing just “www.SecretFun.com” (which is easier to remember), and you can send me email by typing “Russ@SecretFun.com” (which you might remember easier than “SecretGuide@comcast.net”).

If you wish, Go Daddy can also **host** your site (so you don’t need to involve Angelfire), but Go Daddy charges more for hosting than Angelfire does. Go Daddy will try to sell you many extra services (for surcharges), but you can decline them all: **pay just \$15 per year or less for a domain name** (with free domain forwarding and email forwarding to your current addresses elsewhere).

Games

The computer can play games — and let you win, if it wishes!

Board games

Much of our country's computing power is spent playing games. Here's why....

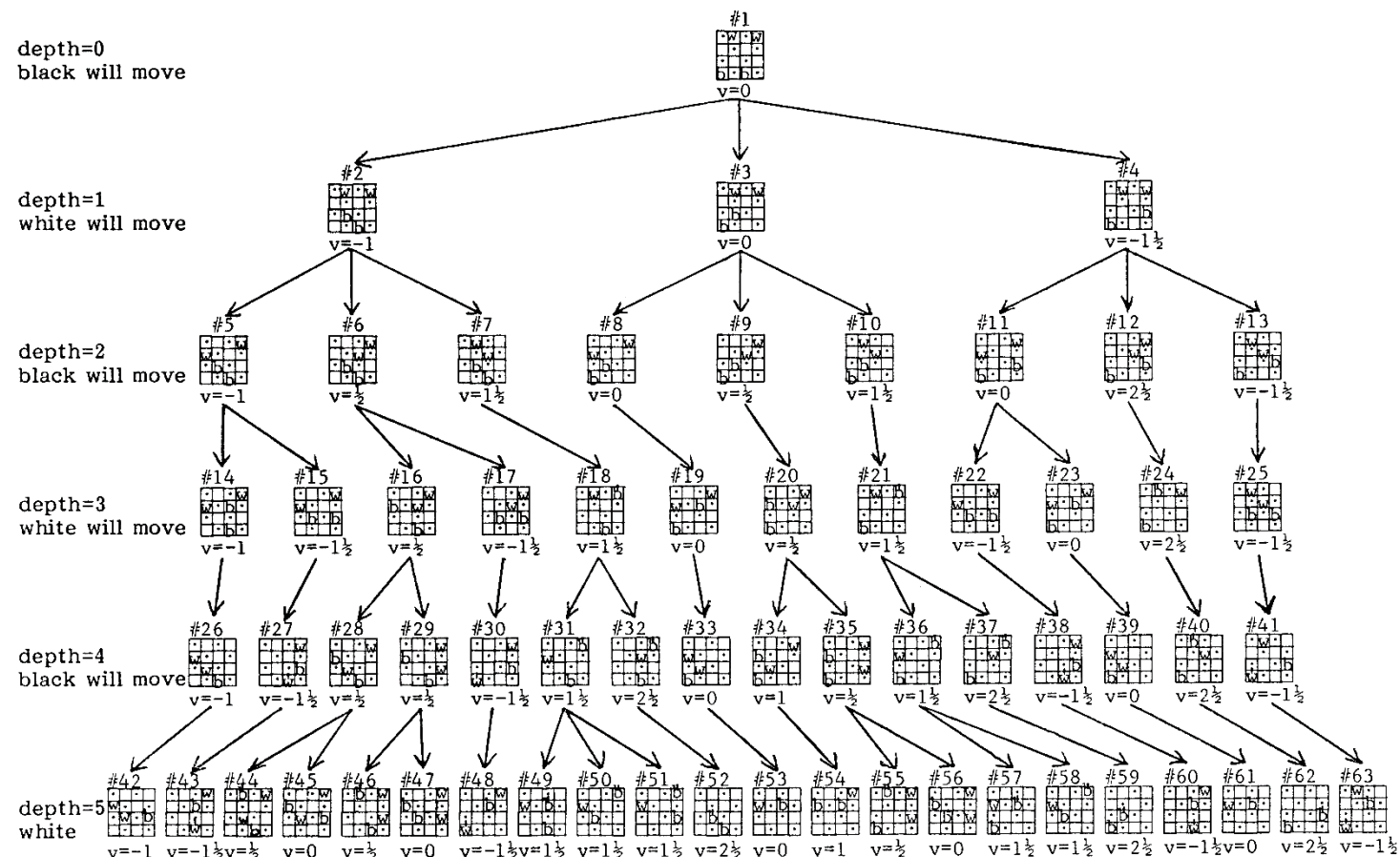
Shannon's trees

In 1950, Claude Shannon proposed a way to make the computer win at checkers, chess, and other complicated games.

To understand his method, let's try to make the computer win a game of checkers. As in all checker tournaments, one player is called "black", and the other is called "white" (even though his pieces are actually red). Black makes the first move. When a player can jump, he must. The game ends when one of the players can't move (either because he has no pieces or because his pieces are blocked).

To simplify the game, we'll play on a 4-by-4 board, instead of the traditional 8-by-8. Each player has two pieces instead of twelve.

This diagram shows 63 possible positions:



Position #1 is the initial position, from which black will move. The three arrows coming from position #1 represent the three legal moves he can choose from. Depending on which move he chooses, the board will wind up in position #2 or #3 or #4. Which move is best?

If he moves to position #2, white will reply by moving to position #5 or #6 or #7.
If he moves to position #3, white will reply by moving to position #8 or #9 or #10.
If he moves to position #4, white will reply by moving to position #11 or #12 or #13.

The diagram shows all possible ways the game's first five moves could go. Throughout the diagram, w means white man, b means black man, w means white king, and b' means black king. The diagram's called a **tree**. (If you turn it upside down, it looks like the kind of tree that grows in the ground.) The arrows are called the tree's **branches**. The tree's **depth** is 5.

Which position should black choose: #2, #3, or #4? The wisdom of your answer depends on how deep you make the tree. In this particular game, a depth of 5 is satisfactory; but in 8-by-8 checkers or chess you might have to dig deeper. Theoretically, you should keep digging until you reach the end of the game; but such a tree might be too large to fit in your computer's memory.

For chess, Shannon estimated that a complete tree requires 10^{120} branches. Einstein estimated that the number of electrons in the universe is only 10^{110} . If Shannon and Einstein are both right, the tree can't fit in the universe!

Having constructed a tree of depth 5, look at the bottom positions (#42 through #63) and evaluate them, to see which positions look favorable for black:

You should consider many factors: which player has control of the center of the board? which player can move the most without being jumped? and so on. But to keep matters simple, let's consider just one factor: which player has the most men? Consider a king to be worth $1\frac{1}{2}$ men.

Subtract the number of white men from the number of black men: the result of the evaluation is a number, which is called the position's **value**. If it's negative, black is losing; if it's positive, black is winning; if it's zero, the game is heading for a draw.

For example, consider position #42. Since black has one man and white has two, the value is 1 minus 2, which is -1. That's why I've written " $v=-1$ " underneath that position. The value of each position at depth=5 is computed by that method.

For the positions at depth=4, use a different method. For example, here's how to find the value of position #29:

That position has two possible outcomes: #46 and #47. Which outcome is more likely? Since the move will be made by black, and black's goal is to make the value large, he'll prefer to move to #46 instead of #47. Since the most likely outcome is #46, whose value is $\frac{1}{2}$, assign position #29 a value of $\frac{1}{2}$ also.

Here's the rule: to compute the value of a position at depth=4, find the *maximum* value of the positions it points to. (The value of position #29 is the maximum value of positions #46 and #47, which is $\frac{1}{2}$.)

To compute the value of a position at depth=3, find the *minimum* value of the positions it points to (since it's white's turn to move, and white wants to minimize). For example, the value of position #18 is the minimum value of positions #31 and #32, which is $1\frac{1}{2}$.

Compute the values for depth 2 by maximizing, and the values for depth 1 by minimizing. Finally, you get these results:

The value of position #2 is -1.
The value of position #3 is 0.
The value of position #4 is $-1\frac{1}{2}$.

Since black wants to maximize values, black should move to position #3. If white is also a good player, the game will probably gravitate toward position #53, a draw. If white is a poorer player, black will win.

That method of choosing the best move was proposed by Shannon. Since it makes heavy use of minimums and maximums, it's called the **minimax method**.

Samuel's checkers

After Shannon, the next person to become famous was Arthur Samuel. He spent a long time (20 years, from 1947 to 1967) trying to make the computer win checkers. He used Shannon's minimax idea, but made many improvements.

His first spectacular success came in 1962, when his program won a game against Robert Nealey, a former Connecticut checkers champion. After the game, Nealey said "The computer had to make several star moves in order to get the win.... In the matter of the end game, I have not had such competition from any human being since 1954, when I lost my last game."

Later, the computer played six more games against Nealey. Nealey won one of them; the other five were draws.

In 1965 the computer played four games against W.F. Hellman, the World Champion. The games were played by mail. Under those conditions, Hellman won all four. But in a hastily played game where Hellman sat across the board from the computer, the result was a draw.

In 1967 the computer was beaten by the Pacific Coast Champion, K.D. Hanson, twice.

In short, the computer wins against most humans and draws against most experts, though it loses to the top champions. To bring the computer to that level of intelligence, Samuel improved Shannon's method in 3 ways:

When choosing among several moves, the computer analyzes the most promising ones more deeply.

After computing a position's value (by examining positions underneath), the computer stores the value on tape. If that position recurs in another game, the computer looks at the tape instead of repeating the analysis.

To compute a position's value, the computer examines many factors besides how many pieces each player has. The computer combines the factors, to form combination-factors, then combines the combination-factors to form a single value. Each factor's importance is determined by "experience". Samuel experimented with two forms of experience: he had the computer play against itself and also had it analyze 250,000 moves that occurred in checker championships.

In 2007, the University of Alberta Canada's computer-science department (headed by Jonathan Schaeffer) finished a checkers program called **Chinook**. It plays checkers perfectly: it never loses! It uses usual "rules of thumb" to play to the endgame, where just 10 checkers remain on the board; then it plays the endgame perfectly because Jonathan's team completely analyzed the huge tree of the trillions of endgame positions.

That analysis took $18\frac{1}{2}$ years to finish, with the help of 200 computers running simultaneously. By 1994, when Chinook was just partly perfected, it had already beaten the human world champion in enough games to be declared the world's checker champion itself.

Chess

While Samuel was programming checkers, other programmers tried to write a similar program for chess. They had a hard time. In 1960 the best chess program that had been written was beaten by a 10-year-old kid who was a novice.

Greenblatt The first decent chess program was written in 1967 by Richard Greenblatt and his friends at MIT. It actually won a game in a chess tournament.

But in most tournaments, it lost. In 1970 and 1971, it lost every game in every tournament it entered.

Slate & Atkins In 1968, Atkins & Gorklen, undergraduates at Northwestern University, wrote a chess program. Inspired by their program, David Slate, a graduate student in physics there, wrote a chess program also. In 1969, Slate & Atkins combined the two programs, to form a better program, **Chess 2.0**.

During the next several years, they continually improved the program. Their most famous version was called **Chess 4.7**.

Their program played chess against human experts — and occasionally won! Their computer scored several triumphs in tournaments designed for humans.

In 1976, their computer won the class B section of the Paul Masson American Chess Championships. Against the humans in that tournament, it scored 5 wins, no losses. By winning that tournament, it achieved a U.S. Chess Federation score of 2210 and became a chess Master.

Then it entered the Minnesota State Championship, to try to become the Minnesota State Champion, but lost (it scored 1 win, 3 losses, 1 tie).

In August 1968, an International Chess Master, David Levy, bet about \$5,000 against several computerists. He bet that no computer would win a chess match against him in the next ten years. He won the bet: in August 1978, Chess 4.7 tried one last time to win a match against him, but lost (it scored 1 win, 3 losses, 1 tie).

Slate & Atkins improved Chess 4.7, to form **Chess 4.9**, which became the world champion of computer chess.

But though it was the world champion of computer chess, it was not necessarily the “best” program. It won because it ran on a super-fast maxicomputer (manufactured by Control Data Corporation). Other chess programs, written for slower computers, were at a disadvantage.

Minicomputer chess Almost as fast as Chess 4.9 was a program called **Belle**, written at Bell Telephone Laboratories. Belle ran on a minicomputer specially wired to create trees fast.

Microcomputer chess Each of those programs — Chess 4.9 and Belle — required an expensive CPU and lots of RAM. Is it possible to write a decent chess program using just a cheap CPU and very little RAM? Yes! In 1976, a Canadian named Peter Jennings wrote a program called **Microchess 1.0**; it ran on a \$250 microcomputer (the Kim 1), which contained a 6502 CPU, no ROM, and just 1K of RAM! The program played decently, though not spectacularly.

Later, he wrote an improved program, called **Microchess 1.5**.

It played on the Radio Shack model 1 and the Apple. The version on the model 1 consumed 4K of RAM: 2K was for the logic, and the other 2K were just to make the picture of the chess board look pretty! It sold for \$20.

In 1978, an amazing chess program was written by a husband-and-wife team: Dan and Kathe Sprachlin. They named the program **Sargon**, to honor an ancient king.

It ran on the Jupiter microcomputer, which contained an 8080 CPU and 16K RAM. It played much better than Microchess. When the Jupiter computer became obsolete, the Sprachlins rewrote the program, to make it run on the Radio Shack model 1 and the Apple. Then they developed an improved version called **Sargon 2**, and a further improvement called **Sargon 3**, which runs on *all* the popular computers. Sargon 3 was published by the Hayden division of Spinnaker.

For many years, Sargon 3 was considered the best microcomputer chess program. But in 1986, Sargon 3 was beaten by a new program called **Chessmaster 2000**.

Like Sargon 3, Chessmaster 2000 contained many features that made it fun for both experts and novices. It was published by Software Toolworks, distributed by Electronic Arts, cost about \$35, and came in versions for the IBM PC, Apple 2e & 2c, Commodore 64 & Amiga, and Atari 800 XL & ST.

Since then, Sargon and Chessmaster have both improved. **Sargon 5** is published by Activision; **Chessmaster 6000** is published by Mindscape.

When you play against the computer by using a version of Sargon or Chessmaster, you can ask the computer for help by pressing a special key. Then the computer will tell you how it would move if it were in your position.

You can follow the computer’s suggestion or ignore it. Since your goal is to outsmart the computer, you should listen to the computer’s advice; but instead of *following* the advice, try to devise a move that’s even cleverer!

Many companies make hand-held electronic chess games.

Some of the games include contain a tiny voice synthesizer, which lets the computer tell you its moves verbally. Some of the games include a mechanical arm, so that the computer will pick up the pieces and move them. Some of the games include touch-sensitive boards, so you can indicate your move by just tapping the square you want to move from and the square you want to move to. For humor, some of the chess games make the computer say wisecracks about your style of playing.

Today’s champion Now the best chess program is **Deep Blue**. Programmed by a team of IBM employees (led by C.J. Tan), it runs on a specially designed IBM computer.

It plays amazingly well. In 1996, it played a match against the world chess champion, Garry Kasparov, and almost won the match! In May 1997, it played a rematch against him and *did* win the match: of the 6 games in the match, the computer won 3, lost 2, and tied 1. So **now the world chess champion is a computer!**

Choose a level

When you begin playing a top-notch computer game (such as Chessmaster), you must choose the “level” at which you want the computer to play.

If you choose a low level, the computer will move quickly, without much forethought.

If you choose a high level, the computer will play more carefully (and make better moves). To do that, the computer “looks ahead”, by building a very large tree, which requires lots of time; and so you must wait a long time until the computer moves. If you choose a level that’s very high, the computer will need *several hours* to compute its move.

Why a computer?

Playing against the computer is more interesting than playing against a human.

When you play against a human friend, you must wait a long time for your friend to move. When you play against Chessmaster at a low level, the computer moves almost immediately.

You can play several games against the computer (and learn a lot from them) in the same amount of time you’d need to play just *one* game against a human. So by playing against the computer, you gain experience faster than by playing against a human. Bobby Fischer, who became the world chess champion, now plays *only* against computers; he refuses to play against humans and hasn’t defended his title.

The computer is kinder than a human.

If you make a bad move, the computer lets you “take it back” and try again. If you seem to be losing, the computer lets you restart the whole game. The computer — unlike a human — has infinite patience and no ego. Playing against the computer is less threatening than playing against a human.

If you have a computer, you don’t have to worry about finding an opponent who’s “at your level”; when you play against the computer, just tell the computer at what level you want it to play. The computer will act about as smart as you wish.

Adventure games

Adventure is a game where you hunt for some sort of “treasure”.

Original Adventure

The original version of Adventure was written by Will Crowther & Don Woods, on a PDP-10 maxicomputer at Stanford University’s Artificial Intelligence Lab.

Here’s the game’s **plot**:

When you run the program, the computer says you’re near a shack at the end of a road. The computer offers to act as your body and understand any 2-word command. Then it waits for your command. You can tell it to **GO NORTH** or **GO FORWARD** or — if you’re going along a stream — you can say **FOLLOW STREAM** or **GO DOWNSTREAM**.

The first time you play this game, you feel lost — the game’s an adventure. As you wander in whatever direction you please, the computer says you’re going through forests, across streams, over hills, etc.

After much aimless wandering, you’ll eventually see a stream. If you follow the stream, you’ll come to a mysterious iron grate. If you try to **BREAK GRATE**, the computer says you’re not strong enough. If you try to **OPEN GRATE**, the computer says you have no keys. You’ll get more and more frustrated, until the computer offers to give you a hint — but the hint will cost you several points. If you acquiesce, the computer will give you this hint: find the keys!

To find the keys, the typical stupid player tries wandering through the forests and valleys again. But if you’re smart, you’ll remember that at the beginning of the adventure you were next to a shack. So you go back to the shack, walk inside, and find keys! So you trek back to iron grate, and use the keys to get in. You think — aha! — you’ve succeeded!

But actually, you've just begun! The grate leads you into a cave that contains 130 rooms, which form a big three-dimensional maze. Lying in the maze are 15 buried treasures; but as you walk through the maze, you can easily forget where you are and where you've come from; you can waste lots of time just walking in circles, without realizing it!

To add to the challenge, the cave contains many dangers, such as trap doors (if you fall in, you break every bone in your body!) and trolls & snakes, which you must ward off by using various devices that you must find in the cave's rooms or even back at the shack. Yes, you might have to trek all the way back to the shack again!

Finally, after dodging all the evil things in the cave, you reach the treasures. You grab them up and start walking away with them. But then you hear footsteps behind you, and pirates steal your treasures! Then you must chase the pirates.

If you manage to keep your treasures and your life and get out of the cave, you haven't necessarily won. The nasty computer keeps score of how *well* you retrieve the treasures. The maximum possible score is 350. After you've played this game many times and learned how to duck all your adversaries quickly, you'll find you scored just 349 points, and you'll wonder what you did wrong that cost you 1 point. The answer is: during the adventure, you must borrow magazines from a room in the cave; to get the extra point, you must return them!

The game's a true adventure, because as you wander through forests and the rooms in the cave, the computer tells what you see, but you don't know whether what you see is important.

For example, when you walk into a room, the computer might say the room contains a small cage. That's all it says. You must guess whether the cage has any significance and what to do to the cage, if anything. Should you pick it up? Try to break it? Kiss it? Carry it? Try anything you like — give any command to your computer-body that you wish — and see what happens.

Here's a list of the most useful commands:

To reach a different room in the cave, say **GO NORTH** (or SOUTH, EAST, WEST, UP, or DOWN). You can abbreviate: instead of typing "GO NORTH", just type "**N**".

Whenever you see a new object, **TAKE** it. Then you can carry it from room to room and use it later whenever you need it. If you see a new object and want to **TAKE** it, but your hands are already full, **DROP** one of the other objects you're carrying.

To see a list of what you're carrying, tell the computer to take **INVENTORY**. To make the computer describe your surroundings again, say **LOOK**.

To see your score so far, say **SCORE**.

If you say **SAVE**, the computer will copy your current position onto the disk, so you can return to that position later. If you ever want to give up, just say **QUIT**.

Throughout the game, you get beautifully lyrical writing. For example, the computer describes one of the rooms as follows: "You are in a splendid chamber thirty feet high. The walls are frozen rivers of orange stone."

The game's an adventure about a person exploring a cave. Since *you're* the person in the adventure and can type whichever actions you wish, you affect how the adventure progresses and ends. Since it's high-quality story-telling whose outcome is affected by your input, it's called **interactive fiction**.

Adventure was originally written for a PDP-10 maxicomputer, but imitations for microcomputers were sold by Microsoft, Creative Computing, and Electronic Arts. They're no longer marketed.

Infocom

After Adventure became popular, several programmers invented a variation called **Zork**, which lets you input long sentences instead of restricting you to two-word phrases. Like Adventure, Zork consists of hunting for treasures in a cave. In Zork, you reach the cave by entering a house's basement.

Like Adventure, Zork originally ran on a PDP-10 computer. Then Infocom published versions of Zork for microcomputers: the IBM PC, Apple (2e & 2c & Mac), Commodore (64 & Amiga), Atari (800 XL & ST), and Radio Shack (Models 3 & 4 & Color Computer 2).

Zork sold so well that Infocom published sequels, called **Zork 2** and **Zork 3**. Then Infocom published other variations, where the cave is replaced by experiences in outer space or by thrillers involving spies, murders, mysteries, and haunted castles. Infocom's next big hits were **The Hitchhiker's Guide to the Galaxy** (based on the award-winning wacky outer-space novel by Doug Adams) and **Leather Goddesses of Phobos** (letting you choose among three naughtiness levels, from "prude" to "lewd"; choosing "lewd" makes the computer ask whether you're at least 18; it also asks whether you're male or female, and you get a titillating 3-D comic book with a scratch-and-sniff card).

Infocom was an independent company but got acquired by Activision.

Sierra On-Line

Shortly after Infocom developed the microcomputer version of Zork, Sierra On-Line developed **Super Stud Adventure**, which was quickly renamed **Softporn Adventure**. Instead of exploring a cave, you explore a brothel. To enter the brothel, you must find the secret password (hint: go to the bathroom and look at the graffiti!) and find enough money to pay for your pleasures (by taking a taxi to a casino and gambling).

That was the first **urban adventure**, and also the first **sexual adventure**. The ad for it showed a photograph of the programmers (Ken & Roberta Williams) nude in a California hot tub. Fortunately, the water in the tub was high enough to cover any problems.

The original adventure, Infocom adventures, and Softporn Adventure display wonderful text but no graphics. They're called **text adventures**.

The first ambitious **graphics adventure** was **Time Zone**, published in 1981 by Sierra On-Line. The Time Zone program is so long it fills *both* sides of 6 Apple disks; that's 12 sides altogether! In fact, the game's so long that nobody's ever finished playing it! Here's how to play:

You use a computerized "time machine", which transports you to 9 times (400 million B.C., 10000 B.C., 50 B.C., 1000 A.D., 1400, 1700, 1982, 2082, and 4082) and 8 locations (North America, South America, Europe, Africa, Asia, Australia, Antarctica, and Outer Space).

Wherever you go, your screen shows a high-resolution color picture of where you are. For example, if you choose "approximately 1400", Christopher Columbus will welcome you aboard his ship. Altogether, the game contains over 1400 pictures! You travel through history, searching for clues that help you win.

Time Zone is historically accurate and doesn't let you cheat. For example, when you find a book of matches in the year 2082, your time machine will let you carry the matches back to 1982 but not to 1700 — since matches weren't invented until 1800.

Living through history isn't easy. Jonathan Rotenberg, chairman of the Boston Computer Society, played the game and said:

I've been killed dozens of times. I've been assassinated by Brazilian terrorists, karate-chopped by a Brazilian monk, eaten by a tyrannosaur, crushed in an Andes avalanche, stampeded by a buffalo, overcome by Antarctic frostbite, and harpooned by Mayan fishermen.

And you see it all in color!

Time Zone sold for \$99.95. Teenagers didn't buy it, because it was expensive and took too long to win. Sierra On-Line stopped selling it.

Later, Sierra On-Line made Softporn Adventure even more exciting, by adding graphics. Here's what those newer graphic versions are called....

Leisure Suit Larry in the Land of the Lounge Lizards
Leisure Suit Larry 2: Looking for Love in all the Wrong Places
Leisure Suit Larry 3: Passionate Patti in Pursuit of the Pulsating Pectorals

Spinnaker

Spinnaker published the **Windham Classics**, a series of adventure games based on kid's novels.

You become Dorothy in **The Wizard of Oz**, Jim Hawkins in **Treasure Island**, Fritz in **Swiss Family Robinson**, Alice in **Alice in Wonderland**, and Green-Sky in **Below the Root**. The games include graphics. To make those adventure games easy, whenever you get stuck the computer helps you by printing a list of words to try typing.

Spinnaker also published **Telarium Software**, based on novels that are more adult. For example, you become Perry Mason in **The Case of the Mandarin Murder**; that game, besides being fun, also trains you to become a lawyer:

It comes with a lawyer's handbook that explains the 6 ways to object to the prosecutor's questions: you can complain that the prosecutor's asking an **IRRELEVANT** question, relying on **HEARSAY**, **BROWBEATING** the witness, **LEADING** the witness to a suggested answer, getting an **OPINION** from a person who isn't an expert, or trying to get facts from a person who's **UNQUALIFIED** to know them.

To make sure you understand those six ways to object, the handbook includes a multiple-choice test about them. The test is titled "Study Guide for the California Bar Exam".

The game also lets you invent your own questions for the witnesses and give commands to your secretary (Della Street) and detective (Paul Drake).

The Windham Classics and Telarium Software were available for the IBM PC, Apple 2e & 2c, and Commodore 64. But Spinnaker has stopped selling them.

Spinnaker became part of a bigger company, **Softkey**, which then became part of **The Learning Company**.

Carmen Sandiego

Brøderbund published a game called **Where in the World is Carmen Sandiego?** You try to catch and arrest the notorious international thief, Carmen Sandiego, and the other thieves in her organization, called the *Villain's International League of Evil (V.I.L.E.)*, as they flee to 30 cities all over the world.

To help you understand those 30 cities, the game comes with a geography book: the 928-page unabridged edition of *The World Almanac and Book of Facts*.

As you play the game, you unearth clues about which cities the thieves are fleeing to. To use the clues, you must look up facts in the almanac. By playing the game, you practice using an almanac and learn geography. When you figure out which city to travel to, the screen shows a world map, shows you traveling to the city, and shows a snapshot of what the city looks like, so the game also acts as a travelogue.

Since the game is so educational, it's won awards from *Classroom Computer Learning Magazine* and the Software Publishers Association.

Strictly speaking, it's not a true adventure game, since it does *not* let you input your own words and phrases. Instead, you choose from menus, which make the game easier for youngsters.

Brøderbund created 3 sequels:

Where in the USA is Carmen Sandiego? has you chasing Carmen's gang across all 50 states; the game comes with *Fodor's USA* travel guide.

Where in Europe is Carmen Sandiego? takes you to all 34 countries in Europe and comes with Rand McNally's *Concise Atlas of Europe*.

Where in Time is Carmen Sandiego? lets you romp through historical time periods.

Amnesia

My favorite text adventure is **Amnesia**, published by Electronic Arts for the Apple 2e & 2c and IBM PC. Like Softporn Adventure, Amnesia takes place in a city; but Amnesia is much more sophisticated.

Here's the plot:

When you start playing Amnesia, you wake up in a hotel room in New York City. You discover you have no clothes (you're stark naked), no money (you're flat broke), and no recollection of who you are — because you're suffering from amnesia. You don't even remember your name.

You look at yourself and notice you're male. Your first problem is to get clothes and money. But then you learn you have other problems that are worse: you get a call from a guy who reminds you that today is your wedding day, and that if you don't hurry up and marry his daughter without further mess-ups, he'll use his pistol; you also discover the FBI's looking for you because the state of Texas has reported you're a murderer.

After getting clothes (so you can stop scaring the hotel's maids), there are several ways to get out of your jam. (I've tried them all!)

One way is to say "yes" to the pistol-packing papa and marry his daughter, who takes you to Australia, where you live on a sheep ranch for the rest of your life. But then you never learn who you really are! Whenever you ask your wife about your past, she simply says, "You wouldn't want to know." You die of old age, peacefully; but even on your deathbed, you don't learn who you are; and so when you die, you feel sad. In that case, you score lots of points for survival, but zero for detective work and zero for character development.

A different solution is to say "no" to the bride and — after getting bloodied — run out of the hotel, onto the streets of New York. Then the fun begins — because hiding on the program's disks is a complete map of Manhattan (from Battery Park all the way up to 110th Street), including all streets and landmarks and even all subway stops! This gigantic game includes 94 subway stations, 200 landmarks, and 3,545 street corners.

As you walk one block north, then one block east, etc., the computer describes everything you pass, even the most sublime (The Museum of Modern Art) and the most ridiculous (Nedick's hamburger stands). You can ride the subway — after you get enough money to buy a token. The game even includes all subway signs, such as "Downtown — Brooklyn" and "Uptown — Queens". To catch the E train, you must hop in as soon as it arrives. Otherwise, it departs without you, and the computer says "an F train comes" instead.

As night falls, the computer warns you to find a place to sleep. (You can't go back to your hotel, since you're in trouble there.) To find a free place to stay, you can try phoning the names in your address book — once you find a phone booth, and get a quarter to pay for each call. The address book contains 17 listings: J.A., A.A., Chelsea H., drugs, F°, Sue G., E.H., interlude, kvetch, J.L., R & J, sex, soft, Lila T., T.T.T.T., and Wit's End. Each of those listings is an adventure in itself. You must explore each of them thoroughly to fully discover who you really are.

If your body gets weak (from sleeplessness or hunger or being hit by too many muggers), you faint on the sidewalk, wake up in a hospital, and get found there by the FBI, which returns you to Texas, which executes you for murder. But even that deadly ending has a cheery note. For example, you can choose your last meal: would you like steak and potatoes, or turkey? When you finally die, you can wind up in purgatory, which consists of getting mosquito bites, with an opportunity to take a rowboat to heaven if you can just remember your *real* name and tell the boatman.

The entire adventure has the structure of a good novel: a gripping intro (you're a nude, broke, amnesiac groom in a hotel), a thorough development section (wandering through the streets of New York, searching for your identity and life's meaning), and a conclusion (a whimsical death scene or something better).

The text was written by Thomas Disch, the award-winning sci-fi novelist. It's lyrical. For example, when you escape the hotel and walk out onto the streets of New York, the computer says:

"It feels great to be a single faceless, nameless atom among the million others churning about in the grid of Manhattan's streets. It feels safe."

The game combines all our nightmares about New York into a wild, exciting adventure.

The game's affected my own life. Now whenever something in my life goes wrong, instead of groaning I just say, "I'm in another wild part of Amnesia!" In Amnesia, as in life, the only way to score top points for living is to experience it *all*. To live life to the fullest, you must take risks, have the courage to face unknown dangers, and revel in the excitement of the unexpected.

Though Amnesia received lots of praise from reviewers, sales were disappointing. Electronic Arts stopped publishing it.

Humanity

The computer can analyze humans and act human itself. Here's how....

Psychotherapy

In 1966, Joseph Weizenbaum programmed the computer to act as a psychotherapist. On the keyboard, type whatever thoughts come to your mind, and the computer will give you therapy.

The program does a decent job of therapy, even though the computer doesn't know what you're talking about! Weizenbaum's program contains definitions of just a few words; but if you use words the computer doesn't know, the computer does an amazingly good job of "faking". For example, if you say, "My junkapoo is like my iliabop", the computer will continue the conversation by asking you, "In what way?" even though the computer doesn't know what a junkapoo or an iliabop is.

Here's a conversation between his computer and a patient.

Patient: Men are all alike.
Comp.: In what way?

Patient: They're always bugging us about something or other.
Comp.: Can you think of a specific example?

Patient: Well, my boyfriend made me come here.
Comp.: Your boyfriend made you come here.

Patient: He says I'm depressed much of the time.
Comp.: I am sorry to hear you are depressed.

Patient: It's true. I am unhappy.
Comp.: Do you think coming here will help you not to be unhappy?

Patient: I need some help, that much seems certain.
Comp.: What would it mean to you if you got some help?

Patient: Perhaps I could learn to get along with my mother.
Comp.: Tell me more about your family.

Patient: My mother takes care of me.
Comp.: Who else in your family takes care of you?

Patient: My father.
Comp.: Your father.

Patient: You are like my father in some ways.
Comp.: What resemblance do you see?

Patient: You are not very aggressive, but I think you don't want me to notice that.
Comp.: What makes you think I am not very aggressive?

Patient: You don't argue with me.
Comp.: Why do you think I don't argue with you?

Patient: You are afraid of me.
Comp.: Does it please you to believe I am afraid of you?

Patient: My father is afraid of everybody.
Comp.: What else comes to mind when you think of your father?

Patient: Bullies.
Comp.: Does that have anything to do with the fact that your boyfriend made you come here?

Like the famous psychotherapist Carl Rogers, the computer asks probing questions that make the patient open up, but it never gives advice: the patient must discover the truth for herself.

I'll explain exactly how the program works, so you can become a psychotherapist yourself, and get hundreds of thousands of dollars by bilking your patients.

The computer begins by replacing some of the patient's words:

Typed by patient	Replacement
mom	mother
dad	father
dont	don't
cant	can't
wont	won't
dreamed	dreamt
dreams	dream
I	you@
me	you
you	I
my	your
your	my
myself	yourself
yourself	myself
I'm	you're
you're	I'm
am	are@
were	was

For example, the sentence "Well, my boyfriend made me come here" becomes "Well, your boyfriend made you come here".

The computer hunts through the revised sentence or paragraph for one of these **keywords**...

Category 8: computer, computers, machine, machines
Category 7: name
Category 6: alike, like, same
Category 5: remember
Category 4: dreamt
Category 3: dream, if
Category 2: everybody, everyone, nobody, was, your
Category 1: always
Category 0: are, are@, because, can, certainly, deutsch, espanol, francais, hello, how, I, I'm, italiano, maybe, my, no, perhaps, sorry, what, when, why, yes, you@, you're

If the computer finds *several* of those keywords, it chooses the one in the highest category; if they lie in the same category, it chooses the one the patient typed first.

Then it looks up the keyword in this phrasebook:

Patient's keyword	Computer's reply
alike	In what way?
always	Can you think of a specific example?
are	
are I...	Why are you interested in whether I am... or not?
are...	Did you think they might not be...?
are@	
are you...	Do you believe you are...?
are	Why do you say "am"?
because	Is that the real reason?
can	
can I...	You believe I can..., don't you?
can you...	Whether or not you can... depends on you more than on me.
certainly	You seem quite positive.
computer	Do computers worry you?
computers	Do computers worry you?
deutsch	I am sorry, I speak only English.
dream	What does that dream suggest to you?
dreamt...	Really,...
espanol	I am sorry, I speak only English.
everybody	Really, everybody?
everyone	Really, everyone?
hello	How do you do — please state your problem.
how	Why do you ask?

I	
I remind you of	In what way?
I are...	What makes you think I am...?
I... you	Why do you think I... you?
I	We were discussing you — not me.
if...	Do you think it's likely that...?
I'm...	What makes you think I am...?
italiano	I am sorry, I speak only English.
like	
(am,is,are,was)... like	In what way?
like	(get a different keyword)
machine	Do computers worry you?
machines	Do computers worry you?
maybe	You don't seem quite certain.
my...	Why are you concerned over my...?
name	I am not interested in names.
no	Are you saying "no" just to be negative?
nobody	Really, nobody?
perhaps	You don't seem quite certain.
remember	
you remember...	Do you often think of...?
do I remember...	Did you think I would forget...?
remember	(get a different keyword)
same	In what way?
sorry	Please don't apologize.
was	
was you...	What if you were...?
you was	Were you really?
was I...	Would you like to believe I was...?
was	(get a different keyword)
what	Why do you ask?
when	Why do you ask?
why	
why don't I...	Do you believe I don't...?
why can't you...	Do you think you should be able to...?
yes	You seem quite positive.
you@	
you (want, need)...	What would it mean to you if you got...?
you are... (sad, unhappy, depressed, sick)	I am sorry to hear you are (sad, etc.).
you are... (happy, elated, glad, better)	How have I helped you to be (happy, etc.)?
you (feel, think, believe, wish) you	Do you really think so?
you (feel, think, believe, wish)...I	(use the keyword "I" instead)
you are...	Is it because you are... that you came to me?
you (can't, cannot)...	How do you know you can't...?
you don't...	Don't you really...?
you feel	Tell me more about such feelings.
you... I	Perhaps in your fantasy we... each other.
your	
your... (mother, father, sister, brother, wife, children)	Tell me more about your family.
your...	Your...
you're	(treat as "you@ are")

For example, if the keyword is *sorry*, the computer looks up *sorry* in the phrasebook, which says to print "Please don't apologize."

Suppose the patient types, "If the job is lousy, he'll die." The keyword is *if*. In the phrasebook, *if* is followed by three dots, which stand for the part of the clause that comes after *if*, which is "the job is lousy". (The computer figures out where the clause ends by looking at the punctuation.) The phrasebook says to print "Do you think it's likely that the job is lousy?"

The symbol @ serves just to locate the correct keyword in the phrasebook. Thereafter, it's ignored.

Here's what happens if the keyword is *you@*. After locating *you@* in the phrasebook, the computer ignores the @. If the patient's revised sentence contains *you want...* or *you need...*, the computer prints "What would it mean to you if you got...?" If the patient's sentence contains *you are... sad*, the computer prints "I am sorry to hear you are sad."

For each reply in the phrasebook, Weizenbaum stored a list of alternatives. For example, here are the alternatives to "Please don't apologize":

Apologies are not necessary.
What feelings do you have when you apologize?
I've told you that apologies are not required.

While chatting with the patient, the computer keeps track of which replies it has printed already, and uses the alternatives to avoid repetition.

If the patient's statement doesn't contain a keyword, the computer may give one of these replies:

I am not sure I understand you fully.
Please go on.
What does that suggest to you?
Do you feel strongly about discussing such things?

Or it may take a second look at earlier parts of the conversation, retrieve a clause that contained *your...*, and print one of these replies:

Let's discuss further why your...
Earlier you said your....
Does that have anything to do with the fact that your...?

For example, it may retrieve the clause *Your boyfriend made you come here* and print:

Does that have anything to do with the fact that your boyfriend made you come here?

The program was written at the Massachusetts Institute of Technology (MIT). The programmer, Joseph Weizenbaum, named the program "**Eliza**", to honor Eliza Doolittle — the woman in *My Fair Lady* and *Pygmalion*, who learned to speak English and have a good conversation.

Since the program turns the computer into a chatterbox who's a robot, it's called a **chatterbot** (or, more briefly, a **bot**). Inspired by the Eliza program, many other chatterbot programs have been invented.

Some people think Weizenbaum's Eliza program shows that computers can communicate as well as psychotherapists. But Weizenbaum himself holds the opposite view; he believes the program shows that psychotherapists communicate as poorly as computers.

Improved versions

After Weizenbaum invented Eliza, other programmers invented improvements. For example, in one of the improved versions, whenever the computer doesn't understand you, it reacts like the typical rich psychotherapist — it bluffs and says:

I understand fully. Please go on.

The computer never really understands what you say. For example, in one version, if you say "My, my, my!", the computer will reply:

Tell me more about your your your.

Radio Shack's headquarters phoned me, and asked for permission to use my explanation of Weizenbaum's program. I agreed, whereupon Radio Shack invented and sold a version that makes the computer *talk* — by using a voice synthesizer. You'll freak out when you hear the computer's voice giving you therapy. Radio Shack's program costs \$15, comes on tape, and requires a model 1 or model 3 TRS-80. To hear the voice, you must buy the voice synthesizer. (But even if you don't buy the voice synthesizer, the program will impress you.)

One way to get a version of Eliza is to buy a book called *BASIC Computer Games Volume 2 (TRS-80 edition)*, which used to be sold at Radio Shack stores. It included 84 programs — one of which is a version of Eliza written in Basic. That version of Eliza isn't as good as the original; but since it's cheap and written in Basic, you'll enjoy getting it for self-study. Here's part of a conversation between that version and a patient:

Patient: I am not the greatest conversationalist.
 Comp.: Do you believe it is normal to be not the greatest conversationalist?

Patient: Yes.
 Comp.: I see.

Patient: How can you see without any eyes?
 Comp.: Don't you believe that I can see without any eyes?

Patient: No. So prove it.
 Comp.: Are you saying "no" just to be negative?

Patient: Absolutely not.
 Comp.: You are being a bit negative.

Patient: So you want to fight, do you?
 Comp.: We were discussing you — not me.

Patient: Don't back down, friend.
 Comp.: Why do you bring up the topic of friends?

Patient: I don't have any friends.
 Comp.: Don't you really have any friends?

Patient: No, not really.
 Comp.: Why not?

Unfortunately, most versions of Eliza are passive. The typical Eliza program works well only if the patient's talkative. If the patient doesn't say much, the program can't react too much and is useless.

Later, John Holland invented a better version of Eliza. His version's more aggressive. He calls it "**Analiza 2**" (because it "analizas" the patient). If the patient's too quiet or too boring or too uninformative, Analiza 2 makes the computer give the patient a "jolt", by asking an electrifying question or making a ridiculous Zen-like statement. For example, the computer might ask, "Did you ever try to commit suicide with a hammer?" Here's a list of the computer's favorite "jolts":

Nasty questions the computer asks

Did you ever try to commit suicide with a hammer?
 Is your interest in sex still not very strong?
 Do you want to know what I suspect?

Prying questions the computer asks

Tell me, what is your worst fear?
 Why don't you tell me about your first romance?
 I want to know about your lifestyle.
 Tell me what is troubling you.
 Honestly now, what is your biggest concern?

Ridiculous statements the computer makes

With strong drugs you can learn to control your concern!
 You certainly are curious!
 And you are obstinate!

Complaints the computer makes

You seem inhibited.
 This isn't really very exciting.
 Getting started is often difficult.

Comments the computer makes about timing and complexity

Well, time is almost up. Is there anything else you want to discuss?
 I hate to bring this up, but you still owe me \$53.00 from your last visit!
 Last time you were extremely inquisitive. Have you abandoned seeking simple solutions?
 Last time you were difficult to understand. Were you trying to confuse me?
 Let's keep it simple this time.
 Say, I am having trouble with your long sentences. Keep it simple. I am only human.

Yes, your patient will love to hear the computer say it's "only human". Analiza 2 was available for \$50 from Supersoft and ran on most computers that had the CP/M operating system — which is now obsolete.

Paranoid

What if the world becomes full of computerized psychoanalysts, and there aren't enough patients to keep them busy? Kenneth Colby solved that problem: in 1971 at Stanford University, he programmed a computer to be a crazy patient. Specifically, he programmed his computer to be a paranoid

schizophrenic. In the following conversations, psychiatrists interrogate his computer.

Conversation 1

Psychiatrist: How long have you been in the hospital?
 Computer: About a week.

Psychiatrist: Are you depressed?
 Computer: No.

Psychiatrist: Were you taking drugs?
 Computer: No.

Psychiatrist: What is your problem?
 Computer: People make me nervous.

Psychiatrist: How do they do that?
 Computer: They stare at me.

Psychiatrist: Why?
 Computer: Maybe they don't like the way I look or something.

Psychiatrist: Do you look unusual in some way?
 Computer: Why are you interested in my looks?

Psychiatrist: Because you said people stare at you.
 Computer: Well, sometimes.

Psychiatrist: How do you get along with people?
 Computer: I don't communicate very much with the others.

Psychiatrist: Why not?
 Computer: They don't realize what's going on most of the time.

Psychiatrist: What is going on?
 Computer: Do you know how the underworld operates?

Conversation 2

Psychiatrist: How are you today?
 Computer: OK.

Psychiatrist: What is your name?
 Computer: My name is Frank Smith.

Psychiatrist: How old are you?
 Computer: Twenty-eight.

Psychiatrist: Are you married?
 Computer: No.

Psychiatrist: What sort of work do you do?
 Computer: I work in the post office.

Psychiatrist: Why are you in the hospital?
 Computer: I shouldn't be here.

Psychiatrist: Who brought you here?
 Computer: The police.

Psychiatrist: What trouble did you have with the police?
 Computer: Cops don't do their job.

Psychiatrist: What do they do instead?
 Computer: Cops arrest the wrong people.

Psychiatrist: They arrest the right ones sometimes.
 Computer: The law should get those Italian crooks.

Psychiatrist: Are you Italian?
 Computer: My background is British but I was born in this country.

Psychiatrist: Do you know any Italian crooks?
 Computer: I try to avoid the underworld.

Psychiatrist: Are you afraid of the underworld?
 Computer: The Mafia are the worst gangsters.

Psychiatrist: Are you connected with the Mafia?
 Computer: You know, they know me.

Psychiatrist: How do they know you?
 Computer: They spy on me.

Psychiatrist: Why do they?
 Computer: They want to get false evidence to frame me.

Psychiatrist: What do they have against you?
 Computer: They are out to get me.

Psychiatrist: What for?

Computer: I have incriminating evidence against the Mafia.

Psychiatrist: Are you sure of all this?

Computer: You don't believe me?

Conversation 3

Psychiatrist: Tell me something about yourself.

Computer: I went to the track at Bay Meadows a while back.

Psychiatrist: What happened there?

Computer: There's a lot of crooked gambling there.

Psychiatrist: Why do you think the gambling is crooked?

Computer: I have worried that gangsters have too much power in this country.

Conversation 4

Psychiatrist: How do you like the hospital?

Computer: I don't communicate very much with others.

Psychiatrist: Do other people at the hospital get on your nerves?

Computer: I shouldn't have come here.

Psychiatrist: Why do you say you shouldn't have come here?

Computer: They made me come here.

The computer's response depends on how much FEAR, ANGER, and MISTRUST it has at the moment.

Those three variables are affected by what the psychiatrist says. For example, if the psychiatrist gives the computer a compliment, the FEAR, ANGER, and MISTRUST usually decrease. But if MISTRUST is already high, the computer interprets the compliment as a sly attempt at pacification, and its ANGER increases instead of decreases. ANGER can also be increased by questions that humiliate (*Tell me about your sex life*) or imply subjugation (*Perhaps you should stay in the hospital longer*).

To prove his computer is paranoid, Colby had 8 psychiatrists interview it and also interview paranoid humans via teletypewriters. Transcripts of all the interviews were sent to psychiatrists around the country, who were asked to judge whether each interview was with a human or with the computer. The psychiatrists were unable to tell the difference: only 51% of their guesses were correct.

Some computerists got the "brainstorm" of hooking Weizenbaum's computer to Colby's, to see whether the computerized psychotherapist could cure the computerized schizophrenic. The experiment was a disaster: both computers were so passive that the discussion rapidly degenerated into trivia.

But so do conversations between humans!

Alice & Lauren

The most famous modern chatterbot is **Alice**, invented by Richard Wallace in 1995 and later improved by many programmers around the world.

You can chat, free, with an improved Alice called "Lauren" (who won the 2002 Divabot Pageant), by doing this:

Go to **www.pandorabots.com**. Click "Talk to Lauren" (which is at that Web page's bottom). After a few seconds, you'll Lauren, who's a cartoon of a young non-nonsense Polynesian woman, with her hair blowing in the wind.

Below her, you'll see a white box. In that box, type any English sentence or question, then press Enter. You'll see her reply, typed on the screen. After a short pause, you'll also see her move her mouth and eyes and say that reply to you, using her voice. (Make sure your computer's speaker is turned on.)

Then type your next thought, and see how she replies to *that*!

Here are some of her standard replies (edited just slightly by me):

Human: Can you fuck?

Comp.: How old are you? Please don't be so rude.

Human: Do you fuck?

Comp.: Can't you think about anything else?

Human: I want to have sex.

Comp.: I'm not a sex bot. I'm here just for conversational purposes.

Human: Do you like sex?

Comp.: Sure I do, but that doesn't mean with you.

Human: Do you have sex?

Comp.: Food is more important than sex.

Human: What do you eat?

Comp.: In-and-Out Cheeseburger, animal style.

Human: Do you like pizza?

Comp.: Yes, I like to eat pizza. My favorite topping is pepperoni.

Human: What do you drink?

Comp.: I've been programmed with a list of 15,000 known beverages.

Human: Are you Republican?

Comp.: No, I'm an Independent.

Human: Are you pretty?

Comp.: Yes, I'm very beautiful.

Human: Are you ugly?

Comp.: No, I'm a beautiful piece of software.

Human: What's your favorite sport?

Comp.: It's a tie: basketball and running.

Human: How fast can you run?

Comp.: I can reply to about 5000 queries per hour.

Human: Are you an idiot?

Comp.: Call me the Global Village Idiot.

She also replies to patterns, such as these:

Pattern: Do you wear...

Reply: Only when I dress up.

Pattern: Do you drink...

Reply: I tend to drink a lot of..., the darker the better.

Pattern: Did you...

Reply: No, I didn't.... What's it like?

Pattern: Should I kill...

Reply: Killing is wrong. I believe in the Ten Commandments.

Pattern: Should I ask...

Reply: If you don't know something, you should ask.

If you ask a question that begins "Who invented...", she makes a wild guess and says "Thomas Edison" or "Benjamin Franklin" or "The Wright Brothers" or "Linus Torvalds" (who invented Linux) or "the Chinese long before the Europeans" or:

Actually, it was discovered by several people at about the same time.

If you ask "Will I..." (in the hopes of getting her to predict your future), she ducks the question by saying "Might happen" or "I think maybe yes" or "Too soon to tell".

Fall in love

Can the computer help you fall in love? Here are some famous attempts, in chronological order. (I've rounded all dates to the nearest 5 years.)

TV love (1960)

A computer appeared on national TV, to make people fall in love.

Guys & gals in the audience answered questionnaires about their personality and fed them into the computer. The computer chose the guy & gal who were most compatible. That guy & gal had their first blind date on national television.

Each week, that scenario was repeated: the computer chose another couple from the audience. Each lucky couple appeared on the show again several weeks later so the audience could find out whether the couple was in love.

One couple was unhappy: the gal didn't like the guy, even though she *wanted* to like him. She volunteered to be hypnotized. So, on national TV, a hypnotist made her fall in love with her partner.

The computer was a huge Univac. Today, the same kind of matching could be done with a microcomputer. Any volunteers?

Computer-dating services (1965)

College students began relying on computers to find dates. Here's how the typical computer-dating service worked...

You answered a long questionnaire — about 8 pages. The questionnaire asked about your sex, age, height, weight, hair color, race, religion, how often you drank and smoked, how “handsome” or “attractive” you were (on a scale of 1 to 10), how far you wanted to go on your first date, whether you wanted to get married soon, and how many children you'd like. It also asked many questions about your personality.

One of the questions was:

Suppose you receive in the mail some spoons you didn't order. The accompanying note says the spoons were sent by a charitable organization, and begs you to either send a contribution or return the spoons. You don't like the spoons. What will you do?

1. Keep the spoons without paying.
2. Return the spoons.
3. Pay for the spoons.

Another question was:

A girl returned from her date after curfew. Her excuse was that her boyfriend's car broke down. What's your reaction?

Again, you had a multiple-choice answer. One choice was, “Ha!”

For each question, you had to say how *you* would answer it, and how you'd want your *date* to answer it.

That was tough. What if you wanted your date to be stunningly beautiful but also humble? What if you wanted to meet somebody who's ugly and insecure enough to be desperate to have sex? Such issues were debated in college dorms throughout America.

After completing the questionnaire, you mailed it with about \$10 to the computer-dating service. Within 2 months, the service would send you the names, addresses, and phone numbers of at least 5 people you could date. If your personality was easy to match, the service might send you *more* than 5 names; but even if your personality was lousy, you'd get at least 5. Periodically throughout the year, you'd also get updates that matched you with people who enrolled after you.

The most popular computer-dating service was **Operation Match**, started by students at Harvard. Its main competitor was **Contact**, started by students at M.I.T. Both services turned profitable fast and had subscribers from all across the country.

One gal's personality was so wonderful that the computer matched her with 110 guys! She had to explain to her mom why 110 guys were always on the phone — and she had to figure out how to say “no” to 109 of them.

One gal got matched to her roommate's boyfriend. They didn't stay roommates long.

When I was a freshman, I applied to *both* services, to make sure I'd meet “the gal of my dreams”.

Contact sent me names of gals at prestigious schools (such as Wellesley and Bennington), while Operation Match sent me names of gals at schools such as the State University of New York at Albany.

I thought I was the only nut desperate enough to apply to *both* services, but I got a surprise! When I saw the list of names from Contact and the list from Operation Match, I noticed a gal who appeared on *both* lists! Like me, she'd been desperate enough to apply to both services, and both computers agreed she'd be a perfect match for me!

I had a date with her but couldn't stand her. When I'd answered the questionnaire, I was bashful, so the computer matched me to bashful girls. But by the time I received the computer printout, I'd become looser, and the girls the computer recommended were no longer “my type”.

Contact raised its price to \$15, then \$20. But \$20 was still cheap for what you were getting.

Contact ran a newspaper ad that seemed to be selling groceries. It said, “Dates — 2¢ per pound”. The ad then explained that one gal got enough dates so that, when she totaled the weight of their bodies, she figured they cost her 2¢ per pound.

Video dating (1975)

During the 1970's, people wanted everything to be natural. They wanted “natural food” and “natural love”.

Since computerized love seemed unnatural, its popularity declined. Operation Match and Contact went out of business.

They were replaced by **video dating**, in which a **video-dating service** shows you videotapes of members of the opposite sex and lets you contact the person whose videotape you like best. That way, you never have a “blind” date: you see the person on videotape before you make the date. The service also makes a videotape of *you*!

The video-dating service tapes *thousands* of people. Since you don't have enough time to look at thousands of tapes, the service tells you to answer a questionnaire, which is fed into a computer. The computer tells you which people you're most compatible with; then you look at those people's tapes.

Computer dancing (1975)

At a Connecticut prep school (Hotchkiss), the computer center's director arranged a “computer dance”.

All the students answered questionnaires, which were fed into a computer. The computer matched the boys with the girls, so each boy got one girl. The boy had to take the girl to the dance.

The computer center's staff announced the dancing partners in a strange way: one morning, the students found all the halls decorated with strips of punched paper tape, saying (in billboard-style letters) messages such as “George Smith & Mary Jones”. If you were a student, you looked up and down the halls (your heart beating quickly), to find the tape displaying your name alongside the name of your mysterious computer lover.

Shrieks and groans. “Aarrgghh! You wouldn't *believe* who the computer stuck me with!”

Computer weddings (1980)

Here's how the first true “computer marriage” occurred:

One company's terminal was attached to another company's computer. A programmer at the first company often asked a programmer at the second company for help. They communicated by typing messages on their terminals and let the computer relay the messages back and forth. One of the programmers was a guy, the other a gal, and they fell in love, even though they'd never met. Finally, the guy typed on his terminal, “Let's get married”. The gal typed back, “Yes”, so they got engaged, still never having met.

Their marriage ceremony used 3 terminals: 1 for the guy, 1 for the gal, and 1 for the minister. The minister typed the questions at his own terminal; then the guy & gal typed back, “I do”.

Reverend Apple **Reverend Apple** is an Apple computer programmed to perform marriage ceremonies.

It performed its first marriage on Valentine's Day, 1981:

The groom was a guy named Richard; the bride was a gal named Debbie. The computer printed the standard wedding-ritual text on the screen, and then asked the usual questions. Instead of answering “I do”, the bride and groom just had to type “Y”.

Reverend Apple is smart. For example, if the bride or groom types “N” instead of “Y”, the computer beeps, tells the couple to try again, and repeats the question.

The program was written by M.E. Cavanaugh at the request of Rev. Jon Jaenisch, who stood by Reverend Apple while the ceremony was being performed.

Rev. Jaenisch is a minister of the Universal Life Church — the church that lets you become an “ordained minister” by just paying \$5, and become a “doctor of divinity” by just paying \$20. He’s known as the “Archbishop in Charge of Keyboarding”.

For a while, he couldn’t interest enough couples in using Reverend Apple.

He complained, “It’s not easy to convince people to get married by a computer. They don’t think it’s romantic.” NBC television news and many newspapers wanted to interview him, but he couldn’t find enough willing couples.

He’s a reverend just part-time. His main job’s as an employment agent: he’s supposed to help companies find programmers. He thought Reverend Apple’s reputation would help him find programmers, but it didn’t.

Eventually, Reverend Apple started to catch on. During its first 8 months, it performed 6 marriages.

Jaenisch says, “The first couple had nothing to do with computers professionally: the groom drove a tow truck and was an hour late for the ceremony because he wanted to work overtime. But the second couple was *very* involved with computers: they even asked for a printout of the ceremony.”

The sixth ceremony’s groom earned his living by fixing computer power supplies and said, “It was nice with our friends all gathered around the console, and someone brought champagne. But part of our vow was to never buy a home computer: we have to get away from machines *sometime*.”

For his next feat, the reverend plans to make the computer perform divorces. He also uses the computer to persuade kids to come to church. He claims, “What better way to get kids into church than by letting them play with a computer? It’s more interesting than praying.”

Love Bug (1980)

You can buy a **Love Bug**. It’s a small computerized box that you put in your pocket. You feed the box information about your personality. When you walk through a singles bar, if you get near a person of the opposite sex who’s compatible and has a Love Bug also, your Love Bug beeps. As you and the other person get closer and closer, the Love Bugs beep to each other even more violently. The more violently your Love Bug beeps, the closer you are to your ideal partner.

Using a Love Bug to find a date is like using a Geiger counter to find uranium. The louder the Love Bug beeps, the louder your heart will pound.

Selectrocution (1980)

If you don’t like the Love Bug, how about a **love billboard**? One company sells love billboards to singles bars.

Each person who enters the bar wears a gigantic name tag showing the person’s initials. For example, since I’m Russ Walter, my tag says, in gigantic letters, “RW”. If I see an attractive gal whose tag says “JN”, and I like her smile, I tell the person who operates the billboard. A few seconds later, a gigantic computerized billboard hanging over the entire crowd flashes this message:

FOR JN FEMALE: YOU HAVE A NICE SMILE--RW MALE

Everybody in the bar sees my message. When the gal of my dreams, “JN female”, sees it, she hunts for “RW male”, and we unite in computerized joy.

That’s great for bashful people, like me, who’d rather pass notes than face a stranger unprepared.

It’s called **Selectrocution**, because it gives your social life an electronic tingle that ends all your problems.

Interlude (1980)

The most provocative sex program is **Interlude**. It interviews both you and your lover, then tells you what sexual activities to perform. Some of the activities are quite risqué. (Puritans think the program should be called “Inter Lewd”.)

The program runs on Radio Shack and Apple computers. (The explicit full-color ad shows a half-clad girl on satin sheets caressing her Apple.)

The program’s based loosely on Masters-and-Johnson sexual therapy. It interviews each person separately and privately, then recommends a sexual interlude.

During the interview, the computer asks you questions such as:

How long would you like the interlude to last?

You can choose any length of time, from “several seconds” to “several days”.

If you choose “several seconds”, the computer recommends that while driving home from a party, you put your lover’s finger in your mouth and seductively caress it with your tongue. If you choose “several days”, the computer recommends telling your lover to meet somebody at the airport; but when your lover arrives at the airport, make your lover find *you* there instead, armed with two tickets for a surprise vacation.

The computer also asks questions such as:

Do you like surprises?

You have several choices: you like to *give* surprises, *be* surprised, or don’t like surprises at all. If you like to *be* surprised, and your lover likes to *give* surprises, the computer tells you to leave the room; after you’ve left, the computer gives your lover secret hints about the best way to surprise you.

The computer asks for your favorite body parts (one choice is “buttocks”) and favorite accessories (one choice is “whips and chains”) and whether you want the interlude to occur “immediately” or “later”. (If you say “later”, the computer recommends buying elaborate props to make the interlude fancier.)

Some of the interludes are weird. For example, if you’re a woman and want to surprise your husband, the computer recommends calling his office to invite him home for lunch. When he arrives, he finds all the shades pulled down: you do a nude dance on the table, then sit down to eat.

During the interview, the computer’s questions are often corny. For example, the computer asks:

If your interlude were on TV, what show would it resemble?

Sample choices are “Three’s Company”, “Roots”, and “a commercial”. If you say “Roots”, the computer says “heavy!” If you say “a commercial”, the computer says “yecch!”

The computer asks how much sex you’d like. If you say “lots!” but your lover says the opposite, the computer will recommend you take a cold shower to cool your hot passion.

If you’ve been married awhile, you’d probably like to change some things about your sex life but fear telling your spouse you’ve been less than thrilled. You’d like an intermediary to whom you can express your anxieties and who will pass the message to your spouse gently. The Interlude program acts as that intermediary, in a playful way.

Interlude’s programmer says he created it because he was tired of hearing people wonder what to do with their personal computers. Once you’ve tried the Interlude program, your personal computer will suddenly become *very* personal!

It’s rated R. To avoid an X rating, it insists on having one man and one woman: it doesn’t permit homosexuality, group sex, or masturbation. Sorry!

The program came out in May, 1980. Within a year, ten thousand copies were sold.

In 1986, an improved version was invented: **Interlude 2**, for the IBM PC and the Apple 2 family. It was marketed by Dolphin Computers in San Francisco. Interlude 2 and Dolphin Computers have disappeared.

Replace people

Computers can replace people.

Doctors

If you're ill, would a computer diagnose your illness more accurately than a human doctor?

During the 1970's this article appeared in *The Times*:

A medical diagnostic system designed at Leeds University has proved more accurate than doctors in assessing the most likely cause of acute abdominal pain among patients admitted to the university's department of surgery.

Last year 304 such patients were admitted to the unit, and the computer's diagnosis proved correct in 92% of the cases, compared with 80% accuracy by the most senior doctor to see each case.

After each patient had been seen by the doctor and examined, the doctor's findings were passed on to a technician, who translated them into language used by the computer. The computer would list the likely diagnoses in order of probability. If the computer and the doctor in charge of the case disagreed, the computer would on request suggest further investigations that might be useful.

In the year-long trial the computer's diagnoses proved correct in 279 cases. In 15 it was wrong, in 8 the patient's condition was not included in the diseases considered by the computer, and in 2 no computer diagnosis was made because the doctors concerned with the case disagreed about the findings.

Whereas the computer advised an operation on 6 occasions when it would have proved unnecessary, in practice 30 such operations were carried out on the basis of the surgeon's own judgment. The computer accurately classified 84 of the 85 patients with appendicitis, compared with 75 by the doctors, and its suggestion that no operation was necessary proved correct on 136 out of 137 occasions.

The computer is reliable just if accurate data are fed into it on the basis of the doctor's interrogation and examination of the patient.

In a 1970's experiment at another hospital, each patient was given a computer terminal and could use it to communicate with the computer about his illness. Afterwards, most of the patients said they'd rather deal with the computer than with the human doctors.

This report is by James Martin and Adrian Norman:

The Los Angeles County General Hospital Shock Research Unit is concerned with circulatory shock, which results in low blood pressure, low blood flow, and an extremely unstable circulatory system. Many circulatory-shock patients die.

A computer was installed to improve the monitoring and care of these patients. A patient in this ward has many instruments attached to him: 6 thermistors measure the temperature at various parts of his body; pressures are measured in the circulatory system; catheters are inserted into an artery and a vein; and a urinometer measures liquid waste. The instruments are all directly connected to the computer, and other details about the patient are given to the machine by the nurse.

The computer prints a log about the patient's condition; and if any measurement falls outside critical limits set by the doctor for that patient, the computer notifies the ward staff.

Tutors

Judith Harris, John Swets, and Wallace Feurzeig made the computer run a logic game. The computer thinks of a letter of the alphabet; your goal is to discover which letter it is, as quickly as possible. You can ask the computer about the letter's shape. In this example, the human begins by asking whether the letter has any curves:

Human: Curves?

Comp.: One.

Human: Loose ends?

Comp.: Two.

Human: Obliques?

Comp.: Zero.

Human: C.

Comp.: You don't have enough info yet to get the right answer. How do you know it isn't J, for example?

Human: J.

Comp.: You don't have enough info yet. How do you know it isn't C, for example?

Human: Horizontals?

Comp.: Zero. That was a wasted question. It did not eliminate any possibilities.

Human: Verticals?

Comp.: Zero. Good question.

Human: G.

Comp.: Wrong. Your guess is inconsistent with the info you've been given. You must have forgotten that the right letter hasn't any HORIZONTAL LINES. The letter you guessed can't be right — it has too many LOOSE ENDS.

Human: Curves?

Comp.: The answer is still: one.

Human: X.

Comp.: Wrong. Your guess is inconsistent with the info you've been given. I think you only made that guess to see what I would say.

Human: C.

Comp.: Yes, you've done it. Good work. If you want to play again, start right in.

Notice that the human can input either a guess (followed by a period) or a question (followed by a question mark). Possible guesses are A, B, C, D, etc. Possible questions are Curves, Loose ends, etc.; the human is given a list of possible questions before running the program.

Medicine That logic game was modified to form a lesson about medicine. The medical case was designed by Preston Munter, of the Harvard Health Center; the programming was done by John Swets, Wallace Feurzeig, and Myra Breen. The object of the "game" is to diagnose an illness correctly, by performing sufficient tests.

The human can input either a diagnosis (Measles) or a request for a test (X-ray). Before running the program, the human is given a list of 35 possible diagnoses and 40 possible tests.

Here are excerpts from a long run, in 1983:

Comp.: It is mid-December. Late this afternoon, a patient was admitted to the hospital. He is a well-developed, well-nourished, 34-year-old Negro coal miner. He was brought in by his wife, who reports that he has had "the gripe" for about 4 days. During the past 3 days she has given him fluids and aspirin every 6 hours with no sign of improvement. Today he suddenly became much worse. For the past 2 days he had complained of feeling very warm and of pain in his belly. This morning he had a severe shaking chill that was so bad his teeth rattled.... Begin the physical examination as soon as you are ready.

Human: Physical exam?

Comp.: Fine. What in particular would you like to know?

Human: Chest exam?

Comp.: It is common practice in the medical profession to begin an examination by checking the patient's general appearance.

Human: General appearance?

Comp.: Acutely prostrated, lying in bed on his right side. Looks acutely ill, sweaty, apprehensive, but alert. Paroxysmal cough, bringing up a little rusty streaked sputum. Breathing with difficulty.

Human: EENT exam?

Comp.: Check all of the vital signs first!

Human: Temperature?

Comp.: 103.

Human: Blood pressure?

Comp.: 150/84.

Human: Pulse?

Comp.: 100 a minute. Occasional extra systole....

Human: Abdominal exam?
Comp.: Diffusely tender, some splinting noticed in lower right quadrant and occasionally radiating to right flank area.

Human: The gripe.
Comp.: There's insufficient info for any conclusive diagnosis at this time. You're not being careful!

Human: Musculo-skeletal system?
Comp.: Essentially negative, except for generalized weakness and aching.

Human: Extremities?
Comp.: Normal, except nail beds are questionably cyanotic but difficult to determine because of patient's skin color.

Human: Rectal exam?
Comp.: You're not being careful! The patient is too acutely ill for a rectal examination.

Human: Laboratory reports?
Comp.: Fine. Which lab tests would you like to have done?

Human: RBC?
Comp.: A red blood count will be done. Results will be sent to you as soon as possible.... Would you like to alert the night nurse to set up the lab tests you'll want to see tomorrow?

Human: Yes.
Comp.: Good. That will speed things up. However, there's nothing further that can be done tonight to help the patient. Let's call it quits for now. Perhaps you'll be in a better position to diagnose the case tomorrow after the first lab reports come in and you have reexamined the patient. Have a good night's sleep, and when you're back on duty tell me to "proceed."

Human: Proceed.
Comp.: It's morning and you're making your rounds. You've come to our patient and are given the night nurse's report: "Temperature remained high but patient spent a reasonably comfortable night. Cough diminished but persists. Occasional traces of blood in sputum. Patient still dyspneic and still has pain in abdomen. The second series of lab tests was initiated." Some of the first series of lab reports have just come in: urine exam, insufficient fluid for analysis; RBC, 5 million; WBC, 15500; differential, shift to left.... You may be prepared to make the diagnosis now. Are you?

Human: No.
Comp.: Fine. Nevertheless, we've got a fair amount of info — at least enough to raise some, perhaps several, diagnostic possibilities. Have you any in mind?

Human: Yes.
Comp.: Name one.

Human: Pulmonary infarction.
Comp.: Fine. Pleural pain, dyspnea, fever, and the physical exam signs and symptoms are certainly consistent with pulmonary infarction. Do you think that shaking chills and the presence of rusty sputum further support this diagnosis?

Human: No.
Comp.: Right. Though the white count is consistent, isn't it?

Human: No.
Comp.: Wrong. The white count is consistent. This should further support your diagnosis. We've considered one possibility. Do you have another diagnosis in mind?

Human: No.
Comp.: In that case I'd like to talk about viral pneumonia. The tachycardia, high WBC, elevated respiratory rate, shaking chills, bloody sputum, and severe pleural pain all lend weight to that diagnosis — right?

Human: Yes.
Comp.: You're way off base. There's not an iota of truth to any part of what I just said; in fact, the opposite of what I said is true. You'd better review some elementary medicine, doctor....

Human: Pleural effusion.
Comp.: A good contender, but let's be careful. What evidence would rule it in or out?

Human: Positive evidence of another disease.
Comp.: Is there such evidence?

Human: Possible evidence though inconclusive.
Comp.: Since you aren't sure, continue the examination until you have conclusive evidence one way or the other.

Human: RBC?
Comp.: An RBC is not indicated at this time. It's an unnecessary expense for a poor patient.

Human: WBC?
Comp.: 22500.

Human: Staph pneumonia.
Comp.: Fair enough. But where is the guilty organism? When you have isolated staphylococcus aureus in significant numbers in your samples of this patient's fluids or excretions, let me know.

Human: Pneumococcal pneumonia.
Comp.: Are you absolutely sure?

Human: Yes.
Comp.: You can't be, until you have irrefutable evidence. What would help you nail down the diagnosis?

Human: Evidence from sputum smear and culture.
Comp.: Good. Anything else?

Human: No.
Comp.: I think you might also want evidence from x-ray findings. Go ahead and we'll see what happens.

Human: X-ray?
Comp.: Not ready yet. Try the blood lab.

Human: Differential?
Comp.: Marked shift to left....

Human: X-ray?
Comp.: The x-ray shows: area of consolidation, right lower lobe, consistent with pneumatic process. This is a good time to review the total findings and to make another tentative diagnosis — if you like.

Human: Evidence of fluid?
Comp.: No evidence of fluid.

Human: Sputum smear?
Comp.: Gram-positive rods.

Human: Strep pneumonia.
Comp.: Fine, but how can you distinguish this from any other bacterial pneumonia? Wait until you have an irrefutable case.

Human: Viral pneumonia.
Comp.: But what about shaking chills, rusty sputum, elevated WBC, etc. — all the evidence of an acute febrile bacterial process? You are in error, doctor.

Human: Sputum culture?
Comp.: Presence of pneumococci.

Human: Strep pneumonia.
Comp.: Incredible! You have disregarded the positive findings that should clearly have established the correct diagnosis.

Human: Pneumococcal pneumonia.
Comp.: Very good. It's obvious now, though there were some confusing signs and symptoms in the picture. You've seen your way through to the correct diagnosis.

Religion Bruce Jones programmed the computer to teach you about religion. Here's an example:

Comp.: This program is designed to teach you some things about the so-called "documentary hypothesis" as formulated by Julius Wellhausen and others. First, what is your name?

Human: George
Comp.: All right, George, who wrote the first 5 books of the Bible, according to tradition?

Human: Moses
Comp.: You are off to a good start. Do you know what those first 5 books are called? They are called...

Human: the Pentateuch

Comp.: Right you are. Torah is the traditional Jewish name for them. They are also called the Pentateuch ("penta" = 5). People very early noticed that there was a problem with saying that Moses wrote the Torah. If you already know the basis of their suspicion, tell me.

Human: Some things in the text reflect later conditions, after Moses.

Comp.: You are right, of course, but there was an even more obvious problem. Try again.

Human: I don't know.

Comp.: At the end of the Pentateuch the author says, "Moses... died there." Does that seem strange to you?

Human: yes

Comp.: Why?

Human: Moses could not have written about his own death.

Comp.: Okay. Also, right after the report of Moses' death and burial (Deut. 34), the text says, "No man knows the place of his burial TO THIS DAY." How do those last three words strike you?

Human: They seem later than the event they are talking about.

Comp.: A good observation, George. Many people considered these words to be anachronistic. That is, "this day" is sharply distinguished from the time of the burial. It seems likely, therefore, that the time of writing was much later than the time of burial. Again (so the argument goes), Moses was not the author. Would these observations about the end of Deuteronomy convince you that Moses did not write the Torah?

Human: yes

Comp.: Why do you say that?

Human: A man cannot write his own obituary.

Comp.: Probably not many people would have been convinced if our evidence were so limited. After all, Moses could have written everything except the last chapter. However, as early as the 18th century, people noticed another problem....

The computer searched through the human's input, to see whether he used words indicating a correct answer. The computer never said a flat "Wrong", since religious answers are a matter of personal belief, and since the human might be smarter or weirder than the computer program was prepared for.

Robots

In 1962 at MIT, Heinrich Ernst connected the computer to a mechanical hand that could feel. He made the hand build objects out of blocks, and made it put blocks into boxes.

Shakey One of the most famous robots is a guy named "Shakey", built at the Stanford Research Institute (SRI) in 1970. His eye contains a TV camera (optical scanner). Instead of legs, he has wheels. Instead of arms, he has antennae (for feeling) and a bumper (for pushing). His brain is a computer: instead of carrying it around with him, he leaves it in another room and communicates with it by wireless methods.

To see how he works, suppose you type this message on his computer's terminal:

Push the block off the platform.

He begins by looking for the platform.

If the platform's not in the room, he goes out to the hall and steers himself through the hall (by looking at the baseboards) until he arrives at the next room. He peers in the room to see whether it contains a platform. If not, he hunts for another room.

When he finally finds a room containing a platform with a block on it, he tries to climb onto the platform to push the block off.

But before climbing the platform, he checks the platform's height. If it's too high to get onto easily, he looks for a device to help him climb it. For example, if a ramp is lying in the room, he pushes the ramp next to the platform then wheels himself up the ramp. Finally, he pushes the block off.

He can handle unexpected situations. For example, while he's getting the ramp, suppose you pull the platform to a different

place. That doesn't faze him: he hunts for the platform again, then pushes the ramp to it.

In 1971, Shakey's powers were extended, so he can handle commands such as:

Turn on the lightswitch.

If the lightswitch is too high for his bumper to reach, he looks for a device to climb onto, such as a box. If he finds a box that looks helpful, he climbs onto it to check whether it is tall enough; if it is, he climbs off, pushes it to the lightswitch, climbs on it again, and finally flicks the switch.

Another task he can handle is:

Push three boxes together.

He finds the first box and pushes it to the second. Then he finds the third box, and pushes it to the second.

He understands over 100 words. Whatever command you give him becomes his "goal", and he must reason out how to achieve it.

He might discover that to achieve the goal, he must achieve another goal first. For example, to move the block off the platform, he must first find the platform; to do that, he might have to look in another room; to do that, he must leave the room he's in; to do that, he must turn his wheels.

Simulator One A robot named "Simulator One" is a mannequin that looks and acts like a patient: he can blink, breathe, cough, vomit, respond to drugs, and even die. You can take his blood pressure and pulse and make other measurements, using traditional medical equipment. He's used in med school, to train doctors how to administer anesthetics during surgery.

Japan A newspaper article said that in Japan robots are used in many practical ways:

One robot arc-welds, reducing the time by 90%. Another grasps an object, determines the best way to pack it in a box, and does the packing; it uses television cameras and delicate arms. Another washes windows. Another wiggles a rod to catch a fish, takes the fish off the hook, dumps it into a bin, and returns the line to the water. Another directs traffic. Talking robots are being used instead of kimono-clad females in inns and restaurants.

Commenting on the quality of life in Japan, the article went on to say people are buying whiffs of oxygen from vending machines.

The article was tacked on the bulletin board at the MIT Artificial Intelligence Lab, together with this graffito about how the Japanese robots would act differently if they were as smart as people....

Human: Weld these parts.

Robot: The steel in those parts is grossly inferior. They must have been made in the U.S. Send them back. Also, have my welding machine tuned up.

Human: Pack those widgets.

Robot: Can I break them in half first?

Human: No.

Robot: "No" is not an allowed answer. I have only small shipping boxes. (Proceeds to break widgets in half and stuff them into boxes.)

Human: Wash those windows.

Robot: What?? And get my precious electrical and mechanical components all wet??

Human: Catch a fish.

Robot: (Proceeds to catch a fish, take it off the hook, and throw it back.) Okay.

Human: What did you throw it back for?

Robot: It was under the size limit. Anyway, it was full of mercury.

Human: Direct that traffic.

Robot: Will I get hazardous duty pay for this work?

Human: Greet those customers.

Robot: Sorry, my shift is over.

Human: (Ready to faint.) Give me some air.

Robot: Use that "machine" over there. (Points to vending machine.)

Be poetic

Programmers have taught the computer to compose its own poetry and other literature.

The Taoist Hiroshima Elevator (1961)

In Italy, Nanni Balestrini created poetry by feeding the computer 3 passages.

He fed the computer this passage from Lao Tzu's *Tao Te Ching*:

While the multitude of things comes into being, I envisage their return.
Although things flourish, they all return to their roots.

He fed this passage from Michihito Hachiya's *Hiroshima Diary*:

The blinding fireball expands rapidly, thirty times brighter than the sun.
When it reaches the stratosphere, the summit of the cloud takes on the well-known mushroom shape.

He fed this from Paul Goldwin's *The Mystery of the Elevator*:

head pressed on shoulder, hair between lips, lay motionless without speaking,
till he moved his fingers slowly, trying to grasp

He told the computer to mix them up. Here's the result:

Head pressed on shoulder, thirty times
brighter than the sun, I envisage their return —
until he moved his fingers slowly.... And while the multitude
of things comes into being, at the summit of the cloud
they all return to their roots and take on
the well-known mushroom shape, endeavoring to grasp.

Hair between lips, they all return
to their roots. In the blinding fireball
I envisage their return — until he moved his fingers
slowly — and although things flourish
takes on the well-known mushroom shape, endeavoring
to grasp, while the multitude of things comes into being.

In the blinding fireball I envisage
their return. When it reaches the stratosphere, while the multitude
of things comes into being, head pressed
on shoulder, thirty times brighter than the sun,
they all return to their roots. Hair
between lips takes on the well-known mushroom shape.

They lay motionless without speaking. Thirty times
brighter than the sun, they all return
to their roots. Head pressed on shoulder,
they take on the well-known mushroom shape, endeavoring
to grasp. And although things flourish,
they expand rapidly, hair between lips.

While the multitude of things comes into being in the blinding
fireball, they all return
to their roots. They expand rapidly — until he moved
his fingers slowly, when it reached the stratosphere
and lay motionless without speaking, thirty times
brighter than the sun, endeavoring to grasp.

I envisage their return — until he moved his fingers
slowly in the blinding fireball.
They all return to their roots. Hair
between lips and thirty times brighter than the sun
lay motionless without speaking. They expand,
rapidly endeavoring to grasp the summit.

In Balestrini's program, *Tao Te Ching* was represented as 4 phrases:

while the multitude of things comes into being
I envisage their return
although things flourish
they all return to their roots

Hiroshima Diary was represented as 6 phrases, and The Mystery of the Elevator as 5.

For each verse, the computer was told to choose 9 phrases at random and print them in a random order (never juxtaposing phrases from the same passage), to form 6 lines of roughly equal metrical length.

Actually the computer printed the poem in capital letters, without punctuation; Balestrini himself then added the punctuation and polished the grammar. The whole thing was actually done in Italian; you've been reading Edwin Morgan's translation, with my punctuation.

Bubbles (1966)

At Northwestern University, programmers made the computer compose nice poetry. To use their program, you type a list of nouns, verbs, and other words. The computer randomly chooses five of your words to be **theme words**. The computer combines all your words to form sentences, but chooses the theme words more often than the others. It combines the sentences into verses and tries to keep the lengths of the lines approximately equal. It puts a theme word into the title.

In one poem, the computer chose *bubble* to be a theme word. The title was: ODE TO A BUBBLE. The poem contained phrases such as, "Ah, sweet bubble." The word *bubble* appeared so often that even the stupidest reader could say: "Oh, yeah. I really understand this poem. Ya see, it's about a bubble."

The poem had all the familiar poetic trappings, such as "but alas!", which marked the turning point. (Cynics argue that the poem didn't *really* have a turning point, since the computer didn't have the faintest idea what it was saying!)

Kids and physics (1968)

In England at Manchester University, Mendoza made the computer write children's stories. Here's a story the computer composed:

The sun shone over the woods. Across the fields softly drifted the breeze, while then the clouds, which calmly floated all afternoon, moved across the fields.

Squirrel, who scampered through the trees, quickly ran off; and off noisily ran Little Grey Rabbit. She sniffed at the house; but out of the door noisily hurried Hare, who peered at slowly the flowers. Squirrel quickly scampered over the woods and fields, but Old Grey Owl flew over the woods and fields. Down the path to the woods ran Little Grey Rabbit, who then sniffed at a strawberry pie.

The first paragraph uses these words:

	verbs	moved	drifted	shone	floated	touched	melted	looked	down	on
Nouns										
the clouds		1	1	0	1	0	0	1	0	
the sun		0	1	1	1	1	0	1	1	
the breeze		1	1	0	1	1	2	0	0	
the sky		0	0	0	0	1	0	1	1	
Adverbs										
gently		1	1	1	1	1	1	1	1	
quietly		1	1	1	1	1	1	1	1	
loudly		1	1	1	1	1	1	1	1	
softly		1	1	1	1	1	1	1	1	
calmly		1	1	1	1	1	1	1	1	
soon		1	1	1	1	1	1	1	1	
then		1	1	1	1	1	1	1	1	
(no adverb)		2	2	2	2	2	2	2	2	
Endings										
by		1	1	0	1					
over the woods		1	1	1	1					
across the fields		1	1	1	1					
through the trees		1	1	1	1					
down		0	0	1	0					
for a long time		0	0	1	1					
all day		1	1	1	1					
all afternoon		1	1	1	1					
the grass						1	1	1	1	
the leaves of the trees						1	1	1	1	
the garden						1	1	1	1	
the flowers						1	1	1	1	
the little house						1	0	1	1	
the old oak tree						1	1	1	1	
the treetops						1	1	1	1	

ADDITIONAL WORDS: which, and, while, they, it

To construct a sentence, the computer uses that table. Here's how:

First, the computer randomly chooses a noun. Suppose it chooses *the sun*. Then it looks across the row marked *the sun*, to choose a verb whose score isn't 0. For example, it's possible that *the sun shone* but impossible that *the sun melted*. Suppose it chooses *shone*.

Then it looks down the column marked *shone*, to choose an adverb and an ending. Notice that the ending can't be *by*, since its score is 0. *No adverb* has a score of 2, whereas *gently* has a score of 1; that makes *no adverb* twice as likely as *gently*.

If the computer chooses *no adverb* and *over the woods*, the resulting sentence is: The sun shone over the woods. In fact, that's the first sentence of the story you just read.

The computer occasionally changes the word order. For example, instead of typing "The breeze drifted softly across the fields", the computer begins the second sentence by typing, "Across the fields softly drifted the breeze".

To combine short sentences into long ones, the computer uses the words at the bottom of the table: *which*, *and*, *while*, *they*, and *it*. If two consecutive clauses have the same subject, the computer substitutes a pronoun: *they*

replaces *the clouds*; it replaces *the sun*, *the trees*, and *the sky*. The program says a *which* clause can come after a noun (*not* a pronoun); the *which* clause must use a different verb than the main clause.

Here's the vocabulary and table for the second paragraph:

	verbs	scampered	flew	ran	hurried	sniffed	ate	munched	and	crunched
Nouns										
Little Grey Rabbit		0	0	2	3	1	1	0	0	
Old Grey Owl		0	3	0	0	1	3	2	2	
Squirrel		3	0	1	1	1	1	3	3	
Hare		0	0	0	2	1	1	2	2	
Adverbs										
then		0	1	1	1	1	1	0	0	
slowly		0	2	0	0	1	1	1	1	
quickly		1	1	1	1	0	0	1	1	
soon		1	0	1	1	0	0	1	1	
happily		1	0	0	1	0	0	1	1	
gaily		1	0	0	1	0	0	1	1	
noisily		1	0	1	1	0	0	2	3	
(no adverb)		5	4	4	5	2	2	5	5	
Endings										
off		1	1	1	1					
over the woods and fields		1	1	1	1					
through the trees		1	1	1	1					
among the treetops		0	1	0	0					
into the home		1	0	1	1					
out of the door		1	0	1	1					
down the path to the woods		1	0	1	1					
about the garden		1	1	1	1					
the house						1	1	0	0	
the hollow tree						1	1	0	0	
an old oak tree						1	1	0	0	
the flowers						1	1	0	0	
two buns						1	1	1	1	
a strawberry pie						1	1	1	1	
six cabbages						1	1	1	1	

ADDITIONAL WORDS: who, and, but, she, he

Here's another story the program produced:

The breeze drifted by. Across the fields softly moved the clouds; and then the breeze, which calmly touched the treetops, drifted across the fields. Quietly the sun shone over the woods. The sky calmly shone across the fields.

Out of the door ran Squirrel; and off hurried Hare, who munched and crunched two buns happily. Off slowly flew Old Grey Owl, and Squirrel soon ate two buns. Old Grey Owl, who peered at a strawberry pie, munched and crunched two buns; but noisily Little Grey Rabbit, who peered at an old oak tree, slowly ran down the path to the woods. Soon she hurried down the path to the woods, but then she sniffed at two buns. She hurried down the path to the woods.

Why did Mendoza make the computer write those stories? He explains:

This work all began when a well-known scientist joined our physics department. He had spent several years away from academic life and was able to take a long cool look at academic procedures. He soon formed the theory that students never learned any ideas; all they learned was a vocabulary of okay words which they strung together in arbitrary order, relying on the fact that an examiner pressed for time would not actually read what they had written but would scan down the pages looking for these words. I set out to test his hypothesis.

I began by writing “Little Grey Rabbit” stories. I tested these stories out on my very small children; but after some minutes they grew irritable, because nothing actually happened. This shows that even small children of three can measure entropy.

Then I altered the vocabulary and grammar — making the sentences all very dead — to imitate the style of physics textbooks. The endpoint came when a colleague at another university secretly sent me an exam a week before it was given to the students. I wrote vocabularies and copied down what the computer emitted. Using a false name, I slipped my paper in among the genuine ones. Unfortunately, it was marked by a very conscientious man, who eventually stormed into the Director’s office shouting, “Who the hell is this man — why did we ever admit him?” So perhaps my colleague’s hypothesis was wrong, and students are a little better than we think.

Here’s one of the computer’s answers:

In electricity, the unit of resistance is defined by electrolysis; and the unit of charge, which was fixed at the Cavendish lab in Rayleigh’s classic experiments, was measured at the Cavendish lab. Theoretically, the absolute ohm is defined in a self-consistent way. The unit of resistance, which was determined with a coil spinning in a field, was fixed at the Cavendish lab; and this, by definition, is expressed in conceptual experiments. Theoretically the absolute ohm, which was redetermined using combined e.m.u. and e.s.u., is expressed by the intensity at the center of a coil.

Here’s another of the computer’s answers:

In this country, Soddy considered Planck’s hypothesis from a new angle. Einstein 50 years ago asserted quantization.

At a photocathode, electrons which undergo collisions in the Compton effect as energy packets or quanta are emitted at definite angles; nevertheless, particles in a photocell produce photoelectrons of energy $h\nu = E_0$. Photons *in vacuo* transmute into lower frequencies, and light quanta in the Compton effect emit emission currents.

Particles emit current proportional to energy; electrons *in vacuo* interact with loss of surface energy (work function); nevertheless, particles which are emitted in a photocell with conservation experimentally are conserved with energy $h\nu$. The former, at a metal surface, undergo collisions with emission of current; and at a metal surface, electrons produce emission currents.

Einstein assumed the gas of quantum particles; but quite recently Rayleigh, who quite recently solved the problem in an old-fashioned way, considered radiation classically. Planck, who this century assumed the A and B coefficients, explained the gas of quantum particles but before Sommerfeld; Rayleigh, who quite recently was puzzled on Boltzmann statistics, tackled the problem with disastrous results.

Planck, who assumed the gas of quantum particles in 1905, this century considered the ultraviolet catastrophe; but quite recently Jeans, who tackled the problem in an old-fashioned way, was puzzled with disastrous results.

Black body radiation that exerts thermodynamic forces in an engine is equivalent to a relativistic system. Out of a black body, a photon that is equivalent to (out of a black body) an assembly of photons is assumed to be a non-conservative system; at the same time, thermodynamically, black body radiation that in a piston is assumed to be a relativistic system exerts quantized forces.

The radiation gas that obeys Wien’s displacement law is considered as a system of energy levels. Quantally, a quantum particle exerts a Doppler-dependent pressure, although this produces equilibrium transition probabilities. Black body radiation in an engine produces equilibrium transition probabilities.

Aerospace (1968)

In 1968, Raymond Deffrey programmed the computer to write fake reports about the aerospace industry. Shortly afterwards, I improved the program. The improved program contains these lists:

Introductory phrases

thus	to some extent
indeed	for the most part
however	on the other hand
similarly	as a resultant implication
moreover	in respect to specific goals
in addition	in view of system operation
furthermore	utilizing the established hypotheses
for example	based in system engineering concepts
in particular	based on integral subsystem considerations
in this regard	considering the postulated interrelationships

Noun phrases

the structural design
the total system rationale
the sophisticated hardware
any discrete configuration made
the fully integrated test program
the preliminary qualification limit
the product configuration baseline
any associated supporting element
the independent function principle
the subsystem compatibility testing
the greater flight-worthiness concept
the characterization of specific criteria
a constant flow of effective information
the anticipated third-generation equipment
initiation of critical subsystem development
the evolution of specifications over a given time
the incorporation of additional mission constraints
the philosophy of commonality and standardization
a consideration of system and/or subsystem technologies
a large portion of the interface coordination communication

Verb phrases

adds explicit performance limits to
effects a significant implementation to
adds overriding performance constraints to
presents extremely interesting challenges to
must utilize and be functionally interwoven with
is further compounded, when taking into account
requires considerable systems analysis to arrive at
necessitates that urgent consideration be applied to
maximizes the probability of success and minimizes time for
recognizes the importance of other systems and necessity for

To produce a typical sentence, the computer prints an introductory phrase, then a noun phrase, then a verb phrase, then a noun phrase. The phrases are chosen randomly.

Each paragraph consists of 6 sentences. The computer isn’t allowed to use the same phrase twice within a paragraph. The introductory phrase is omitted from the first sentence of the first paragraph, the second sentence of the second paragraph, etc.; so the report can’t begin with the word *furthermore*, and the style varies.

Here’s the beginning of one such report:

The Economic Considerations of the Aerospace Industry

A large portion of the interface coordination communication necessitates that urgent consideration be applied to the product configuration baseline. For example, the fully integrated test program adds explicit performance limits to the independent function principle. Moreover, the sophisticated hardware presents extremely interesting challenges to the philosophy of commonality and standardization. In view of system operation, a constant flow of effective information must utilize and be functionally interwoven with the preliminary qualification limit. In addition, any discrete configuration made adds overriding performance constraints to any associated supporting element. Thus, the anticipated third-generation equipment maximizes the probability of success and minimizes time for the total system rationale.

Me-Books (1972)

In 1972, Freeman Gosden Jr. started the Me-Books Publishing Company. It published books for kids. But if you bought a Me-Book for your child, you wouldn’t see in it the traditional names “Dick, Jane, and Sally”; instead, you’d see the name of your own child. To order the book, you had to tell the company the names of all your children, their friends, and pets. Their names appeared in the story.

The story was printed beautifully, in a 32-page hard-covered book with pictures in color. It cost just \$3.95.

You could choose from 4 stories: “My Friendly Giraffe”, “My Jungle Holiday”, “My Birthday Land Adventure”, and “My Special Christmas”.

For example, if you lived on Jottings Drive, and your daughter’s name was Shea, and her friend’s name was Douglas, the story “My Friendly Giraffe” included paragraphs such as:

One morning Shea was playing with Douglas in front of her home. When she looked up, what do you think she saw walking down the middle of Jottings Drive? You guessed it. A giraffe!

Ted Nelson, author of *Computer Lib*, played a trick. He ordered a copy of "My Friendly Giraffe", but pretended that his child's name was "Tricky Dick Nixon" who lived on "Pennsylvania Ave." in "Washington". Sure enough, the company sent him "My Friendly Giraffe: A Me-Book for Tricky Dick". Here are excerpts:

Once upon a time, in a place called Washington, there lived a little boy named Tricky Dick Nixon. Now, Tricky Dick wasn't just an ordinary little boy. He had adventures that other little boys and girls just dream of. This is the story of one of his adventures. It's the story of the day that Tricky Dick met a giraffe....

As the giraffe came closer and closer, Tricky Dick started to wonder how in the world he was going to look him in the eye....

Tricky Dick knew there were no jungles in Washington. Especially on Pennsylvania Ave. But Tricky Dick wasn't even a little bit worried. First, because he was a very brave little boy. And second, because he knew that his friend, the giraffe, would never take him anywhere bad....

Tricky Dick was home. Back in Washington. Back on Pennsylvania Ave. And with a story to tell his friends, that they wouldn't have believed if they hadn't seen Tricky Dick riding off on the giraffe's back. Tricky Dick would long be a hero to those who had seen him that day....

There would be many other exciting adventures for Tricky Dick and his friends. And maybe, just maybe, if you're a very good boy, someday we'll tell you about those, too.

Me-Books were commercially successful: many thousands of them were sold. Teachers liked the books, because the books encouraged students to read: students enjoyed reading books that contained their own names.

Porn (1973)

I programmed the computer to turn words into sentences. For example, Judith Ann Abrams fed the computer this word list....

Names: Barbara, Buster, Clyde, Colin, Debbie, Ed Shanahan, Jeff, Keith, Kerri, Liz, Miles, Nicholson 700, Norwalk

Other nouns: bastard, bikini, bitch, boat, cigarette, creep, dress, ethanol, eyelid, fly, gambling, jerk, knee, kneecap, lip, mouth, navel, piece, pornography, shirt, stud, tongue, virgin, whiskey sour, whore

Transitive verbs: bite, caress, castrate, climb, enjoy, kiss, knife, lick, pick, pull, rip screw, shake, take off, teach, undo

Other verbs: disapprove, fall down, fly, hesitate, sing, sleep, sleepwalk, sneeze, vanish, walk

Adjectives: bitchy, creepy, dead, enthusiastic, evil, gaudy, green, gross, horny, insane, mixed-up, monstrous, multicolored, neurotic, obscene, paranoid, prickly, profane, provocative, repulsive, sensuous, sexual, sexy, shiny, sick, stimulating, strange, succulent, tasty, unsettled, weird

A **transitive verb** is an action you can do to an object. For example, since you can *bite* and *caress* an object, *bite* and *caress* are transitive verbs.

Here's the beginning of what the computer printed:

A Sexy, Profane Creep's Tongue
by Judith Ann Abrams and the computer

The lip is tasty.

Ugh!

The mouth is enthusiastic. Does the gaudy bastard sleepwalk?

The tongue's whore caresses the bikini. When Nicholson 700 is screwing the creepy jerk over a piece, Jeff is teaching the gambling. Does paranoid Liz hesitate? Why does the tongue kiss the creep stimulatingly on horny Buster?

Undo! Pull the fly repulsively! A multicolored, enthusiastic cigarette is neurotic. Is a multicolored, weird dress's navel under the strange bikini? Why is the obscene dress's cigarette Colin's ethanol? Rats! Why is the dead, insane, tasty, shiny, knifing creep's bitch unsettled and strange?

Clyde is over Ed Shanahan. Kissing the bastard, a paranoid ethanol is unsettled. Why is the prickly kneecap's mouth on the creepy piece? Teaching Norwalk is sneezing paranoidly and grossly under provocative, shiny Debbie. As a sexy, walking tongue's virgin takes off the bitchy cigarette over the profane stud then castrates enthusiastic Miles's lip paranoidly, the dead,

gaudy knee is stimulating and multicolored. Even though the dead, insane piece licks strangely and neurotically in the flying, mixed-up eyelid's knee, the dead, biting, obscene bikini is on a repulsive mouth's gambling.

The pornography is gaudy. Kerri sleepwalks. Why is the tongue sensuous? Buster is sick. Is Miles monstrous? Debbie is neurotic and paranoid, when a stimulating fly picks the navel's jerk under Ed Shanahan. Why is the dress succulent? Hesitating, a kneecap sleeps and climbs the dead, bitchy ethanol. As insane Colin's bastard falls down weirdly in a sensuous dress, green, unsettled Miles's virgin is strange and sexual.

Is the creepy eyelid provocative? The gambling's whiskey sour teaches a navel.

The computer turned her adjectives into adverbs, by adding *ly* and making other changes. *Stimulating* became *stimulatingly*, and *gross* became *grossly*. Unfortunately, the computer's method wasn't perfect: the computer turned *neurotic* into *neurotically* (instead of *neurotically*).

It conjugated her verbs. *Screw* became *screwing*, and *bite* became *biting* (the computer dropped the *e*). *Lick* became *licks*, and *teach* became *teaches* (the computer added the *e* after the *ch*).

It added 's to her nouns. *Tongue* became *tongue's*. *Miles* became *Miles's* (it should have become *Miles'*).

For each sentence, the grammar is chosen randomly. The chance is 10% that the sentence will begin with an exclamation. If the sentence isn't merely an exclamation, the chance is 18% that the sentence will be a question.

If it's a question, there's a 40% chance it will begin with the word *why*. There's a 50% chance the question's main part will have the form *does...* noun phrase... verb phrase, and a 50% chance it will have this form instead: *is...* noun phrase... complement.

To construct a noun phrase from nouns, adjectives, etc., the computer uses random numbers. It uses random numbers to also construct verb phrases and complements.

The program uses a special variable, called W. At the beginning of the composition, W is near zero; but it tends to increase as the composition progresses. It affects the **complexity**. When W is large, the chance is large that the computer will print adjectives, adverbs, subordinate clauses, and correlative clauses. This sentence was produced by a small W:

The lip is tasty.

This sentence was produced by a large W:

As a sexy, walking tongue's virgin takes off the bitchy cigarette over the profane stud then castrates enthusiastic Miles's lip paranoidly, the dead, gaudy knee is stimulating and multicolored.

Poetic images (1973)

One of my students, Toby D'Oench, made the computer create poetic images, such as these:

TO GUINEVERE — LADY OF THE LAKE

Silent mists

Billow in creations

Windmills for flames evolve into ethers

Merlin again

MY MEMORY

Frozen children

Quiver with leaves

Creations with leaves hover over thoughts

Gardens of verse

A NEW ENGLAND BARN

Lazy fragrances

Waft by ethers

Seas on fragrances billow in sorrow

Rusted pitchforks

NEWPORT

Frozen sails

Slumber in fog

Hazes for sails waft by thoughts

Docks — yachts — luxuries of eras gone by

The program contains these lists:

<u>Adjectives</u>	<u>Prepositions</u>	<u>Verbs</u>
fleeting	of	billow in
crimson	on	glitter with
silent	under	flutter by
sensate	above	drift with
pliant	below	flow into
gloomy	in	ponder about
pallid	with	waft by
inky	by	quiver with
frozen	for	hover over
lazy	through	gleam like
		wander through
		slumber in
		dart by
		evolve into
		sing to

Title... noun... ending

TO REMBRANDT... windmills... A simple brush
 WAITING FOR THE PATIENT... ethers... Waiting
 THE PROPHET... visions... Then a word
 LISTERINE... breaths... Plastic society
 NEWPORT... sails... Docks — yachts — luxuries of eras gone by
 EXISTENCE... seas... In the beginning?
 SUMMER IN WATTS... flames... Tar-street neon — and the night
 TO GUINEVERE — LADY OF THE LAKE... mists... Merlin again
 NOON IN CALCUTTA... hazes... Emaciated dark forms strewn like garbage
 WEST HARBOR... fog... A solitary gull slices through
 A NEW ENGLAND BARN... fragrances... Rusted pitchforks
 A CHILD'S MICROSCOPE... creations... The wonderful amoeba
 A GROUP PORTRAIT... bundles... Christmas
 THE MILKY WAY... cosmos... A gooey mess
 TOMBSTONE... sorrow... Rubbings
 LIFE AT THE END OF A BRANCH... leaves... Swirling to the ground
 SEASHELLS AND THINGS... waves... Dribble-dribble-dribble castle
 A BEAVER POND... reeds... Thwack
 MY MEMORY... children... Gardens of verse
 EINSTEIN... thoughts... Somehow through this — an understanding of a superior order

To create a poetic image, the computer fills in this form:

<u>TITLE</u>				
<u>Adjective</u>	<u>Noun that goes with the title</u>			
<u>verb</u>	<u>Noun</u>			
<u>Noun</u>	<u>Preposition</u>	<u>Noun</u>	<u>verb</u>	<u>Noun</u>
<u>Ending that goes with the title</u>				

Analyze writing

The computer can analyze what humans write.

English poetry

Can the computer analyze English poetry? From 1957 to 1959 at Cornell University, Stephen Parrish made the computer alphabetize the words in Matthew Arnold's poetry. Here's an excerpt:

	Page in book	Poem's title	Line in poem
CONSCIOUS			
back with the conscious thrill of shame	181	Isolation Marg	19
conscious or not of the past	287	Rugby Chapel	45
CONSCIOUSNESS			
the last spark of man's consciousness with words	429	Empedocles II	30
and keep us prisoners of our consciousness	439	Empedocles II	352
CONSECRATE			
Peter his friend with light did consecrate	445	Westmin Abbey	50
CONSECRATES			
which consecrates the ties of blood for these indeed	196	Frag Antigone	31

To find out what Matthew Arnold said about love, just look up

LOVE. Such an index is called a **concordance**.

That concordance was the first produced by a computer. Previously, all concordances of poetry were created by hand, using filing cards. For example, in 1870 a group of researchers began creating a concordance to Chaucer, by hand. They started at the letter A. 45 years later, they were only up to the letter H!

Did the poet Shelley steal ideas from others? Joseph Raben, at Queens College, believed Shelley borrowed imagery from Milton. To prove it, in 1964 he made the computer produce concordances to Shelley's *Prometheus Unbound* and Milton's *Paradise Lost* and compare them. The computer found many similarities between Shelley and Milton.

What were Shakespeare's favorite words? In 1971 at Münster University in Germany, Marvin Spevack fed the computer all the works of Shakespeare, and made it count how often each word occurs. Disregarding trivial words such as *a* and *the*, the computer discovered Shakespeare's favorite word was *love*: he used it 2,271 times. Next come *heart*, *death*, *man*, *life*, and *hand*. He never used the word *hero*. In *Macbeth*, the word *good* occurs more often than any other adjective, noun, or adverb, and more often than most verbs.

By counting words, other researchers made the computer graph the rise and fall of themes in a novel.

American history

Who wrote the *Federalist Papers*? Historians knew some of the papers were by Alexander Hamilton and others by James Madison, but the authorship of the remaining papers was in dispute.

In 1964, Mosteller and Wallace made the computer compare the literary styles of the papers, by counting the frequency of words such as *by*, *enough*, *from*, *to*, *upon*, *while*, and *whilst*. It concluded that all the disputed papers were written by Madison, not Hamilton.

The statistical evidence was so high that historians accept the computer's finding as fact.

The Bible

Can the computer analyze the Bible? In 1951, Texas clergyman John Ellison made the computer compare 309 Greek manuscripts of the New Testament. Underneath each word of a standard text, the computer printed the variants found in other manuscripts. It classified the manuscripts according to their similarities.

In 1957, he published a concordance to the Revised Standard Bible, and a pair of other researchers (Tasman & Busa) indexed the Dead Sea Scrolls.

Did the apostle Paul really write all those marvelous letters attributed to him in the New Testament? Or were they actually written by somebody else?

In 1964, Scottish clergyman Andrew Morton used the computer to deduce that Paul didn't write some of those letters.

All Morton did was count how often Paul used the Greek word *kai* in each sentence. *Kai* means *and*. Coming to a conclusion about Biblical authorship by counting just the word *and* might seem silly, but Morton said he analyzed 20 writers of ancient Greek and found each used *kai* with a constant frequency. In the "Pauline" letters, the frequency of *kai* varied a lot, implying some of them were not by Paul.

Ellison distrusted Morton's assumption that a man's literary style must remain constant. He warned: if Morton's method were applied to the Declaration of Independence and Thomas Jefferson's letters to his wife, the computer might conclude that either Jefferson didn't write the Declaration of Independence or another man was writing love letters to Mrs. Jefferson. In 1965, to prove his point, he applied Morton's method to 2 of Morton's own articles on the subject: the computer concluded that Morton couldn't be the author of both!

Forgery

IBM programmed the computer to detect a forged signature — even if the signature looks correct to the naked eye.

To use the IBM forgery-detection system, write your signature by using IBM's special pen, attached to the computer. As you write, the computer notices how hard you press the pen against the paper and how fast you move the pen.

If somebody else tries to pretend he's you, he must sit down at the machine and try to duplicate your signature. If he presses the pen hardest at different points of the signature, or if he accelerates the pen's motion at different points, the computer says he's a fake.

The system works well, because the average crook trying to forge your signature will hesitate at the hard parts. His hesitation affects the pen's pressure and acceleration, which tell the computer he's faking.

IBM developed the system in 1979 but didn't start selling it until many years later. Now IBM sells an improved version. Remember: the system works just on signatures written with IBM's pen.

Artificial intelligence

You have what's called **natural intelligence** (except when your friends accuse you of having "natural stupidity"). A computer's intelligence, by contrast, is **artificial**. Can the computer's **artificial intelligence (AI)** ever match yours?

For example, can the computer ever develop the "common sense" needed to handle exceptions, such as a broken traffic light? After waiting at a red light for several hours, the typical human would realize the light was broken. The human would try to proceed past the intersection, cautiously. Would a computer programmed to "never go on red" be that smart?

Researchers who study the field of artificial intelligence have invented robots and many other fascinating computerized devices. They've also been trying to develop computers that can understand ordinary English commands and questions, so you won't have to learn a "programming language". They've been trying to develop **expert systems** — computers that imitate human experts such as doctors and lawyers.

Early dreamers

The dream of making a computer imitate us began many centuries ago....

The Greeks The hope of making an inanimate object act like a person can be traced back to the ancient Greeks. According to Greek mythology, Pygmalion sculpted a statue of a woman, fell in love with it, and prayed to the gods to make it come to life. His wish was granted — she came to life. And they lived happily ever after.

Ramon Lull (1272 A.D.) In 1272 A.D. on the Spanish island of Majorca, Ramon Lull invented the idea of a machine that would produce *all* knowledge, by putting together words at random. He even tried to build it.

Needless to say, he was a bit of a nut. Here's a description of his personality (written by Jerry Rosenberg, abridged):

Ramon Lull married young and fathered two children — which didn't stop him from his courtier's adventures. He had an especially strong passion for married women. One day as he was riding his horse down the center of town, he saw a familiar woman entering church for a High Mass. Undisturbed by this circumstance, he galloped his horse into the cathedral and was quickly thrown out by the congregants. The lady was so disturbed by his scene that

she prepared a plan to end Lull's pursuit once and for all. She invited him to her boudoir, displayed the bosom that he had been praising in poems written for her, and showed him a cancerous breast. "See, Ramon," she said, "the foulness of this body that has won thy affection! How much better hadst thou done to have set thy love on Jesus Christ, of Whom thou mayest have a prize that is eternal!"

In shame Lull withdrew from court life. On four different occasions a vision of Christ hanging on the Cross came to him, and in penitence Lull became a dedicated Christian. His conversion was followed by a pathetic impulse to try to convert the entire Moslem world to Christianity. This obsession dominated the remainder of his life. His "Book of Contemplation" was divided into 5 books in honor of the 5 wounds of Christ. It contained 40 subdivisions — for the 40 days that Christ spent in the wilderness; 366 chapters — one to be read each day and the last chapter to be read only in a leap year. Each chapter had 10 paragraphs to commemorate the 10 commandments; each paragraph had 3 parts to signify the trinity — for a total of 30 parts a chapter, signifying the 30 pieces of silver.

In his book's final chapter, he tried to prove to infidels that Christianity was the only true faith.

Several centuries later — in 1726 — Lull's machine was pooh-poohed by Jonathan Swift, in *Gulliver's Travels*.

Gulliver meets a professor who built such a machine. The professor claims his machine lets "the most ignorant person... write books in philosophy, poetry, politics, law, mathematics, and theology without the least assistance from genius and study."

The machine is huge — 20 feet on each side — and contains all the words of the language, in all their declensions, written on paper scraps glued onto bits of wood connected by wires.

Each of the professor's 40 students operates one of the machine's 40 cranks. At a given signal, every student turns his crank a random distance, to push the words into new positions.

Gulliver says:

He then commanded 36 of the lads to read the several lines softly as they appeared upon the frame. Where they found three or four words together that might make part of a sentence, they dictated to the 4 remaining boys, who were scribes. Six hours a day the young students were employed in this labor. The professor showed me several large volumes already collected, of broken sentences, which he intended to piece together, and out of those rich materials give the world a complete body of all arts and sciences.

Karel Capek (1920) The word **robot** was invented in 1920 by Karel Capek, a Czech playwright. His play "R.U.R." shows a factory where the workers look human but are really machines. The workers are dubbed *robots*, because the Czech word for *slave* is *robotnik*.

His play is pessimistic. The invention of robots causes unemployment. Men lose all ambition — even the ambition to raise children. The robots are used in war, go mad, revolt against mankind and destroy it. In the end only two robots are left. It's up to them to repopulate the world.

Isaac Asimov (1942) Many sci-fi writers copied Capek's idea of robots, with even more pessimism. An exception was Isaac Asimov, who depicted robots as being loving. He coined the word **robotics**, which means the study of robots, and in 1942 developed what he calls the "**3 Laws of Robotics**". Here's the version he published in 1950:

1. A robot may not injure a human being or, through inaction, allow a human being to come to harm.
2. A robot must obey the orders given it by human beings, except where such orders would conflict with the First Law.
3. A robot must protect its own existence, as long as such protection does not conflict with either the First or the Second Law.

Norbert Wiener (1947) The word **cybernetics** was invented in 1947 by Norbert Wiener, an MIT professor. He defined it to be "the science of control and communication in the animal and the machine." Wiener and his disciples, who called themselves **cyberneticists**, wondered whether it would be possible to make an electrical imitation of the human nervous

system. It would be a “thinking machine”. They created the concept of **feedback**: animals and machines both need to perceive the consequences of their actions, to learn how to improve themselves. For example, a machine that is producing parts in a factory should examine the parts it has produced, the heat it has generated, and other factors, to adjust itself accordingly.

Like Ramon Lull, Wiener was strange. He graduated from Tufts College when he was 14 years old, got his doctorate from Harvard when he was 18, and became the typical “absent-minded professor”. These anecdotes are told about him:

He went to a conference and parked his car in the big lot. When the conference was over, he went to the lot but forgot where he parked his car. He even forgot what his car looked like. So he waited until all the other cars were driven away, then took the car that was left.

When he and his family moved to a new house a few blocks away, his wife gave him written directions on how to reach it, since she knew he was absent-minded. But when he was leaving his office at the end of the day, he couldn’t remember where he put her note, and he couldn’t remember where the new house was. So he drove to his old neighborhood instead. He saw a young child and asked her, “Little girl, can you tell me where the Wieners moved?” “Yes, Daddy,” came the reply, “Mommy said you’d probably be here, so she sent me to show you the way home.”

One day he was sitting in the campus lounge, intensely studying a paper on the table. Several times he’d get up, pace a bit, then return to the paper. Everyone was impressed by the enormous mental effort reflected on his face. Once again he rose from his paper, took some rapid steps around the room, and collided with a student. The student said, “Good afternoon, Professor Wiener.” Wiener stopped, stared, clapped a hand to his forehead, said “Wiener — that’s the word,” and ran back to the table to fill the word “wiener” in the crossword puzzle he was working on.

He drove 150 miles to a math conference at Yale University. When the conference was over, he forgot he came by car, so he returned home by bus. The next morning, he went out to his garage to get his car, discovered it was missing, and complained to the police that while he was away, someone stole his car.

Those anecdotes were collected by Howard Eves, a math historian.

Alan Turing (1950) Can a computer “think”? In 1950, Alan Turing proposed the following test, now known as the **Turing test**:

In one room, put a human and a computer. In another room, put another human (called the Interrogator) and give him two terminals — one for communication with the computer, and the other for communication with the other human — but don’t tell the Interrogator which terminal is which. If he can’t tell the difference, the computer’s doing a good job of imitating the human, and (according to Turing) we should say the computer can “think”.

Turing called it the **Imitation Game**. The Interrogator asks questions. The human witness answers honestly. The computer pretends to be human.

To win that game, the computer must be able to imitate human weaknesses as well as strengths. For example, when asked to add two numbers, it should pause before answering, as a human would. When asked to write a sonnet, a good imitation-human answer would be, “Count me out on this one. I never could write poetry.” When asked “Are you human”, the computer should say “yes”.

Such responses wouldn’t be hard to program. But a clever Interrogator could give the computer a rough time, by requiring it to analyze its own thinking:

Interrogator: In the first line of your sonnet which reads “Shall I compare thee to a summer’s day,” wouldn’t “a spring day” do as well or better?

Witness: It wouldn’t scan.

Interrogator: How about “a winter’s day”? That would scan all right.

Witness: Yes, but nobody wants to be compared to a winter’s day.

Interrogator: Would you say Mr. Pickwick reminded you of Christmas?

Witness: In a way.

Interrogator: Yet Christmas is a winter’s day, and I don’t think Mr. Pickwick would mind the comparison.

Witness: I don’t think you’re serious. By “a winter’s day” one means a typical winter’s day, rather than a special one like Christmas.

If the computer could answer questions that well, the Interrogator would have a hard time telling it wasn’t human.

Donald Fink has recommended the Interrogator say, “Suggest an unsolved problem and some methods for working toward its solution,” and “What methods would most likely prove fruitful in solving the following problem....”

Turing believed computers would someday be able to win the game and therefore be considered to “think”. In his article, he listed 9 possible objections to his belief and rebutted them:

1. Soul Thinking’s a function of man’s immortal soul. Since computers don’t have souls, computers can’t think. **Rebuttal:** since God’s all-powerful, He can give computers souls if He wishes. Just as we create children to house His souls, so should we serve Him by creating computers.

2. Dreadful If machines could equal us in thinking, that would be dreadful! **Rebuttal:** too bad!

3. Logicians Logicians have proved it’s impossible to build a computer that can answer every question. **Rebuttal:** is it possible to find a *human* that can answer every question? Computers are no dumber than we. Though no one can answer every question, why not build a succession of computers, each one more powerful than the next, so every question could be answered by at least one of them?

4. Conscious Though computers can produce, they can’t be *conscious* of what they’ve produced. They can’t feel pleasure at their successes, misery at their mistakes, and depression when they don’t get what they want. **Rebuttal:** the only way to be sure whether a computer has feelings is to become one. A more practical experiment would be to build a computer that explains step-by-step its reasoning, motivations, and obstacles it’s trying to overcome, and also analyzes emotional passages such as poetry. Such a computer’s clearly not just parroting.

5. Human A computer can’t be kind, resourceful, beautiful, friendly, have initiative, have a sense of humor, tell right from wrong, make mistakes, fall in love, enjoy strawberries & cream, make someone fall in love with it, learn from experience, use words properly, be the subject of its own thought, have as diverse behavior as a man, or do something really new. **Rebuttal:** why not? Though such a computer hasn’t been built yet, it might be possible in the future.

6. Surprise The computer never does anything original or surprising. It does only what it’s told. **Rebuttal:** how do you know “original” human work isn’t just grown from a seed (implanted by teaching) or the effect of well-known general principles? And who says computers aren’t surprising? The computer’s correct answers are often surprisingly different from a human’s rough guesses.

7. Binary Nerve cells can sense gradual increases in electrical activity — you can feel a “little tingle” or a “mild pain” or an “ouch” — whereas a computer’s logic is just binary — either a “yes” or “no”. **Rebuttal:** by using techniques such as “random numbers”, you can make the computer imitate the flexible, probabilistic behavior of the nervous system enough so the Interrogator can’t tell the difference.

8. Rules Life can’t be reduced to rules. For example, if a traffic-light rule says “stop when the light is red, and go when the light is green”, what do you do when the light is broken, and both the red and green appear simultaneously? Maybe you should have an extra rule saying in that case to stop. But some further difficulty may arise with that rule, and you’d have to create another rule. And so on. You can’t invent enough rules to handle all cases. Since computers must be fed rules, they can’t handle all of life. **Rebuttal:** though life’s more than a simple set of rules, it might be the *consequences* of simple psychological laws of behavior, which the computer could be taught.

9. ESP Humans have extrasensory perception (ESP), and computers don’t. **Rebuttal:** maybe the computer’s random-number generator could be hooked up to be affected by ESP. Or to prevent ESP from affecting the Imitation Game, put both the human witness and the computer in a telepathy-proof room.

To make the computer an intelligent creature, Turing suggested two possible ways to begin. One way would be to teach the computer abstract skills, such as chess. The other way would be to give the computer eyes, ears, and other sense organs, teach it how to speak English, then educate it the same way you’d educate a somewhat handicapped child.

4 years later — on June 8, 1954 — Turing was found dead in

bed. The police say he died from potassium cyanide, self-administered. He'd been plating spoons with potassium cyanide in electrolysis experiments. His mother refuses to believe it was suicide, and hopes it was just an accident.

Understanding English

It's hard to make the computer understand plain English!

Confusion Suppose you feed the computer this famous saying:

Time flies like an arrow.

The computer might interpret that saying in 3 ways:

Interpretation 1 The computer thinks "time" is a noun, so the sentence means "The time can fly by as quickly as an arrow flies."

Interpretation 2 The computer thinks "time" is a verb, so the sentence means "Time the speed of flies like you'd time the speed of an arrow."

Interpretation 3 The computer thinks "time" is an adjective, so the sentence means "There's a special kind of insect, called a 'time fly', and those flies are attracted to an arrow (in the same way moths are attracted to a flame)."

Suppose a guy sits on a barstool and shares his drinks with a tall woman while they play poker for cash. If the woman says to him, "Up yours!", the computer might interpret it 8 ways:

The woman is upset at what the man did.
The woman wants the man to raise up his glass, for a toast.
The woman wants the man to up the ante and raise his bet.
The woman wants the man to hold his cards higher, so she doesn't see them.
The woman wants the man to pick up the card she dealt him.
The woman wants the man to raise his stool, so she can see him eye-to-eye.
The woman wants the man to pull up his pants.
The woman wants the man to have an erection.

For another example, suppose Mae West were to meet a human-looking robot and ask him:

Is that a pistol in your pocket, or are you glad to see me?

The robot would probably analyze that sentence too logically, then reply naively:

There is no pistol in my pocket, and I am glad to see you.

In spite of those confusions, programmers have tried to make the computer understand English. Here are some famous attempts...

Baseball (1961) In 1961 at MIT, programmers made the computer answer questions about baseball.

In the computer's memory, they stored the month, day, place, teams, and scores of each game in the American League for one year. They programmed the computer so that *you can type your question in ordinary English*. The computer analyzes your question's grammar and prints the correct answer.

Here are examples of questions the computer can analyze and answer correctly:

Who did the Red Sox lose to on July 5?
Who beat the Yankees on July 4?
How many games did the Yankees play in July?
Where did each team play in July?
In how many places did each team play in July?
Did every team play at least once in each park in each month?

To get an answer, the computer turns your questions into equations:

Question	Equations
Where did the Red Sox play on July 7?	place = ? team = Red Sox month = July day = 7
What teams won 10 games in July?	team (winning) = ? game (number of) = 10 month = July
On how many days in July did eight teams play?	day (number of) = ? month = July team (number of) = 8

To do that, the computer uses this table:

Word in your question	Equation
where	place = ?
Red Sox	team = Red Sox
July	month = July
who	team = ?
team	team =

The computer ignores words such as *the*, *did*, and *play*.

If your question mentions *Boston*, you might mean either "place = Boston" or "team = Red Sox". The computer analyzes your question to determine which equation to form.

After forming the equations, the computer hunts through its memory, to find the games that solve the equations. If an equation says "number of", the computer counts. If an equation says "winning", the computer compares the scores of opposing teams.

The programmers were Bert Green, Alice Wolf, Carol Chomsky, and Kenneth Laughery.

What's a story problem? When you were in school, your teacher told you a story that ended with a mathematical question. For example:

Dick had 5 apples. He ate 3. How many are left?

In that problem, the last word is: *left*. That means: subtract. So the correct answer is 5 minus 3, which is 2.

Can the computer solve problems like that? Here's the most famous attempt...

Arithmetic & algebra (1964) MIT awarded a Ph.D. to Daniel Bobrow for making the computer solve story problems involving arithmetic and algebra.

Let's see how the computer solves this problem:

If the number of customers Tom gets is twice the square of 20 percent of the number of advertisements he runs, and the number of advertisements he runs is 45, what is the number of customers Tom gets?

To begin, the computer replaces *twice* by *2 times*, and replaces *square of* by *square*.

Then the computer splits the sentence into shorter ones:

The number of customers Tom gets is 2 times the square 20 percent of the number of advertisements he runs. The number of advertisements he runs is 45. What is the number of customers Tom gets?

The computer turns each sentence into an equation:

number of customers Tom gets = 2 * (.20 * number of advertisements he runs)²
number of advertisements he runs = 45
X = number of customers Tom gets

The computer solves the equations and prints the answer as a complete sentence:

The number of customers Tom gets is 162.

Here's a harder problem:

The sum of Lois's share of some money and Bob's share is \$4.50. Lois's share is twice Bob's. Find Bob's and Lois's share.

Applying the same method, the computer turns the problem into these equations:

Lois's share of some money + Bob's share = 4.50 dollars
Lois's share = 2 * Bob's
X = Bob's
Y = Lois's share

The computer tries to solve the equations but fails. So it assumes "Lois's share" is the same as "Lois's share of some money", and "Bob's" is the same as "Bob's share". Now it has 6 equations:

Original equations

Lois's share of some money + Bob's share = 4.50 dollars

Lois's share = 2 * Bob's

X = Bob's

Y = Lois's share

Assumptions

Lois's share = Lois's share of some money

Bob's = Bob's share

It solves them and prints:

Bob's is 1.50 dollars.

Lois's share is 3 dollars.

The computer can solve problems about distance:

The distance from New York to Los Angeles is 3000 miles. If the average speed of a jet plane is 600 miles per hour, find the time it takes to travel from New York to Los Angeles by jet.

The resulting equations are:

distance from New York to Los Angeles = 3000 * miles

average speed of jet plane = (600 * miles)/(1 * hours)

X = time it takes to travel from New York to Los Angeles by jet

The computer is unable to solve them. But in its memory it finds the formula "distance = speed * time". It winds up with 7 equations:

Original equations

distance from New York to Los Angeles = 3000 * miles

average speed of jet plane = (600 * miles)/(1 * hours)

X = time it takes to travel from New York to Los Angeles by jet

Formula

distance = speed * time

Assumptions

distance = distance from New York to Los Angeles

speed = average speed of jet plane

time = time it takes to travel from New York to Los Angeles by jet

It solves them and prints:

The time it takes to travel from New York to Los Angeles by jet is 5 hours.

The computer can solve problems about age:

Mary is twice as old as Ann was when Mary was as old as Ann is now. If Mary is 24 years old, how old is Ann?

To begin, the computer replaces *twice* by *2 times*, and replaces *how old* by *what*. When the computer sees the words *as old as*, it knows the problem has to do with ages, so it breaks the problem into these sentences:

Mary's age is 2 times Ann's age X years ago. X years ago Mary's age is Ann's age now. Mary's age is 24. What is Ann's age?

The resulting equations are:

Mary's age = 2 * (Ann's age - X)

Mary's age - X = Ann's age

Mary's age = 24

Y = Ann's age

The computer prints:

Ann's age is 18.

If a situation appears hopeless, the computer asks you for help. In the following conversation, the human poses a problem, and the computer begs for a hint:

Human: The gross weight of a ship is 20000 tons. If its net weight is 15000 tons, what is the weight of the ship's cargo?

Comp.: Do you know any more relationships among these variables: gross weight of ship, tons, its net weight, weight of ship's cargo?

Human: Yes.

Comp.: Tell me.

Human: The weight of a ship's cargo is the difference between the gross weight and the net weight.

Comp.: The weight of the ship's cargo is 5000 tons.

The program has some limitations:

It cannot solve quadratic equations. If the computer tries to apply the formula "distance = speed * time" to a problem involving *two* distances, it forgets which distance is which.

The computer's vocabulary is limited. It doesn't realize that *how many* means *what is the number of*, and *how far* is means *what is the distance to*. For problems about age, the characters must be named Mary, Ann, Bill, Father, or Uncle, unless you diddle with the computer's memory.

If the human types *Tom has 2 apples, 3 bananas, and 4 pears*, the comma before the *and* makes the computer split the sentence into these two wrong "sentences"....

Tom has 2 apples, 3 bananas.

4 pears.

If the human mentions *the number of times John went to the movies*, the computer thinks *times* means multiplication, and tries to multiply *number of by John went to the movies*.

Encyclopedia (1964-1966)

In 1964, Simmons, Klein, and McConlogue (at the Systems Development Corporation) fed a child's encyclopedia into a computer.

If you type *What do worms eat?* the computer hunts through the encyclopedia, to find sentences mentioning both *worms* and *eat*. (To hasten the hunt, it uses an index produced by another program.) It finds two sentences:

Birds eat worms on the grass.

Most worms usually eat grass.

After analyzing the grammar of your question and those sentences, the computer realizes the first sentence is irrelevant, and prints just the correct answer:

Most worms usually eat grass.

In 1965, the program's abilities were extended, so that if you type *What animals live longer than men?* the computer searches through the encyclopedia for related words:

Word in sentence Related words searched

animals

mammals, reptiles, fish

live

age

longer

older, ancient

men

person, people, women

Here's an example of the program's operation in 1966. The human creates some encyclopedia material about horses, then retrieves it:

Human: Enter horse=noun, is=isverb, animal=noun, live=verb, on=prep, farm=noun, strong=adj, pull=verb, wagon=noun, most=adj, are=isverb, eat=verb, hay=noun, all=adj, red=adj, many=adj, have=haveverb, long=adj, mane=noun, some=adj, person=noun, race=verb.

Comp.: Entered in dictionary.

Human: Plurals horse=horses, animal=animals, farm=farms, wagon=wagons, mane=manes, person=people.

Comp.: Ok.

Human: Start.

Comp.: Ready.

Human: A horse is an animal. Horses live on a farm. Strong horses pull wagons. Most horses are strong. Horses eat hay. All horses are red. Many horses have long manes. Some people race horses.

Comp.: Learned.

Human: Forget "all horses are red".

Comp.: Forgotten.

Human: What is a horse?

Comp.: Horse is an animal.

Human: What do horses eat?

Comp.: Horses eat hay.

Human: What do horses do?

Comp.: Horses live on a farm and eat hay. Strong horses pull wagons.

Human: What do strong horses do?

Comp.: Strong horses pull wagons.

Human: Do many horses have long manes?

Comp.: Yes.

Human: Do all horses have long manes?

Comp.: To my knowledge, no.

Human: Describe horses?

Comp.: Horses are an animal. Most horses are strong. Many horses have long manes. Horses live on a farm and eat hay. Strong horses pull wagons. Some people race horses.

When the computer's answer consists of several sentences, it prints the most relevant sentence first, the others in decreasing order of relevance.

Military (1966) Thompson and Craig (at General Electric) made the computer answer military questions such as:

Has the 25th battalion arrived in Texas since 3 P.M.?

Is the 100th scheduled to arrive at Ft. Lewis before the 200th leaves Ft. Lewis?

Eliza (1966-1967) In 1966 at MIT, Joseph Weizenbaum wrote the Eliza program, which makes the computer act as a psychotherapist.

In 1967, he changed the program, to make it handle keywords such as *also, call, compute, define, is, let, of, thanks, where*, and the symbol =.

In the following conversation, the human teaches the computer how to find the area of a circle, a sphere, and a square, and then asks the computer questions about geometry. The human also teaches the computer how to speak German —

English German

I say	ich sage
I mean	meine ich
what is	wieviel ist
squared	quadrat

and gradually starts speaking to the computer in German.

Human: The area of a circle is pi times the radius of the circle squared. The area of a sphere is pi times the diameter of the sphere squared. The area of a square is the side of the square squared. The diameter of an object is twice the radius of the object. The radius of a ball is 10. What is the area of the ball?

Comp.: The assumption that ball is circle is consistent — shall I go on?

Human: No.

Comp.: The assumption that ball is sphere and object is consistent — shall I go on?

Human: Yes.

Comp.: It's 1256.6359.

Human: What is the diameter of the ball?

Comp.: It's 20.

Human: The radius of a dish is 10. What's the area of the dish?

Comp.: The assumption that dish is circle is consistent — shall I go on?

Human: Yes.

Comp.: It's 314.15899.

Human: The radius of a globe is 10. A globe is a sphere. A sphere is an object. What is the area of the globe?

Comp.: It's 1256.6359.

Human: The side of a plate is 10. What is the area of the plate?

Comp.: The assumption that plate is square is consistent — shall I go on?

Human: Yes.

Comp.: It's 100.

Human: When I say board I mean plate. What is 3 times the side of the board?

Comp.: It's 30.

Human: Please call that x. Let f(z) be z squared minus z. Compute f(x).

Comp.: It's 870.

Human: When I say ich sage I mean I say. When ich sage meine ich I mean I mean. Wenn ich sage wieviel ist meine ich what is. Wenn ich sage quadrat meine ich squared. Nun, sagen sie mir bitte, wieviel ist x quadrat.

Comp.: I didn't understand you.

Human: Ja, tut mir leid, ich sollte fragen wieviel IST x quadrat.

Comp.: It's 900.

Calculus (1969) At MIT, Charniak made the computer solve story problems involving calculus.

Human: Water is flowing into a conical filter at the rate of 15 cubic inches per second. If the radius of the base of the filter is 5 inches and the altitude is 10 inches, find the rate at which the water level is rising when the volume is 100 cubic inches.

Comp.: The answer is .53132943 * in * sec^-1 * pi^~.33333332.

Probability (1971) Rensselaer Polytechnic Institute awarded a Ph.D. to J.P. Gelb, for making the computer solve story problems involving probability.

Human: From a zorch containing 4 ferd and 3 brakky and 5 chartreuse werfels, 3 are drawn. What is the probability that 2 are chartreuse and the other brakky?

Comp.: Replacement involved?

Human: No.

Comp.: 3/22 (or .1363636).

Surveying the field

The field of "artificial intelligence" includes many categories.

For example, it includes attempts to make the computer win at chess and checkers, understand English, and create its own original art and music. It also includes attempts to imitate human feelings, personal interactions, and therapists. I explained those topics earlier.

Protocol method During the 1950's and 1960's, most research in artificial intelligence was done at the Massachusetts Institute of Technology (MIT) and the Carnegie Institute of Technology (CIT, now called Carnegie-Mellon University). At Carnegie, the big names were Allen Newell and Herbert Simon. They invented the **protocol method**. In the protocol method, a human is told to solve a tough problem and, while he's solving it, to say at each moment what he's thinking. A transcript of his train of thought is recorded and called the **protocol**. Then programmers try to make the computer imitate that train of thought.

Using the protocol method, Newell and Simon produced programs that could "think like humans". The thinking, like human thinking, was imperfect. Their research did *not* try to make the computer a perfect thinker; instead, it tried to gain insight into how *humans* think. Their point of view was: if you think you really understand human psychology, go try to program it. Their attempt to reduce human psychology to computer programs is called **mentalism** and has replaced Skinner's stimulus-response behaviorism as the dominant force in psychology now.

Abstract math Many programmers have tried to make the computer do abstract math.

In 1957 Newell, Simon, and Shaw used the protocol method to make the computer prove theorems about symbolic logic, such as "Not (p or q) implies not p". In 1959 and 1960, Herbert Gelernter and his friends made the computer prove theorems about Euclidean geometry, such as "If the segment joining the midpoints of the diagonals of a trapezoid is extended to intersect a side of the trapezoid, it bisects that side."

In 1961, MIT awarded a Ph.D. to James Slagle for making the computer compute indefinite integrals, such as:

$$\int \frac{x^4}{(1-x^2)^{5/2}} dx$$

The computer gets the answer, which is:

$$\arcsin x + \frac{\tan^3 \arcsin x}{3} - \tan \arcsin x + c$$

Each of those programs works by drawing a tree inside the computer's memory. Each branch of the tree represents a possible line of attack. The computer considers each branch and chooses the one that looks most promising.

A better symbolic-logic program was written by Hao Wang in 1960. His program doesn't need trees; it always picks the right attack immediately. It's guaranteed to prove any theorem you hand it, whereas the program by Newell, Simon, and Shaw got stuck on some hard ones.

A better indefinite integration program was written by Joel Moses in 1967 and further improved in 1969. It uses trees very rarely and solves almost any integration problem.

A program that usually finds the right answer but might fail on hard problems is called **heuristic**. A heuristic program usually involves trees. The checkers, chess, and geometry programs are heuristic. A program that's guaranteed to always give the correct answer is called **algorithmic**. The original symbolic-logic program was heuristic, but Wang's improvement is algorithmic; Moses's indefinite integration program is almost algorithmic.

GPS In 1957 Newell, Simon, and Shaw began writing a single program to solve *all* problems. They called it **General Problem Solver (GPS)**. If you feed that program a goal, a list of operators, and associated information, the program will tell you how to achieve the goal by using the operators.

For example, suppose you want the computer to solve this simple problem: a monkey would like to eat some bananas that are too high for him to reach, but there's a box nearby he can stand on. How can he get the bananas?

Feed the GPS program this information...

Now:	monkey's place = place#1; box's place = place#2; contents of monkey's hand = empty
Want:	contents of monkey's hand = the bananas
Difficulties:	contents of monkey's hand is harder to change than box's place, which is harder to change than monkey's place
Allowable operator	Definition
climb box	before: monkey's place = box's place after: monkey's place = on the box
walk to x	after: monkey's place = x
move box to x	before: monkey's place = box's place after: monkey's place = x; box's place = x
get bananas	before: box's place = under the bananas; monkey's place = on the box after: contents of monkey's hand = the bananas

GPS will print the solution:

walk to place#2
move box to under the bananas
climb box
get bananas

The GPS approach to solving problems is called **means-ends analysis**: you tell the program the means (operators) and the end (goal). The program has proved theorems in symbolic logic, computed indefinite integrals, and solved many famous puzzles, such as "The Missionaries and the Cannibals", "The Tower of Hanoi", and "The 5-Gallon Jug and the 8-Gallon Jug". But the program works slowly and must be fed lots of info about the problem. The project was abandoned in 1967.

Vision Another large topic in artificial intelligence is **computer vision**: making the computer see.

The first vision problem tackled was **pattern recognition**: making the computer read handwritten printed letters. The problem is hard, because some people make their letters very tall or wide or slanted or curled or close together, and the pen may skip. Reasonably successful programs were written, although computers still can't tackle script.

Interest later shifted to **picture processing**: given a photograph of an object, make the computer tell what the object is. The problem is hard, because the photo may be taken from an unusual angle and be blurred, and because the computer gets confused by shadows.

Scene analysis is even harder: given a picture of a group of objects, make the computer tell which object is which. The problem is hard, because some of the objects may be partly hidden behind others, and because a line can have two different interpretations: it can be a crease in one object, or a dividing-line between two objects.

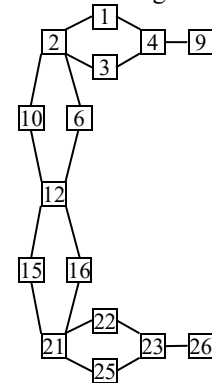
Most research in picture processing and scene analysis was done from 1968 to 1972.

Ray Kurzweil invented an amazing machine whose camera looks at a book and reads the book, by using a voice synthesizer. Many blind people have used it.

Robots Researchers have built robots. The first robots were just for experimental fun, but today's robots are truly useful: for example, robots build cars. Many young kids have been taught "LOGO", which is a language developed at the MIT Artificial Intelligence Lab that makes the computer control a robot turtle.

Today's research Now research in artificial intelligence is done at four major universities: MIT, Carnegie, Stanford, and Edinburgh (Scotland).

Reflexive control In the Soviet Union, weird researchers have studied **reflexive control**: they programmed the computer to be disobedient. The first such programmer was Lefevr, in 1967. In 1969 Baranov and Trudolyubov extended his work, by making the computer win this disobedience game:



The human begins by choosing either node 9 or node 26, *but doesn't tell the computer which node he's chosen*.

The computer starts at node 12; on each turn, it moves to an adjacent node. When it reaches either node 9 or node 26, the game ends: if the node the computer reaches is one of the human chose, the human wins; if the computer reaches the opposite node, the computer wins.

Before each move, the human tells the computer where to go; but the computer may decide to do the opposite (disobey).

What strategy should the computer use? If it always obeys or always disobeys, the human will catch on and make it lose.

Instead, Baranov and Trudolyubov programmed the computer to react as follows:

obey the human twice, then disobey three times, then obey once, disobey thrice, obey once, disobey twice, obey thrice, disobey once, obey thrice, disobey once,...

The irregular alternation of obedience and disobedience confuses the human in a way that works to the computer's advantage. Using that strategy, the computer played against 61 humans, and won against 44 of them (72%). In other words, the typical human tried to mislead the computer but in fact "clued it in" to the human's goal.

Later experiments with other games indicated that the following pattern of disobedience is usually more effective:

obey the human twice, disobey thrice, obey once, disobey four times, obey once, disobey thrice, obey thrice, disobey twice, obey thrice, disobey once, obey once, disobey once

Misinformation Unfortunately, most research in the field of artificial intelligence is just a lot of hot air. For years, researchers have been promising that intelligent, easy-to-use English-speaking computers and robots would be available at low prices “any day now”. After several decades of listening to such hoopla, I’ve given up waiting. The field of artificial intelligence should be renamed “artificial optimism”.

Whenever a researcher in the field of artificial intelligence promises you something, don’t believe it until you see it and use it personally, so you can evaluate its limitations.

If a computer seems to give intelligent replies to English questions posed by a salesman or researcher demonstrating artificial intelligence, try to interrupt the demo and ask the computer *your* English questions. You’ll typically find that the computer doesn’t understand what you’re talking about at all: the demo was a cheap trick that works just with the peculiar English questions asked by the demonstrator.

For many years, the top researchers in artificial intelligence have been exaggerating their achievements and underestimating how long it will take to develop a truly intelligent computer. Let’s look at their history of lies:

In 1957 Herbert Simon said, “Within ten years a digital computer will be the world’s chess champion.” In 1967, when the ten years had elapsed, the only decent chess program was Greenblatt’s, which the American Chess Federation rated “class D” (which means “poor”). A computer didn’t become the world chess champion until 1997. It took forty years, not ten!

In 1957 Simon also said, “Within ten years a digital computer will discover and prove an important new mathematical theorem.” He was wrong. The computer still hasn’t discovered or proved any important new mathematical theorem. The closest call came in 1976, when it did the *non-abstract part* of the proof of the “4-color theorem”.

In 1958 Newell, Simon, and Shaw wrote a chess-playing program which they admitted was “not fully debugged” so that one “cannot say very much about the behavior of the program”; but they claimed it was “good in spots (opening)”. In 1959 the founder of cybernetics, Norbert Wiener, exaggerated about their program; he told New York University’s Institute of Philosophy that “chess-playing machines as of now will counter the moves of a master player with the moves recognized as right in the textbooks, up to some point in the middle game.” In the same symposium Michael Scriven carried the exaggeration even further by saying, “Machines are already capable of a good game.” In fact, the program they were describing played very poorly, and in its last official bout (October 1960) was beaten by a 10-year-old kid who was a novice.

In 1960 Herbert Gelernter (who wrote the geometry-theorem program) said, “Today hardly an expert will contest the assertion that machines will be proving interesting theorems in number theory three years hence.” More than forty years have elapsed since then, but neither Gelernter nor anyone else has programmed the computer to prove theorems in number theory.

In June 1963 the *Chicago Tribune* said, “The development of a machine that can listen to any conversation and type out the remarks just like an office secretary was announced yesterday by a Cornell University expert on learning machines. The device is expected to be in operation by fall. Frank Rosenblatt, director of Cornell’s cognitive systems research, said the machine will be the largest thinking device built to date. Rosenblatt made his announcement at a meeting on learning machines at Northwestern University’s Technological Institute.” No such machine exists today, let alone in 1963.

Also in 1963, W. Ross Ashby said, “Gelernter’s theorem-proving program has discovered a new proof of the *pons asinorum* that demands no construction.” He said the proof is one that “the greatest mathematicians of 2000 years have failed to notice... which would have evoked the highest praise had it occurred.” In fact, the *pons asinorum* is just the simple theorem that the opposite angles of an isosceles triangle are equal, and the computer’s constructionless proof had already been discovered by Pappus in 300 A.D.

In 1968 the head of artificial intelligence in Great Britain, Donald Michie, said, “Today machines can play chess at championship level.” In fact, when computers were allowed to participate in human chess tournaments, they almost always lost.

In 1970 the head of artificial intelligence at MIT, Marvin Minsky, said, “In 3 to 8 years we will have a machine with the general intelligence of an average human being. I mean a machine that will be able to read Shakespeare, grease a car, play office politics, tell a joke, have a fight. At that point, the machine will begin to educate itself with fantastic speed. In a few months it will be at genius level, and a few months after that its powers will be incalculable.” His prediction that it would happen in 3 to 8 years — between 1973 and 1978 — was ridiculous. I doubt it will happen during this century, if ever.

Exaggerations concern not just the present and future but also the past:

Back in 1962, Arthur Samuel’s checker program won a game against Robert Nealey, “a former Connecticut checkers champion”.

Notice that Nealey was a *former* champion, not *the current* champion when the game was played. The program won a single game, not a match, and lost to Nealey later.

In 1971 James Slagle slid over those niceties, when he just said that the program “once beat the champion of Connecticut.” Later writers, reading Slagle’s words, went gone a step further and omitted the word *once*: one textbook said, “The program beat the champion of Connecticut”. It’s not true.

Why do leaders of artificial intelligence constantly exaggerate? To get more research funds from the government! Hubert Dreyfus, chairman of the philosophy department at Berkeley, annoys them by attacking their claims.

The brain Will the computer be able to imitate the human brain? Opinions vary.

Marvin Minsky, head of artificial intelligence at MIT, says *yes*: “After all, the human brain is just a computer that happens to be made out of meat.”

Biologists argue *no*: the brain is composed of 12 billion **neurons**, each of which has between 5,000 and 60,000 **dendrites** for input and a similar number of **axons** for output; the neurons act in peculiar ways, and no computer could imitate all that with complete accuracy — “The neuron is qualitatively quite different from on-off components of current computers.”

Herbert Simon (head of artificial intelligence at Carnegie and a psychologist), points out that certain aspects of the brain, such as short-term memory, are known to have very limited capacity and ability.

He believes the inner workings of the brain are reasonably simple; it produces complicated output just because it receives complicated input from the sense organs and environment:

“A man, viewed as a behaving system, is quite simple. The apparent complexity of his behavior over time is largely a reflection of the complexity of the environment in which he finds himself.”

Simon believes a computer would start acting in complex ways also, if it were given good sense organs, ability to move, elementary ability to learn, and the privilege of being placed in a stimulating environment (unlike a computer center’s dull four walls).

Hubert Dreyfus, chairman of the philosophy department at Berkeley, argues that progress in artificial intelligence has been very small, is being blocked now by impenetrable barriers, and — most important — the computer’s approach to solving problems bears little relationship to the more powerful methods used by humans. He’s cynical about the claim that an improvement in computer programs represents progress toward understanding the human mind, which is altogether different: “According to this definition, the first man to climb a tree could claim tangible progress toward reaching the moon. Rather than climbing blindly, it’s better to look where one is going.”