Software is everywhere. However, it's written by people so it's not perfect, as the following examples show.

1. Disney's Lion King, 1994-1995

In the fall of 1994, the Disney company released its first multimedia CD-ROM game for children, The Lion King Animated Storybook. Although many other companies had been marketing children's programs for years, this was Disney's first venture into the market and it was highly promoted and advertised. Sales were huge. It was "the game to buy" for children that holiday season. What happened, however, was a huge debacle. On December 26, the day after Christmas, Disney's customer support phones began to ring, and ring, and ring. Soon the phone support technicians were swamped with calls from angry parents with crying children who couldn't get the software to work. Numerous stories appeared in newspapers and on TV news.

It turns out that Disney failed to test the software on a broad representation of the many different PC models available on the market. The software worked on a few systems likely the ones that the Disney programmers used to create the game but not on the most common systems that the general public had.

2. Intel Pentium Floating-Point Division Bug, 1994

Enter the following equation into your PC's calculator:

```
(4195835 / 3145727) * 3145727 - 4195835
```

If the answer is zero, your computer is just fine. If you get anything else, you have an old Intel Pentium CPU with a floating-point division buga software bug burned into a computer chip and reproduced over and over in the manufacturing process.

On October 30, 1994, Dr. Thomas R. Nicely of Lynchburg (Virginia) College traced an unexpected result from one of his experiments to an incorrect answer by a division problem solved on his Pentium PC. He posted his find on the Internet and soon afterward a firestorm erupted as numerous other people duplicated his problem and found additional situations that resulted in wrong answers. Fortunately, these cases were rare and resulted in wrong answers only for extremely math-intensive, scientific, and engineering calculations. Most people would never encounter them doing their taxes or running their businesses.

What makes this story notable isn't the bug, but the way Intel handled the situation:

- Their software test engineers had found the problem while performing their own tests before the chip was released. Intel's management decided that the problem wasn't severe enough or likely enough to warrant fixing it or even publicizing it.

- Once the bug was found, Intel attempted to diminish its perceived severity through press releases and public statements.
- When pressured, Intel offered to replace the faulty chips, but only if a user could prove that he was affected by the bug.

There was a public outcry. Internet newsgroups were jammed with irate customers demanding that Intel fix the problem. News stories painted the company as uncaring and incredulous. In the end, Intel apologized for the way it handled the bug and took a charge of more than $400 million to cover the costs of replacing bad chips. Intel now reports known problems on its website and carefully monitors customer feedback on Internet newsgroups.

NOTE

On August 28th, 2000, shortly before the first edition of this book went to press, Intel announced a recall of all the 1.13MHz Pentium III processors it had shipped after the chip had been in production for a month. A problem was discovered with the execution of certain instructions that could cause running applications to freeze. Computer manufacturers were creating plans for recalling the PCs already in customers' hands and calculating the costs of replacing the defective chips. As the baseball legend Yogi Berra once said, "This is like déjà vu all over again."

3. NASA Mars Polar Lander, 1999

On December 3, 1999, NASA's Mars Polar Lander disappeared during its landing attempt on the Martian surface. A Failure Review Board investigated the failure and determined that the most likely reason for the malfunction was the unexpected setting of a single data bit. Most alarming was why the problem wasn't caught by internal tests.

In theory, the plan for landing was this: As the lander fell to the surface, it was to deploy a parachute to slow its descent. A few seconds after the chute deployed, the probe's three legs were to snap open and latch into position for landing. When the probe was about 1,800 meters from the surface, it was to release the parachute and ignite its landing thrusters to gently lower it the remaining distance to the ground.

To save money, NASA simplified the mechanism for determining when to shut off the thrusters. In lieu of costly radar used on other spacecraft, they put an inexpensive contact switch on the leg's foot that set a bit in the computer commanding it to shut off the fuel. Simply, the engines would burn until the legs "touched down."

Unfortunately, the Failure Review Board discovered in their tests that in most cases when the legs snapped open for landing, a mechanical vibration also

tripped the touch-down switch, setting the fatal bit. It's very probable that, thinking it had landed, the computer turned off the thrusters and the Mars Polar Lander smashed to pieces after falling 1,800 meters to the surface.

The result was catastrophic, but the reason behind it was simple. The lander was tested by multiple teams. One team tested the leg fold-down procedure and another the landing process from that point on. The first team never looked to see if the touch-down bit was setit wasn't their area; the second team always reset the computer, clearing the bit, before it started its testing. Both pieces worked perfectly individually, but not when put together.

4. Patriot Missile Defense System, 1991

The U.S. Patriot missile defense system is a scaled-back version of the Strategic Defense Initiative ("Star Wars") program proposed by President Ronald Reagan. It was first put to use in the Gulf War as a defense for Iraqi Scud missiles. Although there were many news stories touting the success of the system, it did fail to defend against several missiles, including one that killed 28 U.S. soldiers in Dhahran, Saudi Arabia. Analysis found that a software bug was the problem. A small timing error in the system's clock accumulated to the point that after 14 hours, the tracking system was no longer accurate. In the Dhahran attack, the system had been operating for more than 100 hours.

5. The Y2K (Year 2000) Bug, circa 1974

Sometime in the early 1970s a computer programmerlet's suppose his name was Davewas working on a payroll system for his company. The computer he was using had very little memory for storage, forcing him to conserve every last byte he could. Dave was proud that he could pack his programs more tightly than any of his peers. One method he used was to shorten dates from their 4-digit format, such as 1973, to a 2-digit format, such as 73. Because his payroll program relied heavily on date processing, Dave could save lots of expensive memory space. He briefly considered the problems that might occur when the current year hit 2000 and his program began doing computations on years such as 00 and 01. He knew there would be problems but decided that his program would surely be replaced or updated in 25 years and his immediate tasks were more important than planning for something that far out in time. After all, he had a deadline to meet. In 1995, Dave's program was still being used, Dave was retired, and no one knew how to get into the program to check if it was Y2K compliant, let alone how to fix it.

It's estimated that several hundred billion dollars were spent, worldwide, to replace or update computer programs such as Dave's, to fix potential Year 2000 failures.

6. Dangerous Viewing Ahead, 2004

On April 1, 1994, a message was posted to several Internet user groups and then quickly circulated as an email that a virus was discovered embedded in several JPEG format pictures available on the Internet. The warning stated that simply opening and viewing the infected pictures would install the virus on your PC. Variations of the warning stated that the virus could damage your monitor and that Sony Trinitron monitors were "particularly susceptible."

Many heeded the warning, purging their systems of JPEG files. Some system administrators even went so far as to block JPEG images from being received via email on the systems.

Eventually people realized that the original message was sent on "April Fools Day" and that the whole event was nothing but a joke taken too far. Experts chimed in that there was no possible way viewing a JPEG image could infect your PC with a virus. After all, a picture is just data, it's not executable program code.

Ten years later, in the fall of 2004, a proof-of-concept virus was created, proving that a JPEG picture could be loaded with a virus that would infect the system used to view it. Software patches were quickly made and updates distributed to prevent such a virus from spreading. However, it may only be a matter of time until a means of transmission, such as an innocuous picture, succeeds in wrecking havoc over the Internet.