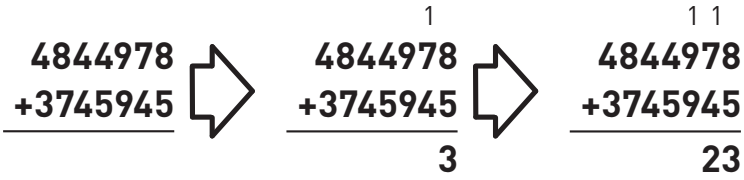
Nine Algorithms That Changed the Future

The Ingenious Ideas That
Drive Today's Computers

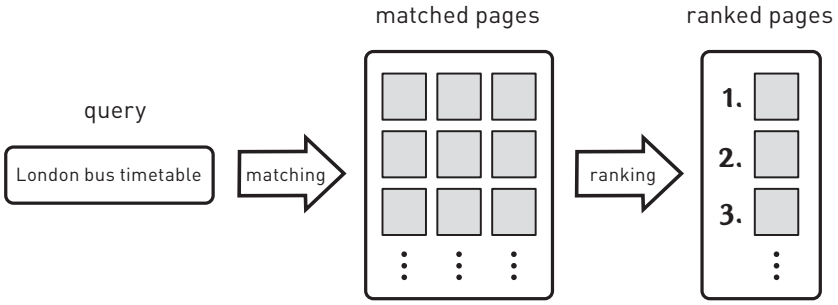
John MacCormick
Narrated by Quentin Cooper

with a foreword by Chris Bishop

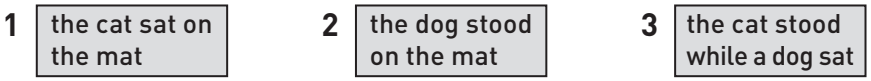
PRINCETON UNIVERSITY PRESS
PRINCETON AND OXFORD



From page 3. The first two steps in the algorithm for adding two numbers.



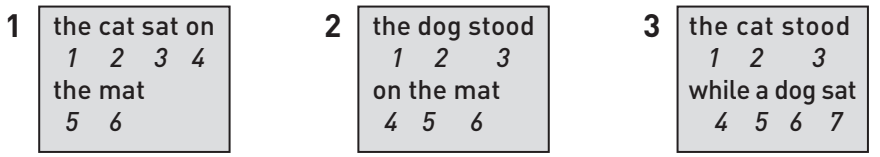
From page 11. The two phases of web search: matching and ranking. There can be thousands or millions of matches after the first (matching) phase, and these must be sorted by relevance in the second (ranking) stage.



From page 13. An imaginary World Wide Web that consists of only three pages, numbered 1, 2, and 3.

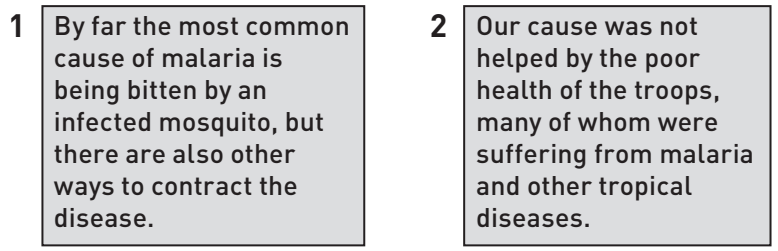
a	3
cat	1 3
dog	2 3
mat	1 2
on	1 2
sat	1 3
stood	2 3
the	1 2 3
while	3

From page 13. A simple index with page numbers.



a	3-5
cat	1-2 3-2
dog	2-2 3-6
mat	1-6 2-6
on	1-4 2-4
sat	1-3 3-7
stood	2-3 3-3
the	1-1 1-5 2-1 2-5 3-1
while	3-4

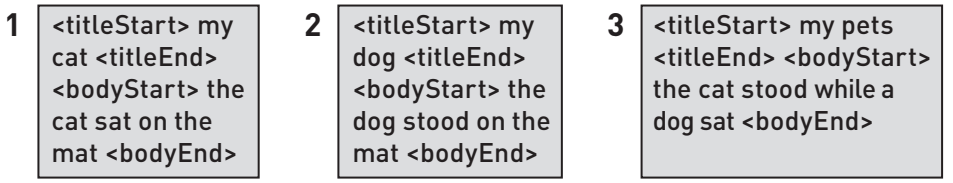
From page 16. Word location trick. Top: Our three web pages with in–page word locations added. Bottom: A new index that includes both page numbers and in-page word locations.



also	1-19	
...		
cause	1-6	2-2
...		
malaria	1-8	2-19
...		
whom	2-15	

From page 18. Top: Two example web pages that mention malaria. Bottom: Part of the index built from the above two web pages.

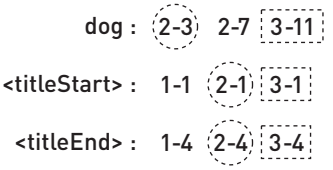
From page 19. An example set of web pages that each have a title and a body.



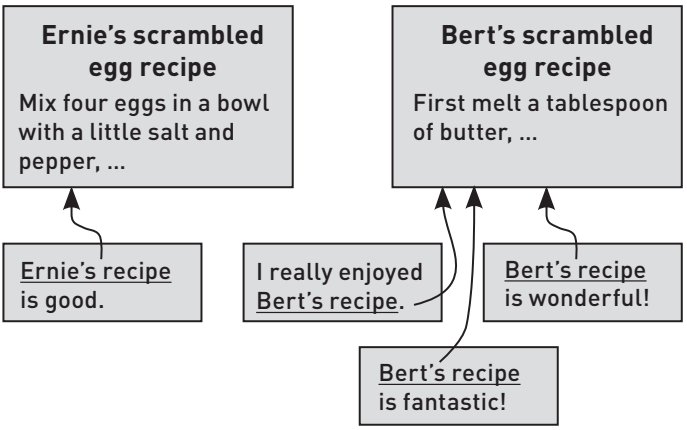
From page 20. The same set of web pages as in the last figure, but shown as they might be *written* with metawords, rather than as they would be displayed in a web browser.

a	3-10
cat	1-3 1-7 3-7
dog	2-3 2-7 3-11
mat	1-11 2-11
my	1-2 2-2 3-2
on	1-9 2-9
pets	3-3
sat	1-8 3-12
stood	2-8 3-8
the	1-6 1-10 2-6 2-10 3-6
while	3-9
<bodyEnd>	1-12 2-12 3-13
<bodyStart>	1-5 2-5 3-5
<titleEnd>	1-4 2-4 3-4
<titleStart>	1-1 2-1 3-1

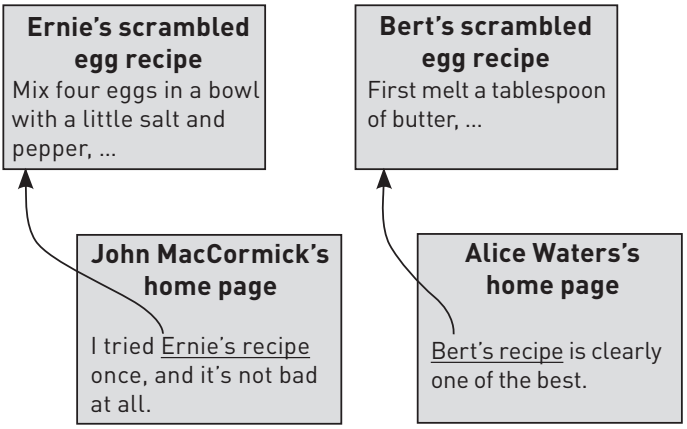
From page 21. The index for the web pages shown in the previous figure, including metawords.



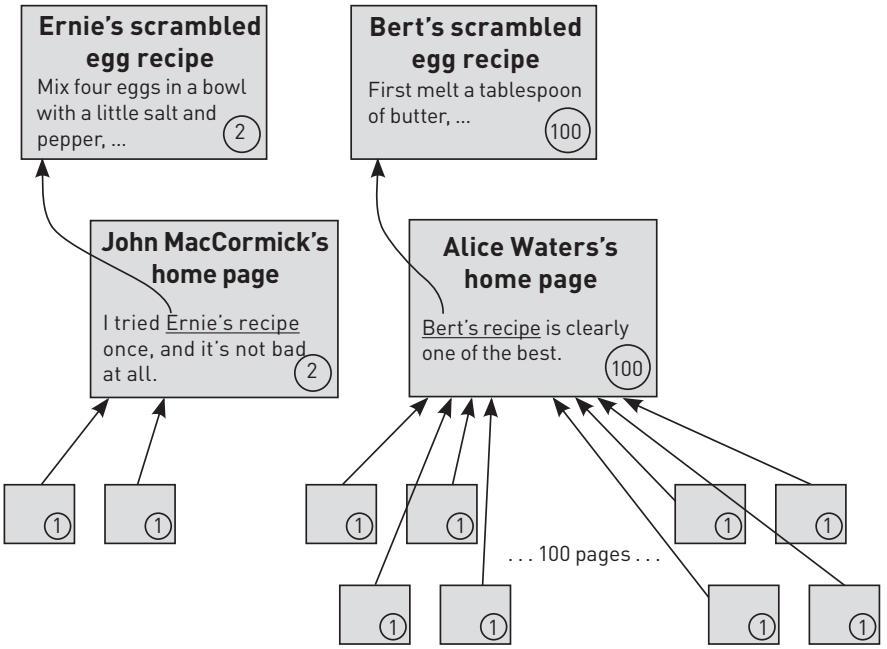
From page 21. How a search engine performs the search `dog IN TITLE`.



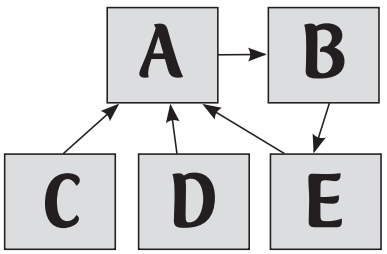
From page 26. The basis of the hyperlink trick. Six web pages are shown, each represented by a box. Two of the pages are scrambled egg recipes, and the other four are pages that have hyperlinks to these recipes. The hyperlink trick ranks Bert's page above Ernie's, because Bert has three incoming links and Ernie only has one.



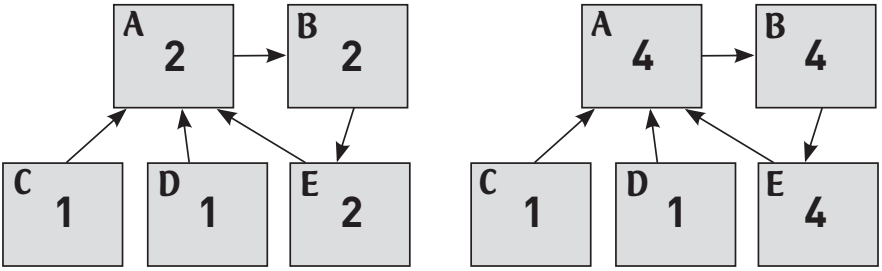
From page 28. The basis for the authority trick. Four web pages are shown: two scrambled egg recipes and two pages that link to the recipes. One of the links is from the author of this book (who is *not* a famous chef) and one is from the home page of the famous chef Alice Waters. The authority trick ranks Bert's page above Ernie's, because Bert's incoming link has greater "authority" than Ernie's.



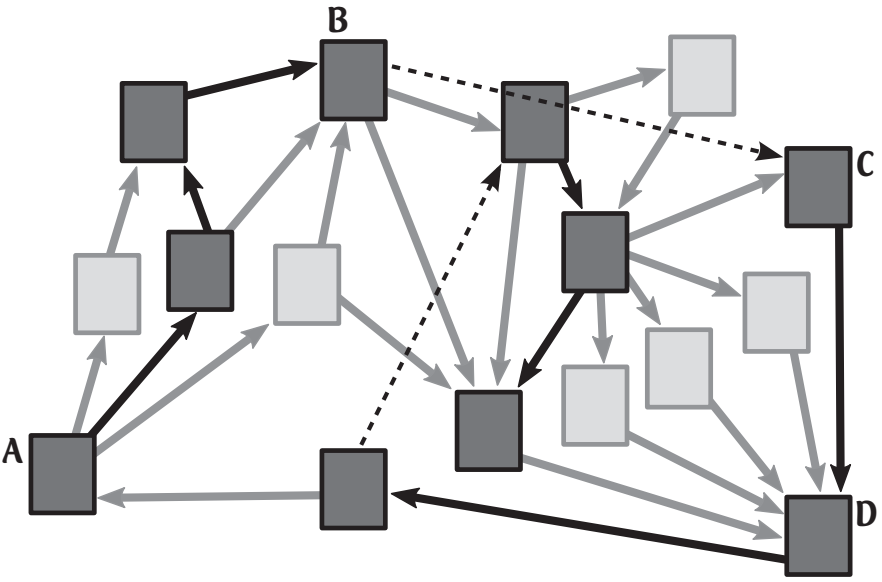
From page 29. Scrambled egg authority scores. A simple calculation of "authority scores" for the two scrambled egg recipes. The authority scores are shown in circles.



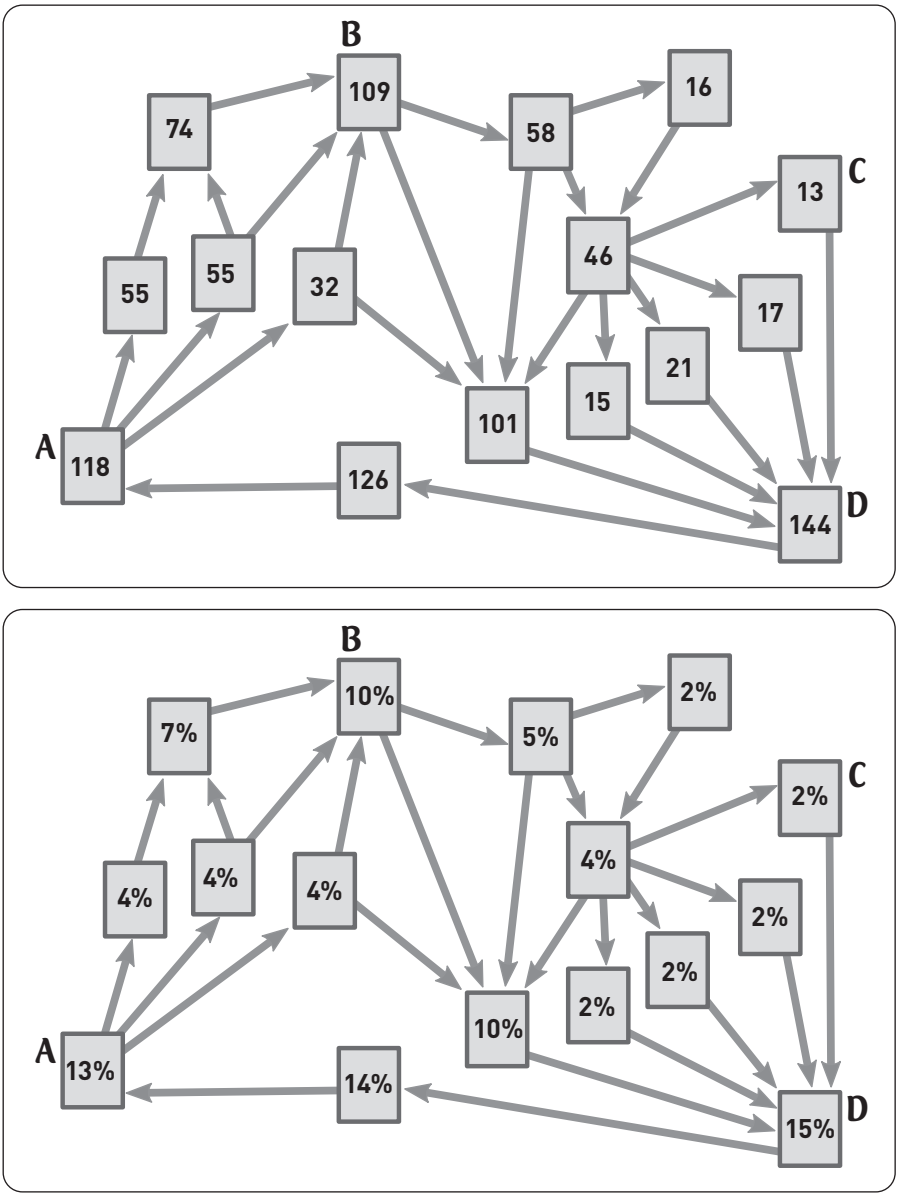
From page 30. Hyperlink cycles. An example of a cycle of hyperlinks. Pages *A*, *B*, and *E* form a cycle because you can start at *A*, click through to *B*, then *E*, and then return to your starting point at *A*.



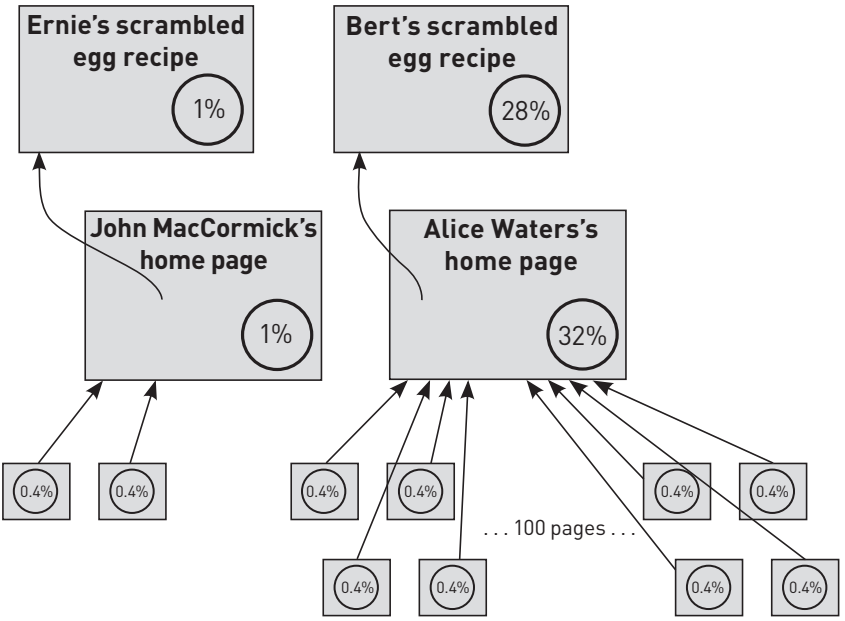
From page 30. The problem caused by cycles. *A*, *B*, and *E* are always out of date, and their scores keep growing forever.



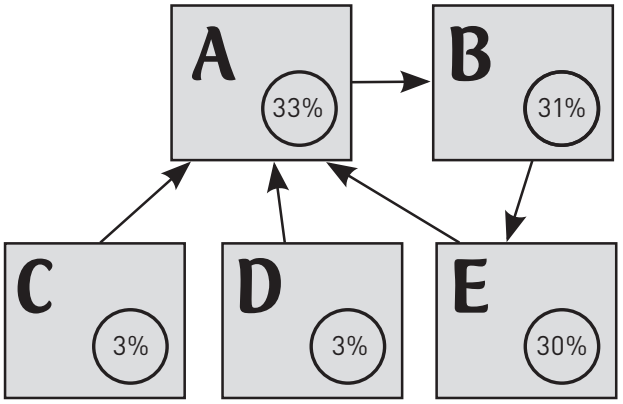
From page 31. The random surfer model. Pages visited by the surfer are darkly shaded, and the dashed arrows represent random restarts. The trail starts at page *A* and follows randomly selected hyperlinks interrupted by two random restarts.



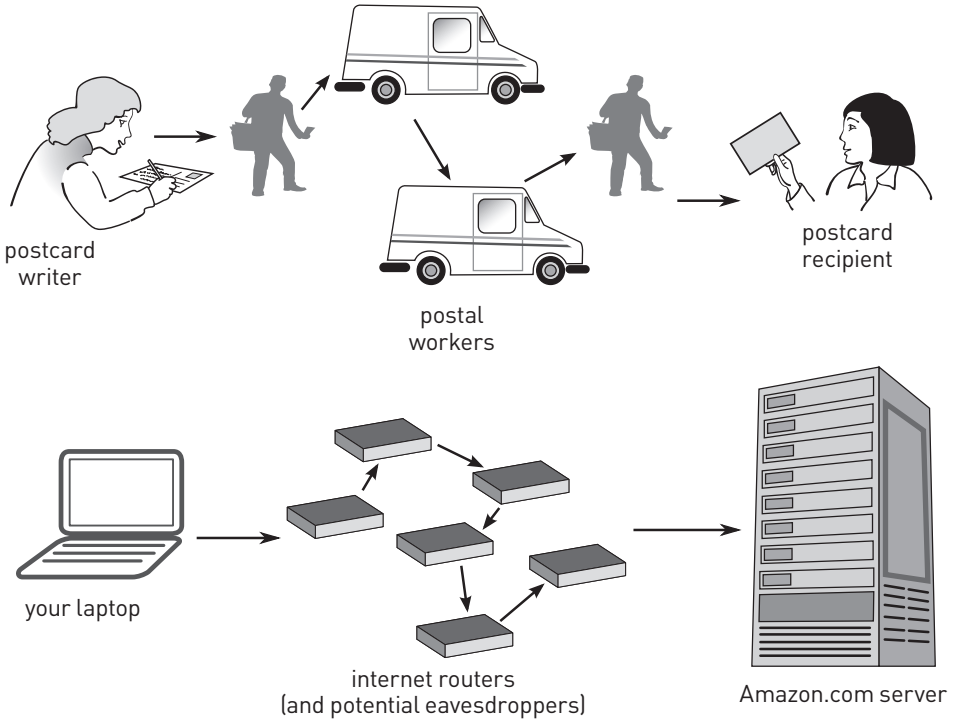
From page 33. Random surfer simulations. Top: Number of visits to each page in a 1000-visit simulation. Bottom: Percentage of visits to each page in a simulation of one million visits.



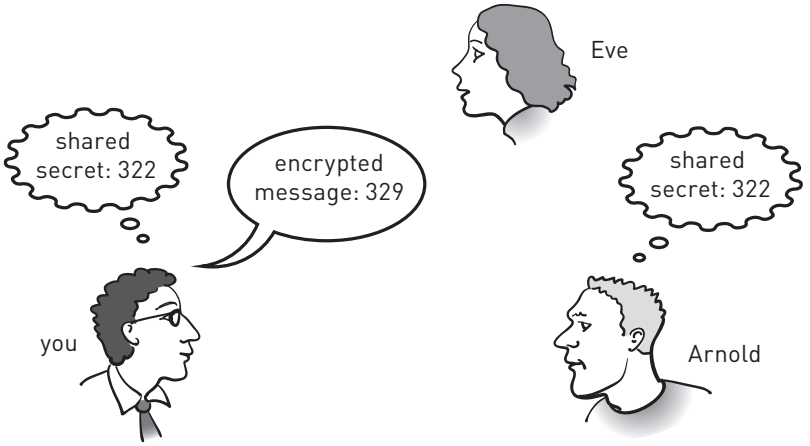
From page 34. Surfer authority scores for the scrambled egg example on page 6. Bert and Ernie each have exactly one incoming link conferring authority on their pages, but Bert's page will be ranked higher in a web search query for "scrambled eggs."



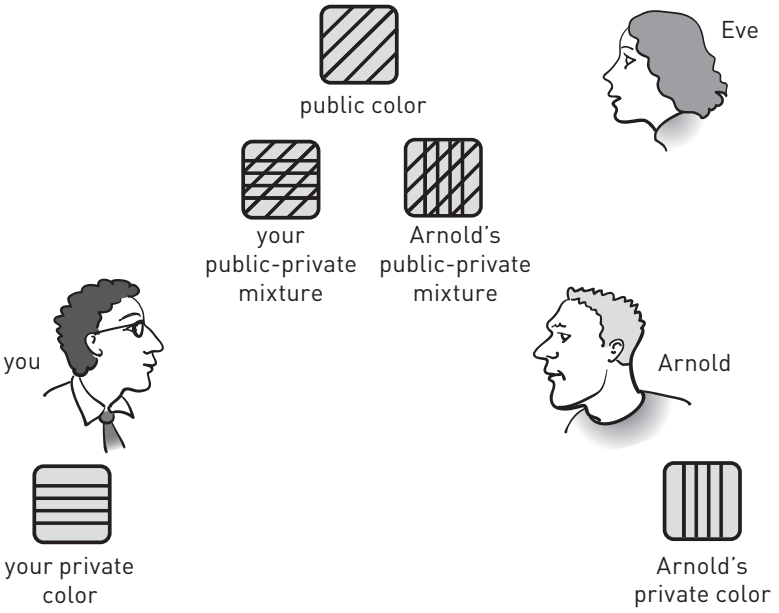
From page 35. Surfer authority scores for the earlier example with a cycle of hyperlinks (page 6). The random surfer trick has no trouble computing appropriate scores, despite the presence of a cycle ($A \to B \to E \to A$).



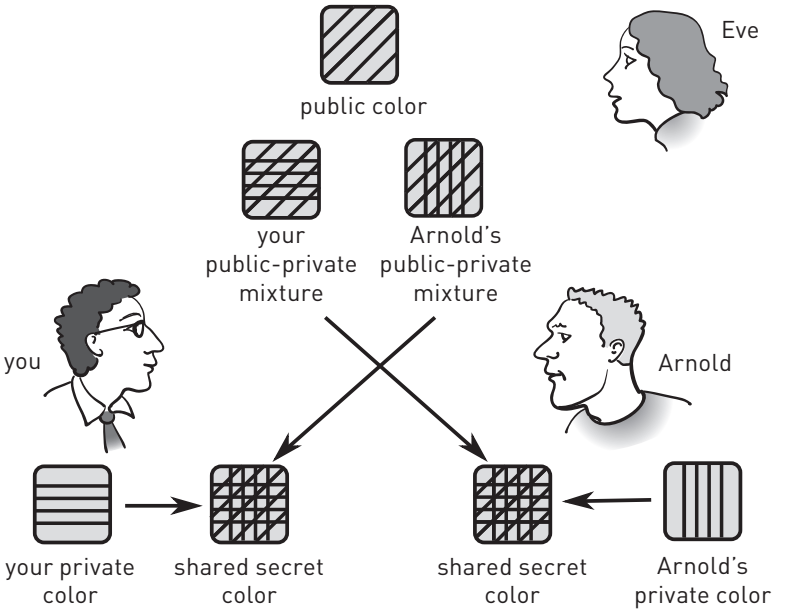
From page 39. The postcard analogy: It's obvious that sending a postcard through the mail system will not keep the contents of the postcard secret. For the same reason, a credit card number sent from your laptop to Amazon.com can easily be snooped by an eavesdropper if it is not properly encrypted.



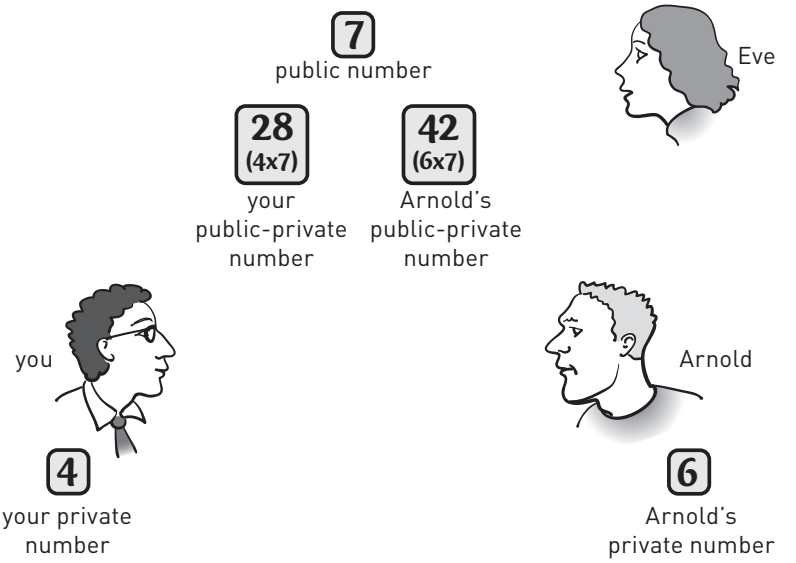
From page 41. The addition trick: The message 7 is encrypted by adding it to the shared secret, 322. Arnold can decrypt it by subtracting the shared secret, but Eve cannot.



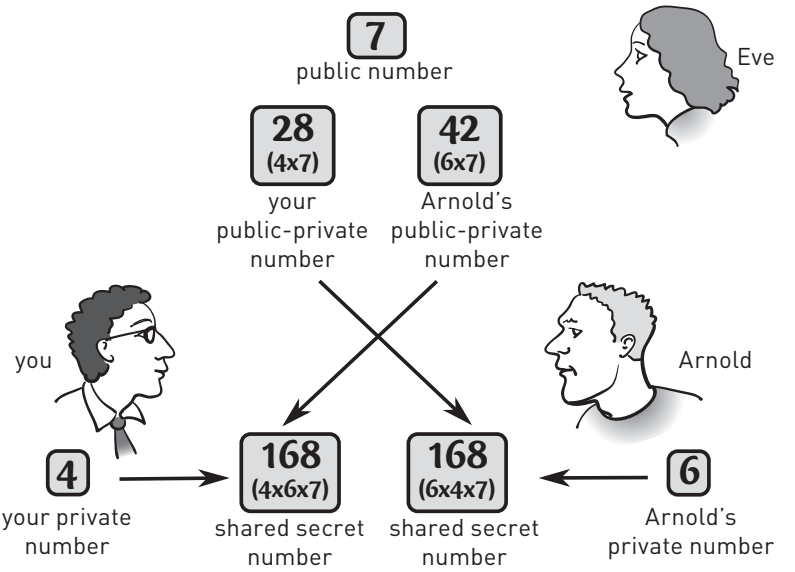
From page 46. The paint-mixing trick, step 3: The public–private mixtures are available to anyone who wants them.



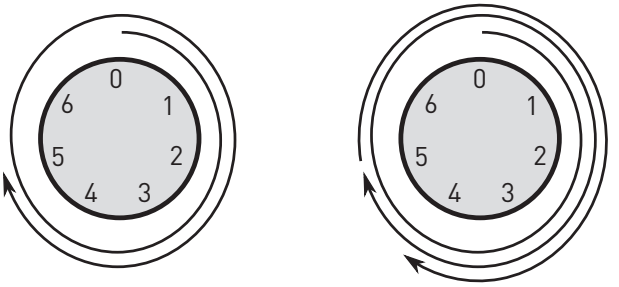
From page 47. The paint-mixing trick, step 4: Only you and Arnold can make the shared secret color, by combining the mixtures shown by arrows.



From page 50. The number-mixing trick, step 3: The public–private numbers are available to anyone who wants them.



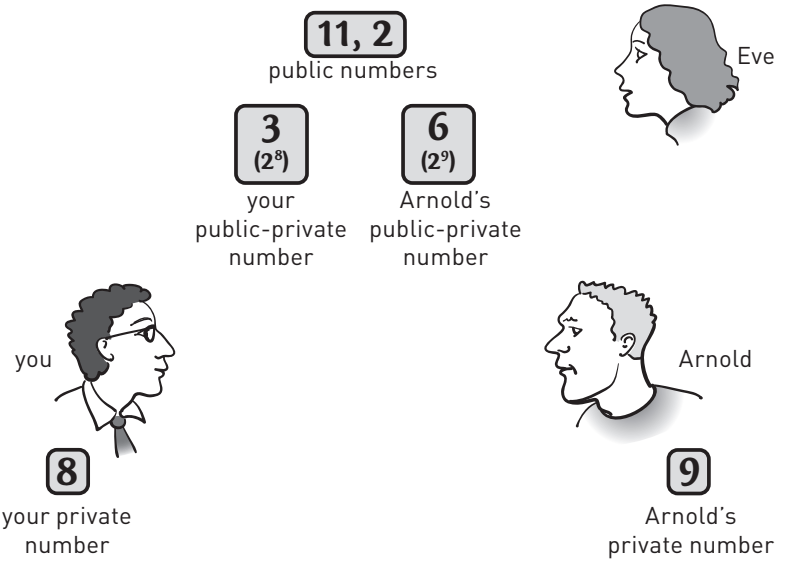
From page 51. The number-mixing trick, step 4: Only you and Arnold can make the shared secret number, by multiplying together the numbers shown by arrows.



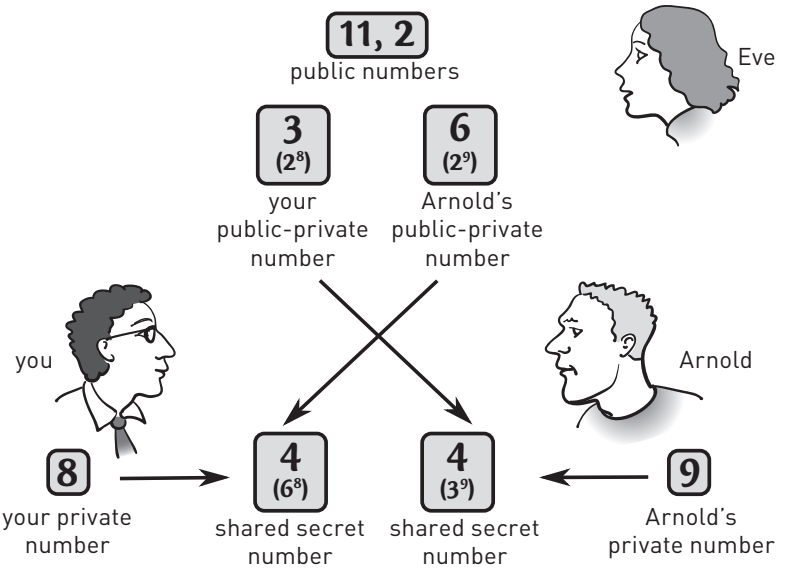
From page 53. Left: When using a clock size of 7, the number 12 is simplified to the number 5—just start at zero and count 12 units in a clockwise direction, as shown by the arrow. Right: Again using a clock size of 7, we find that $12+6=4$—starting at 5, where we ended in the left figure, add on another 6 units in clockwise direction.

n	2^n	3^n	6^n
1	2	3	6
2	4	9	3
3	8	5	7
4	5	4	9
5	10	1	10
6	9	3	5
7	7	9	8
8	3	5	4
9	6	4	2
10	1	1	1

From page 54. The first ten powers of 2, 3, and 6 using Clock Size 11. Each entry can be computed from the one above it by some very simple arithmetic. Let's take a look at the last column. The first entry in this column is 6, which is the same thing as 6^1. The next entry represents 6^2, or 36, but since we're using clock size 11 and 36 is 3 more than 33, the entry in the table is a 3. To calculate the third entry in this column, you might think that we need to work out $6^3 = 6 \times 6 \times 6$, but there is an easier way. We have already computed 6^2 for the clock size we're interested in—it turned out to be 3. To get 6^3, we just need to multiply the previous result by 6. This gives $3 \times 6 = 18 = 7$ (clock size 11). And the next entry is $7 \times 6 = 42 = 9$ (clock size 11), and so on down the column.

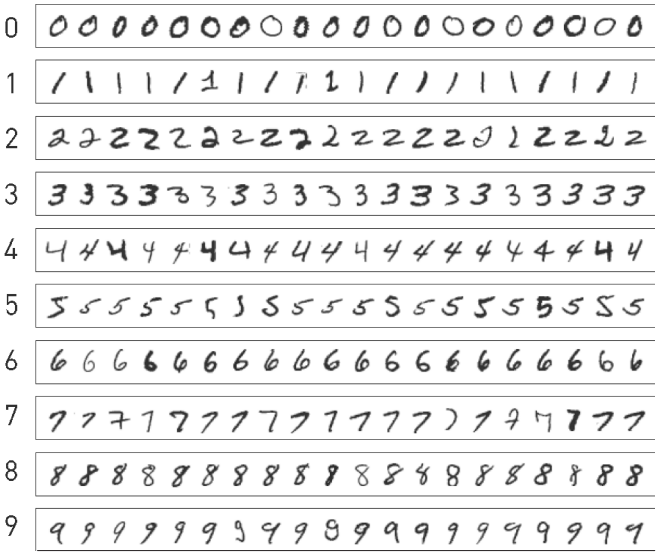


From page 56. Real-life number-mixing, step 3: The public–private numbers (3 and 6), computed using powers and clock arithmetic, are available to anyone who wants them. The "2^8" shown below the 3 reminds us how the 3 was computed, but the fact that $3 = 2^8$ in clock size 11 is not made public. Similarly, the "2^9" shown below the 6 remains private.

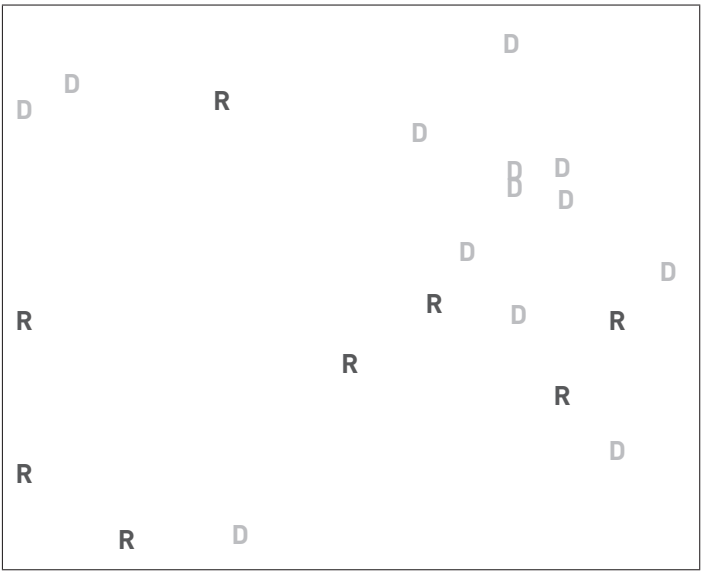


From page 57. Real-life number-mixing, step 4: Only you and Arnold can make the shared secret number, by combining together the elements shown with arrows, using powers and clock arithmetic.

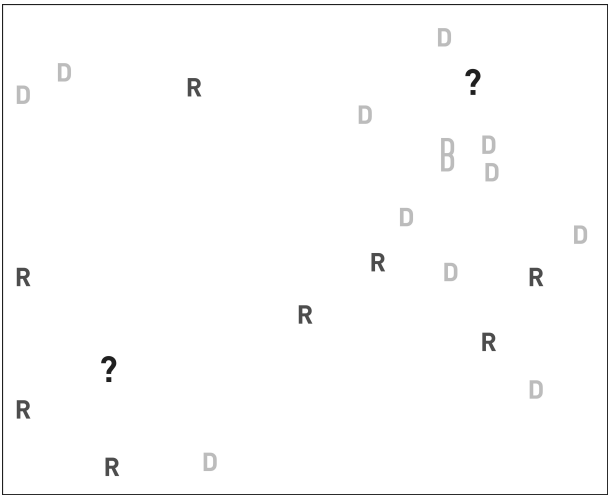
From page 83. Data samples of hand written digits. Most pattern recognition tasks can be phrased as classification problems. Here, the task is to classify each handwritten digit as one of the 10 digits $0, 1, \ldots, 9$. Data source: MNIST data of LeCun *et al.* 1998.



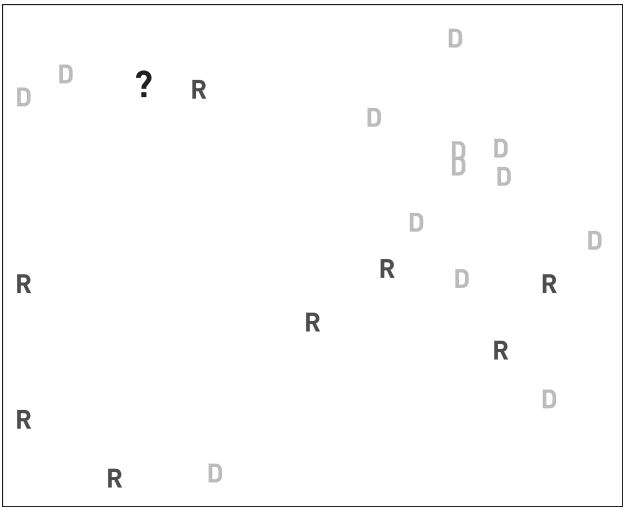
From page 84. To train a classifier, a computer needs some labeled data. Here, each sample of data (a handwritten digit) comes with a label specifying one of the 10 possible digits. The labels are on the left, and the training samples are in boxes on the right. Data source: MNIST data of LeCun *et al.* 1998.



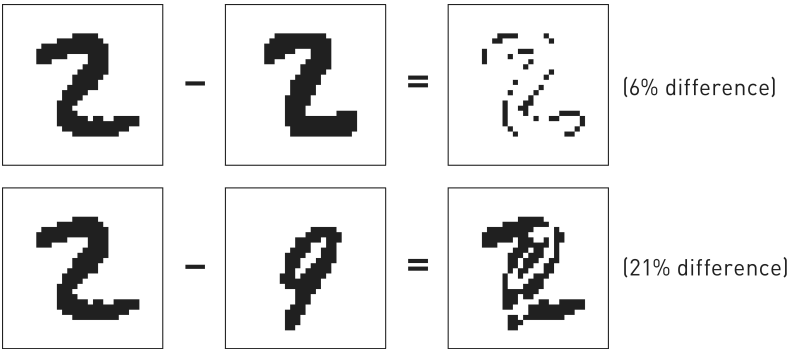
From page 85. Training data for predicting political party donations. A "D" marks a house that donated to the Democrats, and "R" marks Republican donations. Data source: Fundrace project, Huffington Post.



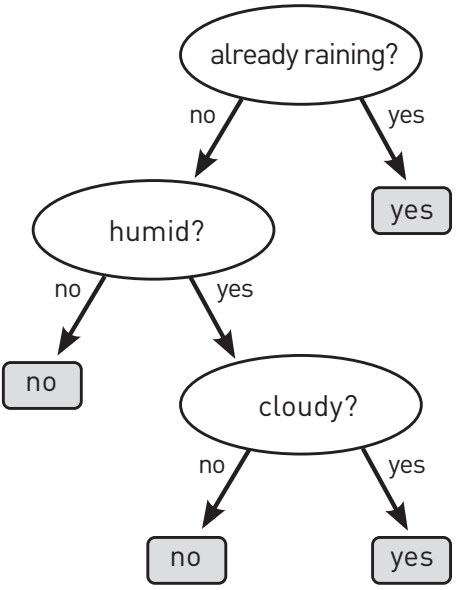
From page 86. Classification using the nearest-neighbor trick. Each question mark is assigned the class of its nearest neighbor. The upper question mark becomes a "D," while the lower one becomes an "R." Data source: Fundrace project, Huffington Post.



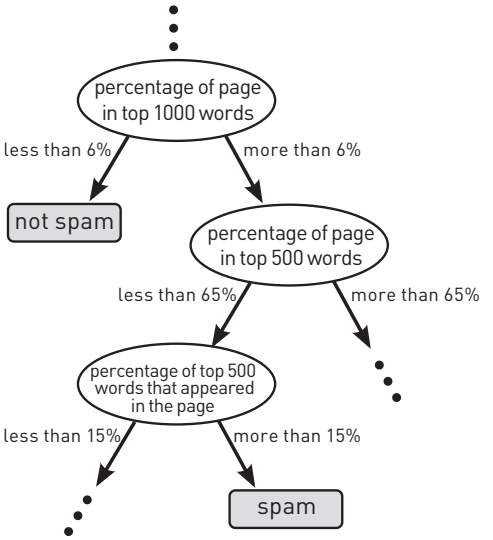
From page 87. An example of using *K*-nearest-neighbors. When using only the single nearest neighbor, the question mark is classified as an "R," but with three nearest neighbors, it becomes a "D." Data source: Fundrace project, Huffington Post.



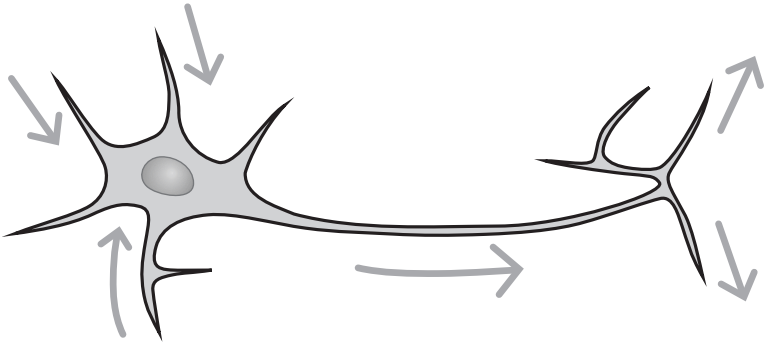
From page 88. Computing the "distance" between two handwritten digits. In each row, the second image is subtracted from the first one, resulting in a new image on the right that highlights the differences between the two images. The percentage of this difference image that is highlighted can be regarded as a "distance" between the original images. Data source: MNIST data of LeCun *et al.*, 1998.



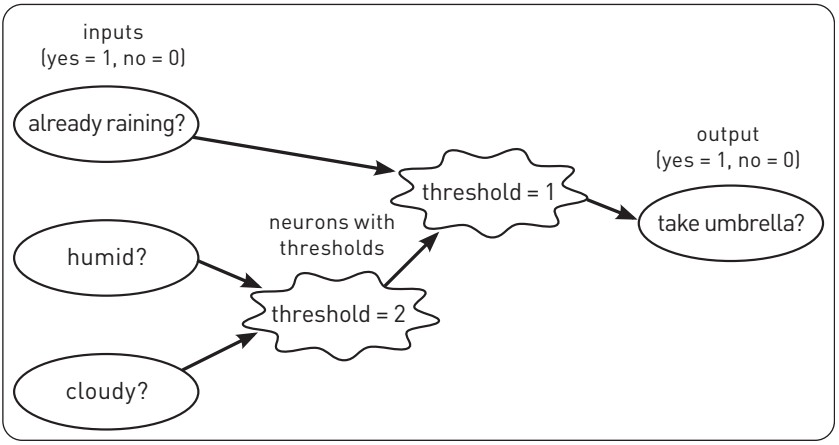
From page 90. Decision tree for "Should I take an umbrella?"

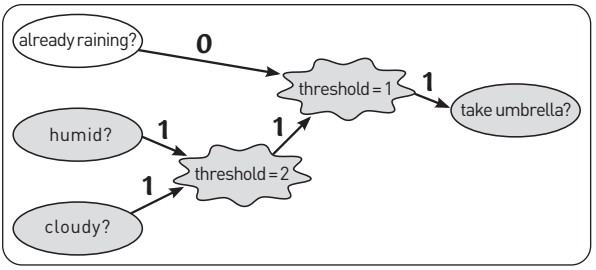


From page 92. Part of a decision tree for identifying web spam. The dots indicate parts of the tree that have been omitted for simplicity. Source: Ntoulas *et al.* 2006.

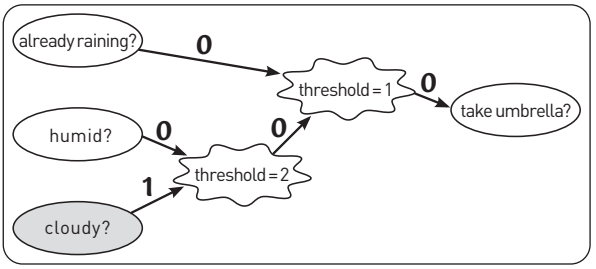


From page 93. A typical biological neuron. Electrical signals flow in the directions shown by the arrows. The output signals are only transmitted if the sum of the input signals is large enough.





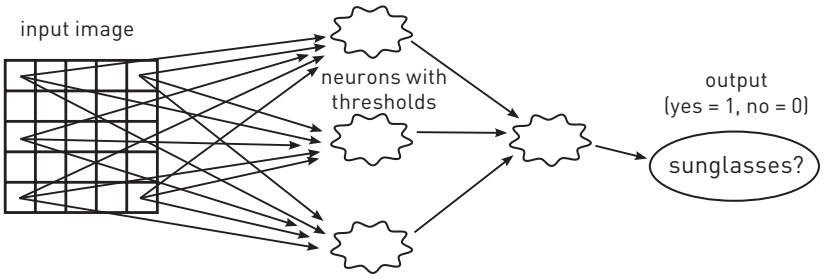
humid and cloudy, but not raining



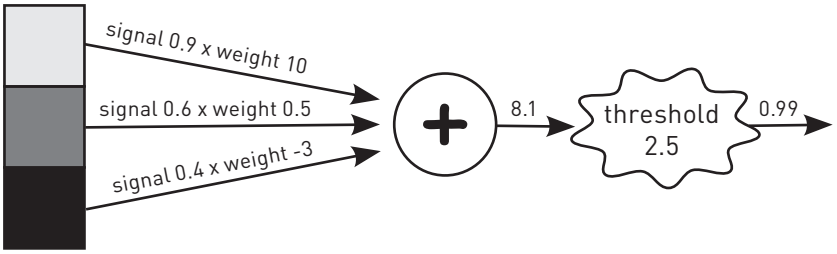
cloudy, but neither humid nor raining

From page 95. Umbrella neural network. Top panel: A neural network for the umbrella problem. Bottom two panels: The umbrella neural network in operation. Neurons, inputs, and outputs that are "firing" are shaded. In the center panel, the inputs state that it is not raining, but it is both humid and cloudy, resulting in a decision to take an umbrella. In the bottom panel, the only active input is "cloudy?," which feeds through to a decision not to take an umbrella.

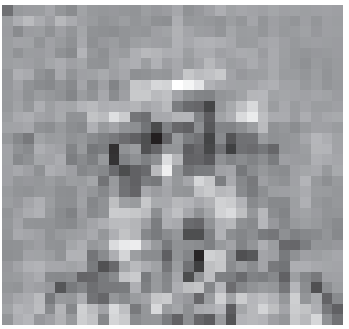
From page 96. Faces to be “recognized” by a neural network. In fact, instead of recognizing faces, we will tackle the simpler problem of determining whether a face is wearing sunglasses. Source: Tom Mitchell, *Machine Learning*, McGraw-Hill (1998). Used with permission.



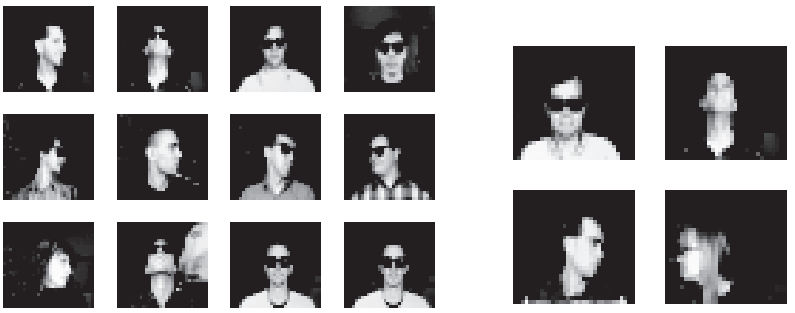
From page 97. A neural network for the sunglasses problem.



From page 99. Signals are multiplied by a connection weight before being summed.



From page 101. Weights (i.e., strengths) of inputs to one of the central neurons in the sunglasses network.



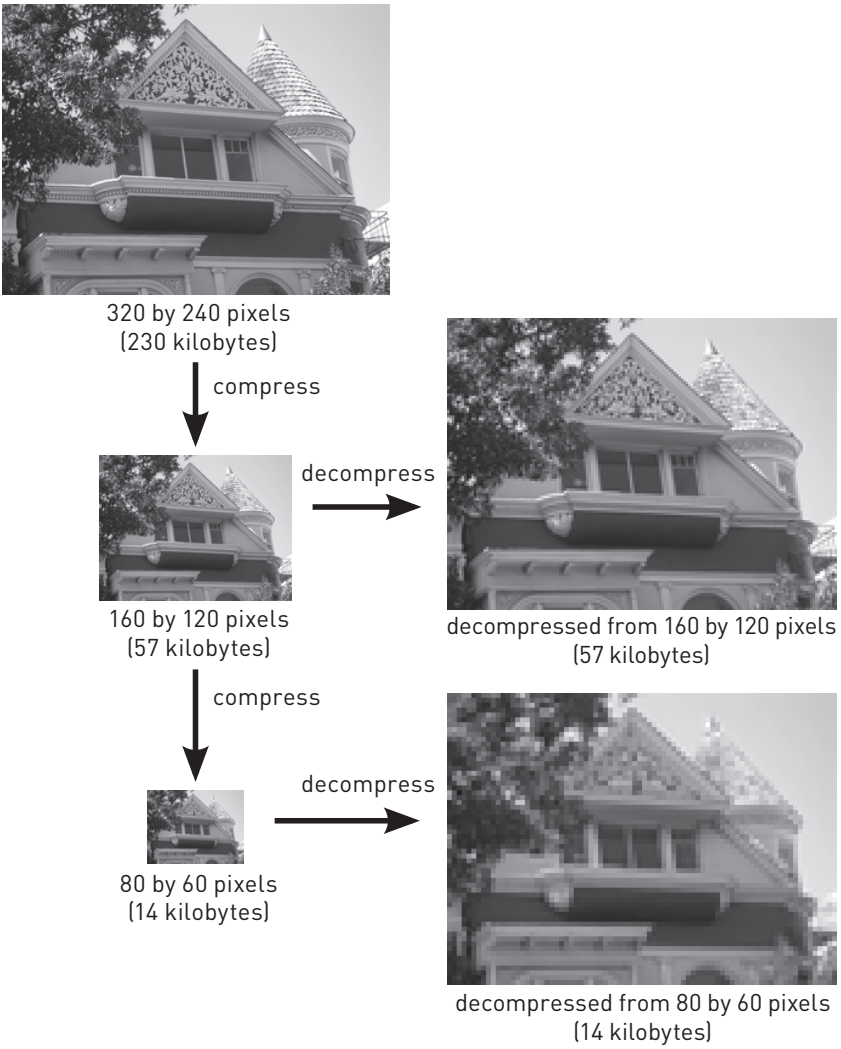
From page 102. Results from the sunglasses network. Source: Tom Mitchell, *Machine Learning*, McGraw-Hill (1998). Used with permission.

space 00	T 20	n 40	(60	á 80
A 01	U 21	o 41	) 61	à 81
B 02	V 22	p 42	* 62	é 82
C 03	W 23	q 43	+ 63	è 83
D 04	X 24	r 44	, 64	í 84
E 05	Y 25	s 45	- 65	ì 85
F 06	Z 26	t 46	. 66	ó 86
G 07	a 27	u 47	/ 67	ò 87
H 08	b 28	v 48	: 68	ú 88
I 09	c 29	w 49	; 69	ù 89
J 10	d 30	x 50	< 70	Á 90
K 11	e 31	y 51	= 71	À 91
L 12	f 32	z 52	> 72	É 92
M 13	g 33	! 53	? 73	È 93
N 14	h 34	" 54	{ 74	Í 94
O 15	i 35	# 55	| 75	Ì 95
P 16	j 36	$ 56	} 76	Ó 96
Q 17	k 37	% 57	– 77	Ò 97
R 18	l 38	& 58	Ø 78	Ú 98
S 19	m 39	’ 59	ø 79	Ù 99

From page 111. Shorter symbol trick part 1. Numeric codes that a computer could use for storing symbols.

space	00	T	20	n	40	(	60	**á**	**780**
A	01	U	21	o	41	)	61	**à**	**781**
B	02	V	22	p	42	*	62	**é**	**782**
C	03	W	23	q	43	+	63	**è**	**783**
D	04	X	24	r	44	,	64	**í**	**784**
E	05	Y	25	s	45	-	65	**ì**	**785**
F	06	Z	26	**t**	**9**	.	66	**ó**	**786**
G	07	a	27	u	47	/	67	**ò**	**787**
H	08	b	28	v	48	:	68	**ú**	**788**
I	09	c	29	w	49	;	69	**ù**	**789**
J	10	d	30	x	50	**<**	**770**	**Á**	**790**
K	11	**e**	**8**	y	51	**=**	**771**	**À**	**791**
L	12	f	32	z	52	**>**	**772**	**É**	**792**
M	13	g	33	!	53	**?**	**773**	**È**	**793**
N	14	h	34	"	54	**{**	**774**	**Í**	**794**
O	15	i	35	#	55	**|**	**775**	**Ì**	**795**
P	16	j	36	$	56	**}**	**776**	**Ó**	**796**
Q	17	k	37	%	57	**–**	**777**	**Ò**	**797**
R	18	l	38	&	58	**Ø**	**778**	**Ú**	**798**
S	19	m	39	’	59	**ø**	**779**	**Ù**	**799**

From page 114. Shorter symbol trick part 2. Changes to the table on page 111 are shown in bold. The codes for two common letters have been shortened, at the expense of lengthening the codes for a larger number of uncommon symbols. This results in a shorter total length for most messages.



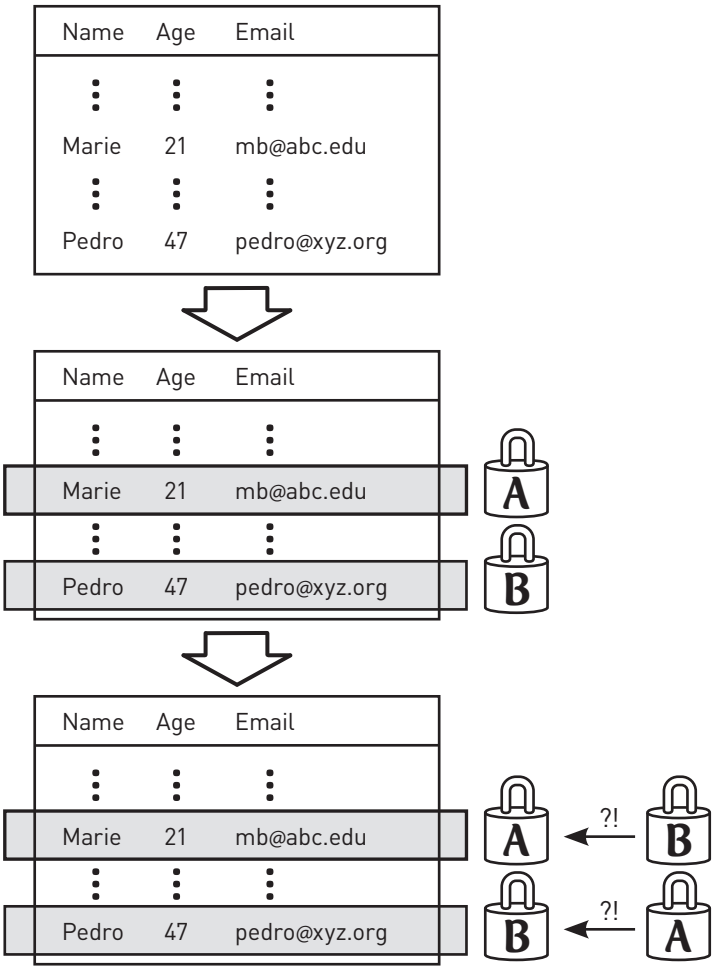
From page 117. Compression using the leave-it-out trick. The left column shows the original image, and two smaller, reduced versions of this image. Each reduced image is computed by leaving out half of the rows and columns in the previous one. In the right column, we see the effect of decompressing the reduced images to the same size as the original. The reconstruction is not perfect and there are some noticeable differences between the reconstructions and the original.

JPEG (35 kilobytes)

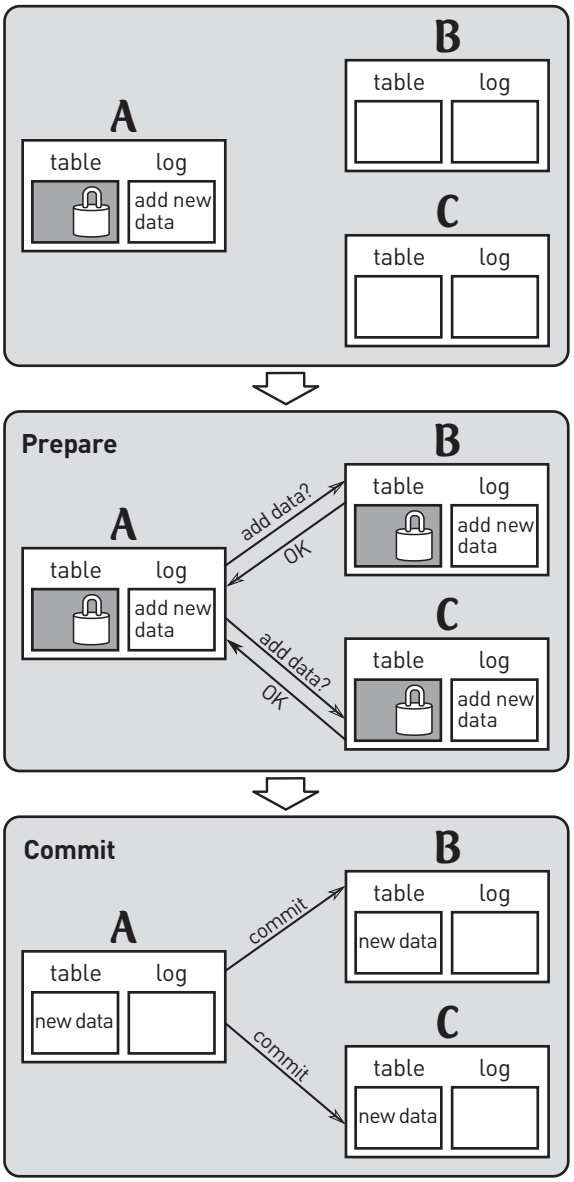
JPEG (19 kilobytes)

JPEG (12 kilobytes)

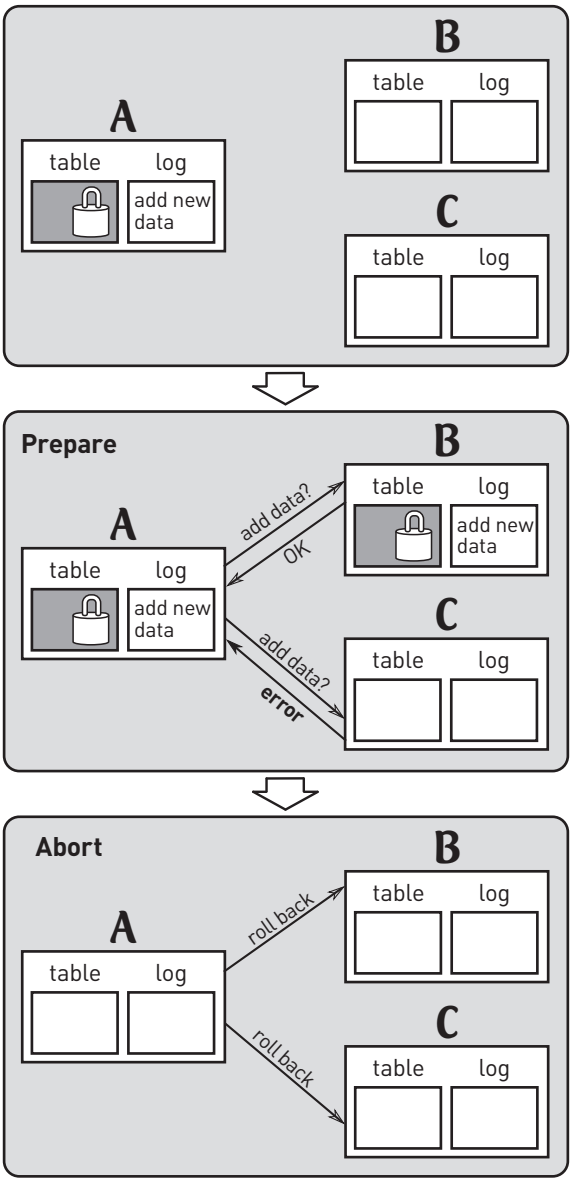
From page 119. With lossy compression schemes, higher compression produces lower quality. The same image is shown compressed at three different JPEG quality levels. At the top is the highest quality, which also requires the most storage. At the bottom is the lowest quality, which requires less than half the storage, but now there are noticeable compression artifacts—especially in the sky and along the border of the roof.



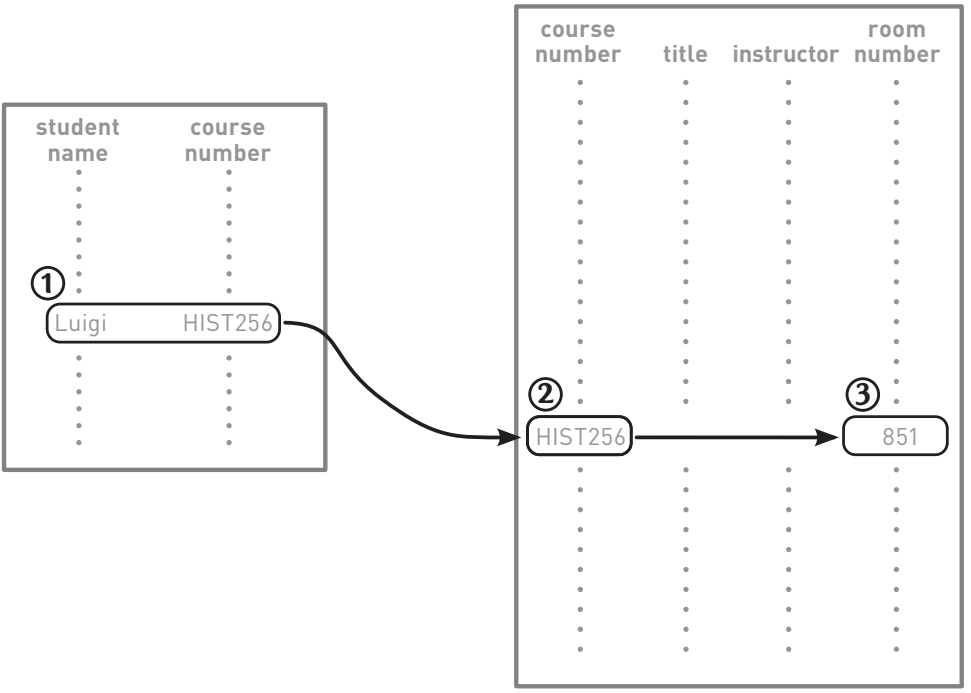
From page 135. Deadlock: When two transactions, *A* and *B*, both try to lock the same rows—but in the opposite order—they become deadlocked, and neither can proceed.



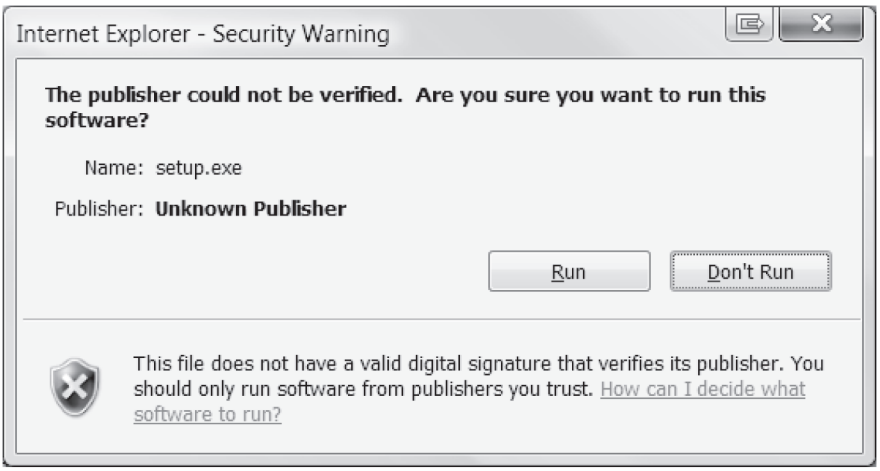
From page 139. The prepare-then-commit trick: The master replica, *A*, coordinates two other replicas (*B*, *C*) to add some new data to the table. In the prepare phase, the master checks whether all replicas will be able to complete the transaction. Once it gets the all clear, the master tells all replicas to commit the data.



From page 140. The prepare-then-commit trick with rollback: The top panel of this figure is exactly the same as in the previous figure. But during the prepare phase, one of the replicas encounters an error. As a result, the bottom panel is an "abort" phase in which each replica must roll back the transaction.

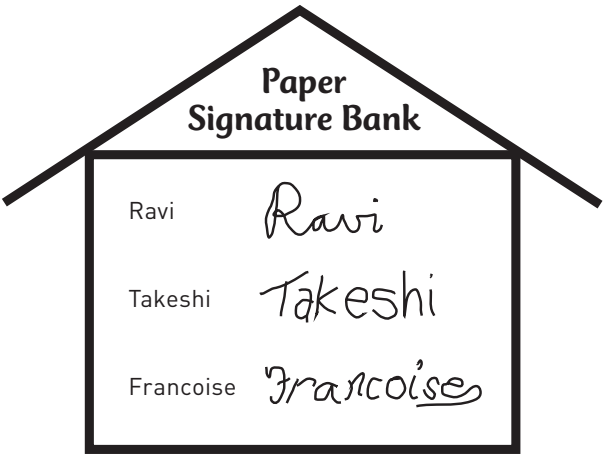


From page 144. Looking up data using a key: To find out the room number for Luigi's history course, we first find the relevant course number in the left-hand table. This value, "HIST256," is then used as a *key* in the other table. Because the column of course numbers is sorted in alphabetical order, we can find the correct row very quickly, then obtain the corresponding room number (851).

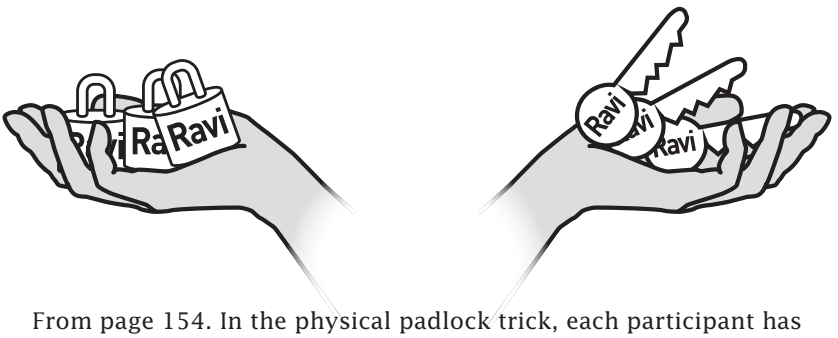


From page 150. Your computer checks digital signatures automatically. Top: The message my web browser displays when I attempt to download and run a program that has a valid digital signature. Bottom: The result of an invalid or missing digital signature.

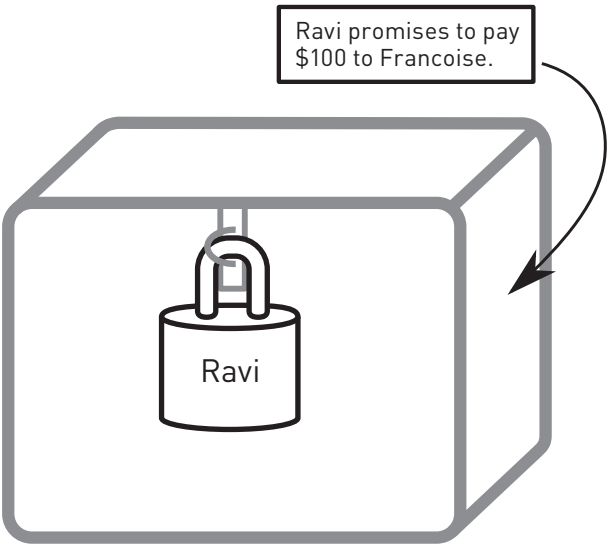
From page 152. A paper document with a handwritten signature.



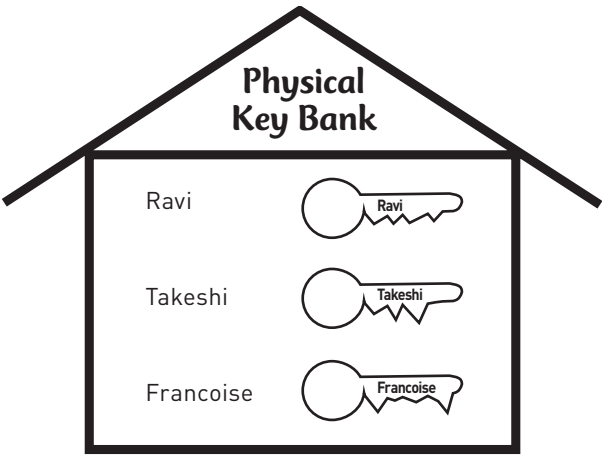
From page 152. A bank that stores the identities of its customers together with handwritten signatures on file.



From page 154. In the physical padlock trick, each participant has an exclusive supply of identical padlocks and keys.



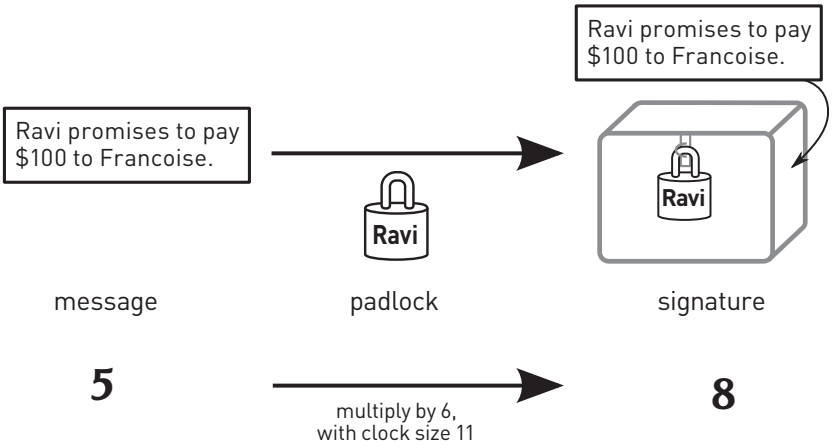
From page 155. To make a verifiable signature using the physical padlock trick, Ravi places a copy of the document in a lockbox and locks it with one of his padlocks.



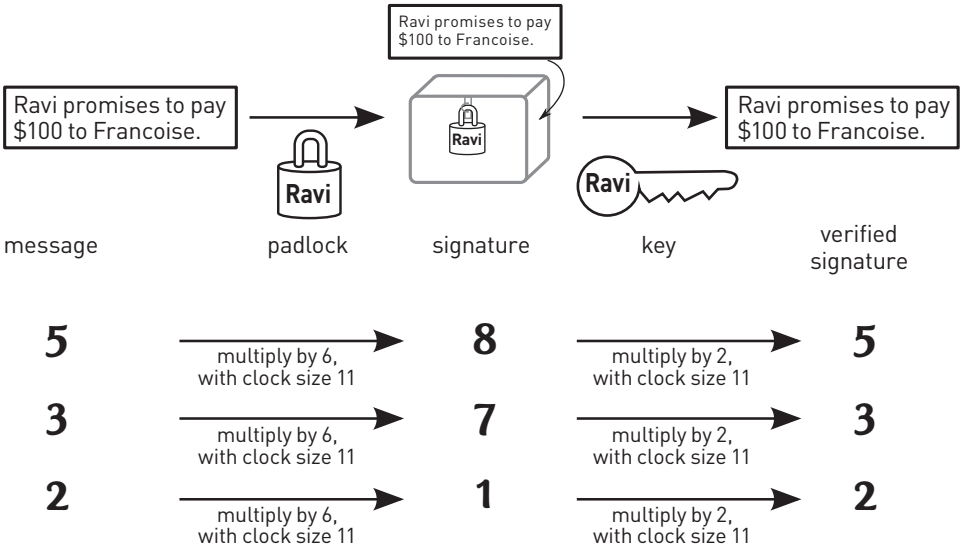
From page 156. A physical key bank stores keys that will open each participant's padlocks. Note that each of the keys is different.

	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	1	3	5	7	9
3	3	6	9	1	4	7	10	2	5	8
4	4	8	1	5	9	2	6	10	3	7
5	5	10	4	9	3	8	2	7	1	6
6	6	1	7	2	8	3	9	4	10	5
7	7	3	10	6	2	9	5	1	8	4
8	8	5	2	10	7	4	1	9	6	3
9	9	7	5	3	1	10	8	6	4	2
10	10	9	8	7	6	5	4	3	2	1

From page 157. Clock size 11 multiplication table.



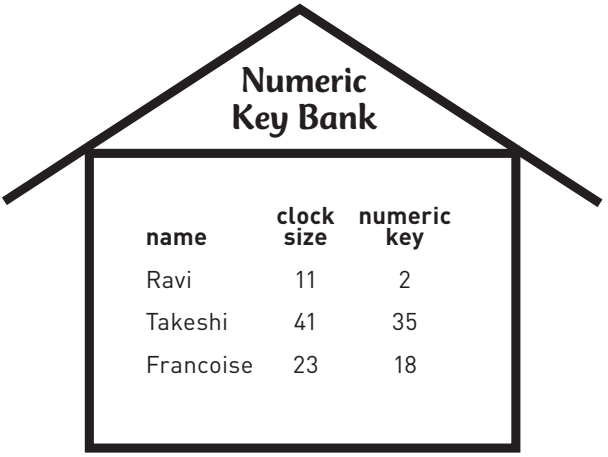
From page 158. How to "lock" a numeric message using a "padlock," creating a digital signature. The top row shows how to physically lock a message in a box using a physical padlock. The bottom row shows the analogous mathematical operation, in which the message is a number (5), the padlock is another number (6), and the process of locking corresponds to multiplication with a given clock size. The final result (8) is the digital signature for the message.



From page 159. How to "lock" and subsequently "unlock" a message using a numeric padlock and a corresponding numeric key. The top row shows the physical version of locking and unlocking. The next three rows show examples of numerically locking and unlocking messages using multiplication. Note that the locking process produces a digital signature, whereas the unlocking process produces a message. If the unlocked message matches the original message, the digital signature is verified and the original message is authentic.

Message	**Digital signature** (For genuine signature, multiply message by padlock value 9. To forge, choose a random number.)	**Unlocked signature** (To unlock signature, multiply by key value 5.)	**Matches message?**	**Forged?**
4	3	4	Yes	No
8	6	8	Yes	No
8	7	2	No!	Yes!

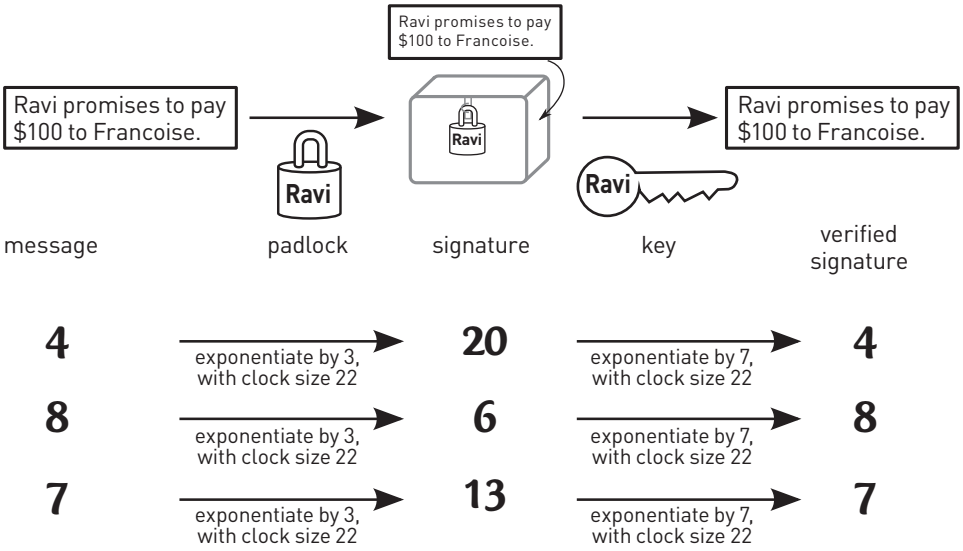
From page 160. How to detect a forged digital signature. These examples use a padlock value of 9 and a key value of 5. The first two signatures are genuine, but the third is forged.



From page 161. A numeric key bank. The role of the bank is not to keep the numeric keys and clock sizes secret. Instead, the bank is a trusted authority for obtaining the true key and clock size associated with any individual. The bank freely reveals this information to anyone who asks for it.

n	n^3	n^7	n	n^3	n^7
1	1	1	11	11	11
2	8	18	12	12	12
3	5	9	13	19	7
4	20	16	14	16	20
5	15	3	15	9	5
6	18	8	16	4	14
7	13	17	17	7	19
8	6	2	18	2	6
9	3	15	19	17	13
10	10	10	20	14	4

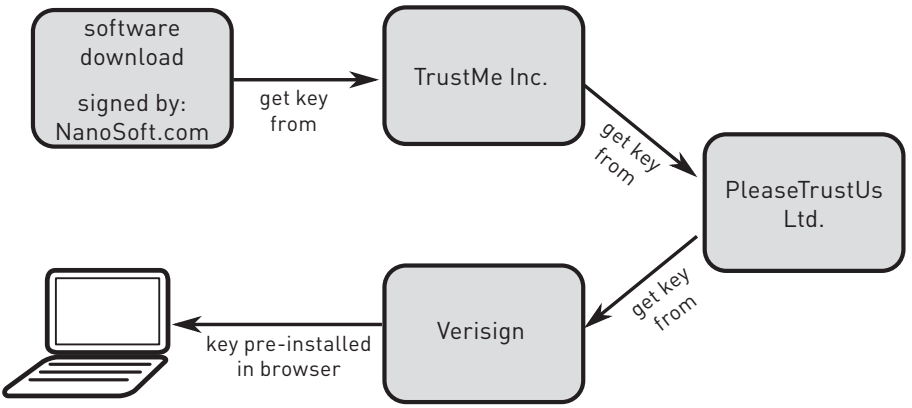
From page 164. Exponentiating by 3 and 7 with clock size 22.



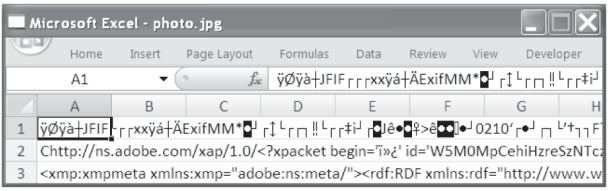
From page 165. Locking and unlocking messages using exponentiation.

Message	**Digital signature** (For genuine signature, exponentiate message by padlock value 3. To forge, choose a random number.)	**Unlocked signature** (To unlock signature, exponentiate by key value 7.)	**Matches message?**	**Forged?**
4	20	4	Yes	No
8	6	8	Yes	No
8	9	15	No!	Yes!

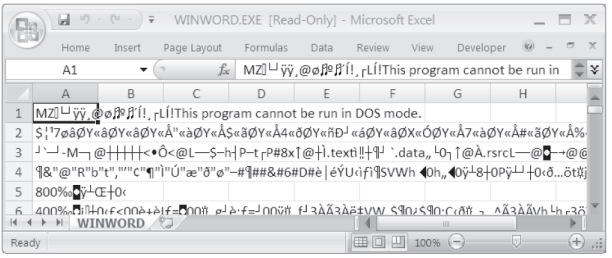
From page 166. How to detect a forged digital signature with exponentiation. These examples use a padlock value of 3, a key value of 7, and a clock size of 22. The first two signatures are genuine, but the third is forged.



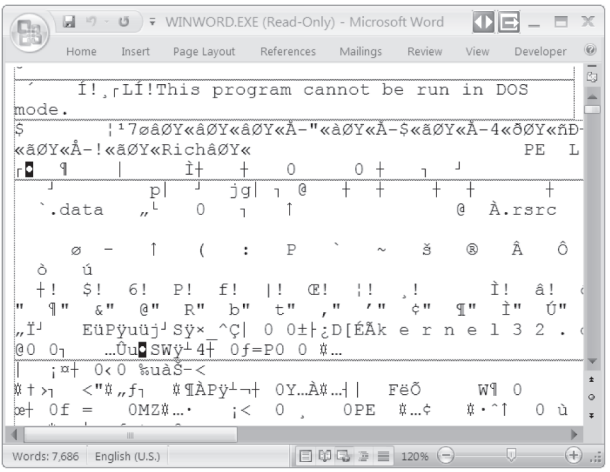
From page 172. A chain of trust for obtaining keys needed to verify digital signatures.



From page 180. Microsoft Excel run with “photo.jpg” as its input. The output is garbage, but the important point is that you can, in principle, run any program on any input you want.



From page 181. Microsoft Excel examines Microsoft Word. When Excel opens the file WINWORD.EXE, the result is—unsurprisingly—garbage.



From page 182. Microsoft Word examines itself. The open document is the file WINWORD.EXE, which is the actual computer program run when you click on Microsoft Word.

From page 193. The result of a crash on one particular operating system. Different operating systems handle crashes in different ways, but we all know one when we see one. This TroubleMaker.exe program was deliberately written to cause a crash, demonstrating that intentional crashes are easy to achieve.