Dori-Mic and the Universal Machine!

A Tragicomic Tale of Combinatorics and Computability
for Curious Children of All Ages

Story and Math by **David Evans**
Illustrations by **Kim Dylla**

dori-mic.org

"If only I had this book when I was a young student, I might have done something useful with my life like discover a new complexity class instead of dropping out and wasting my life on flipping pancakes, playing with basic blocks, and eradicating polo."

Gill Bates,
Founder of Mic-Soft Corporation

"I thought the first few pages were okay, but I was never able to get past page 14."

Kurt Friedrich Stöpdel,
incomplete logician

"The BEST babies' book about computational universality I've read. I love the concept, and the illustrations are adorable!"

Scott J. Aaronson,
Complexity Zookeeper

"I recommend this book to all my graduate students who need to brush up on their computing theory for their PhD exams."

Donald E. Kduck,
Mak-Mak Junior University

creative commons Creative Commons Attribution-NonCommercial-ShareAlike
Second Printing, May 2014

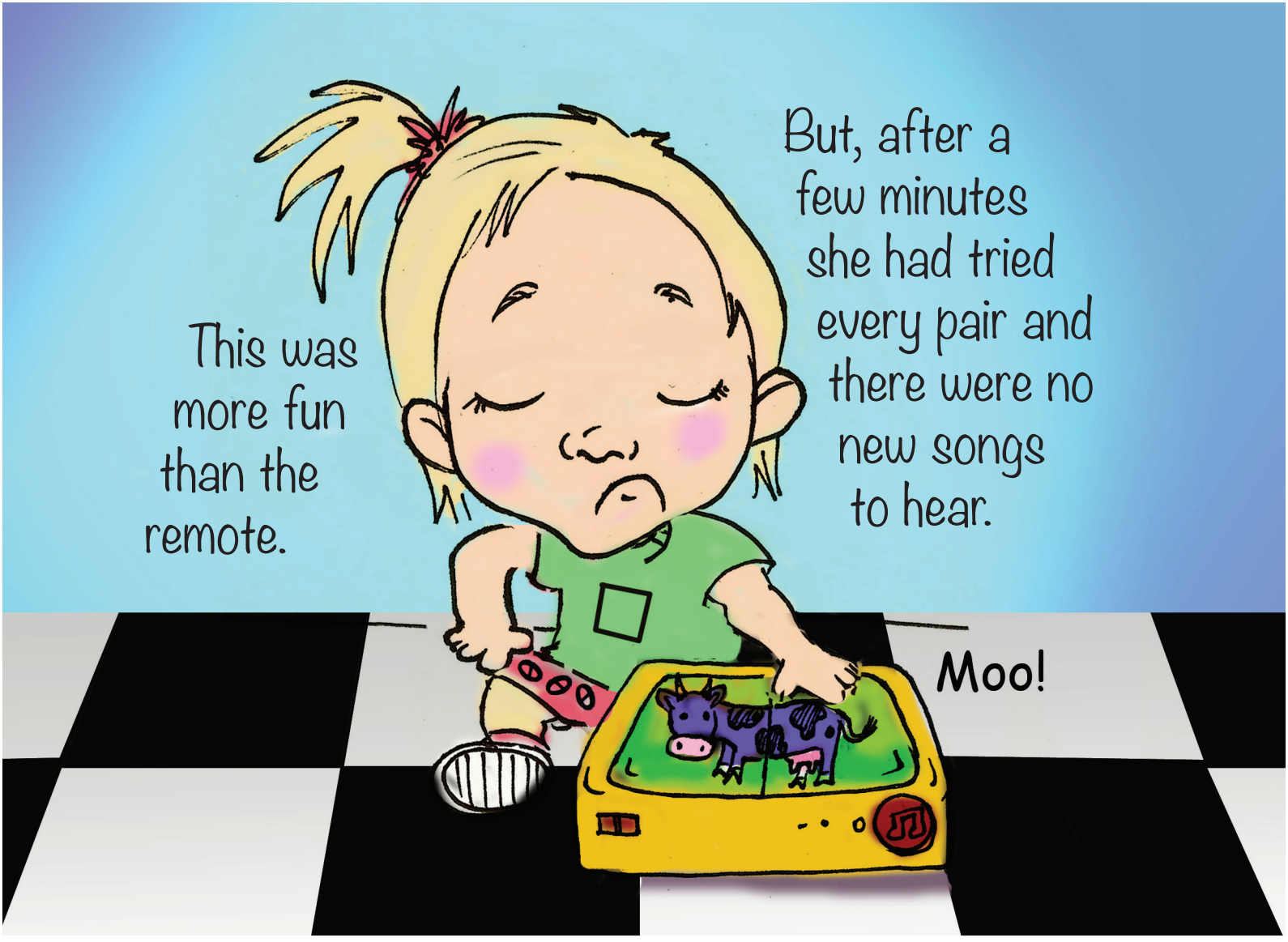

Dori was a very lucky baby.
She had a Mommy and Daddy who loved her very much and many, many toys.

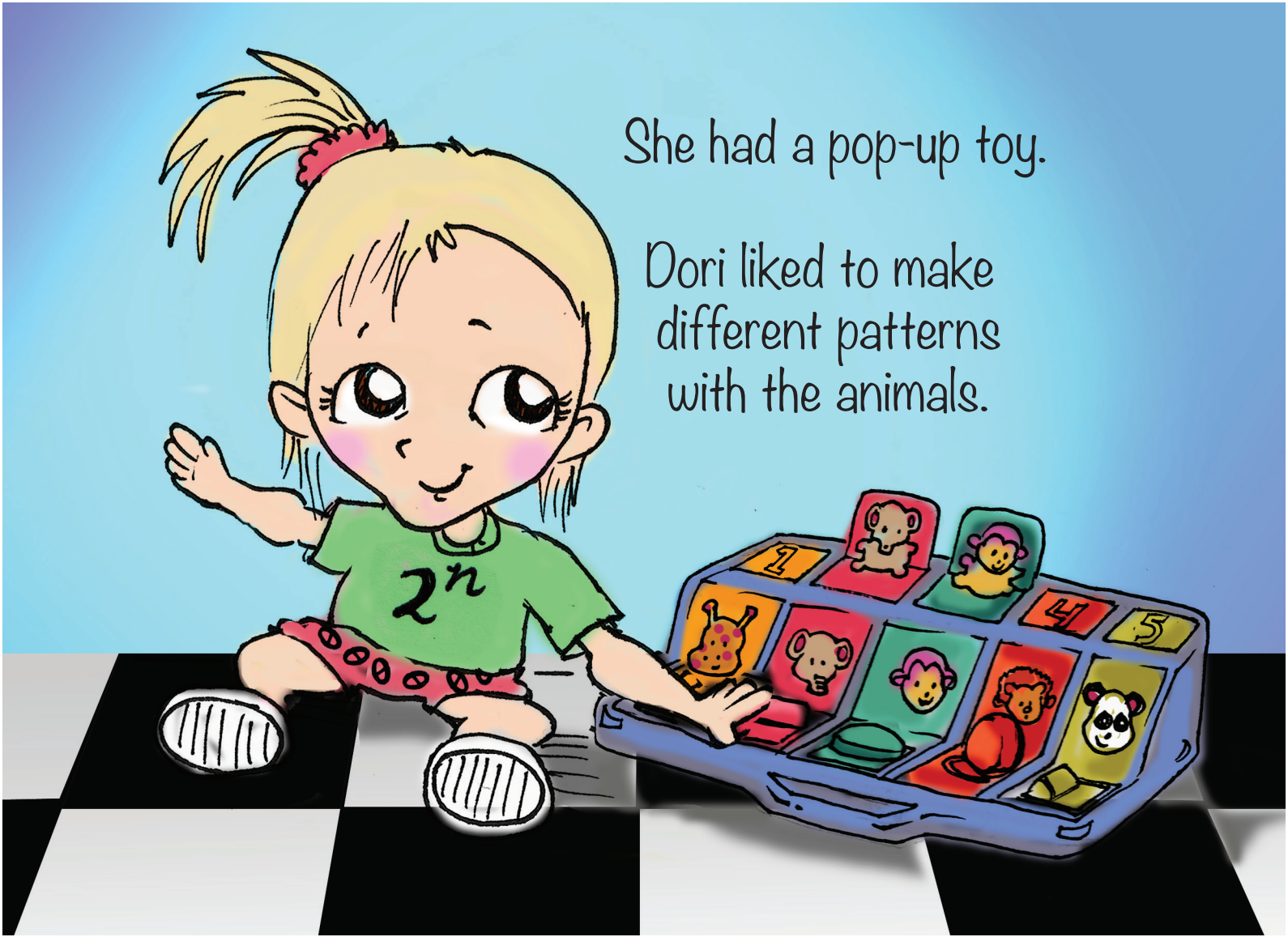
But, she was not
always happy.


She had
a remote
control toy.
n


It was fun for a few minutes, but soon she had tried all the buttons and there was nothing new to do.


She had a farm mash-up toy.
She could put in any two pieces to hear a different song.
"It's a dog-cow!"


This was more fun than the remote.
But, after a few minutes she had tried every pair and there were no new songs to hear.
Moo!

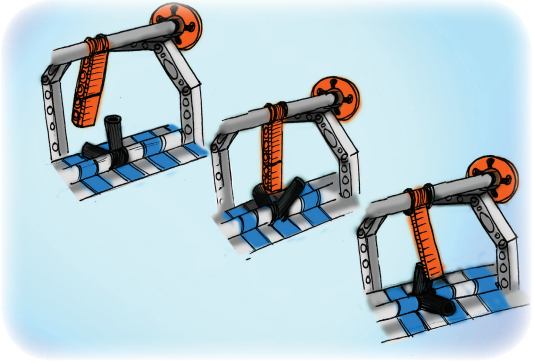

She had a pop-up toy.
Dori liked to make different patterns with the animals.
2^n
1
4
5


But after a few hours, she had made all the patterns and wanted something new to do.
2^n


She had a stacking toy. This was great!
Dori could put the rings in any order, and there were many orders to try!

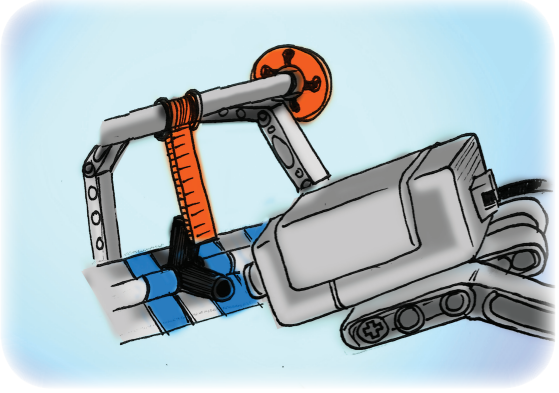

It took a few days to try
them all, but after that
poor Dori was bored.

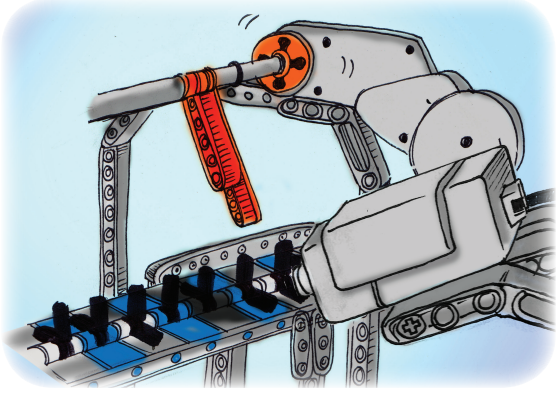

Then, one day
Dori got a very
special toy!



It had a way to remember,

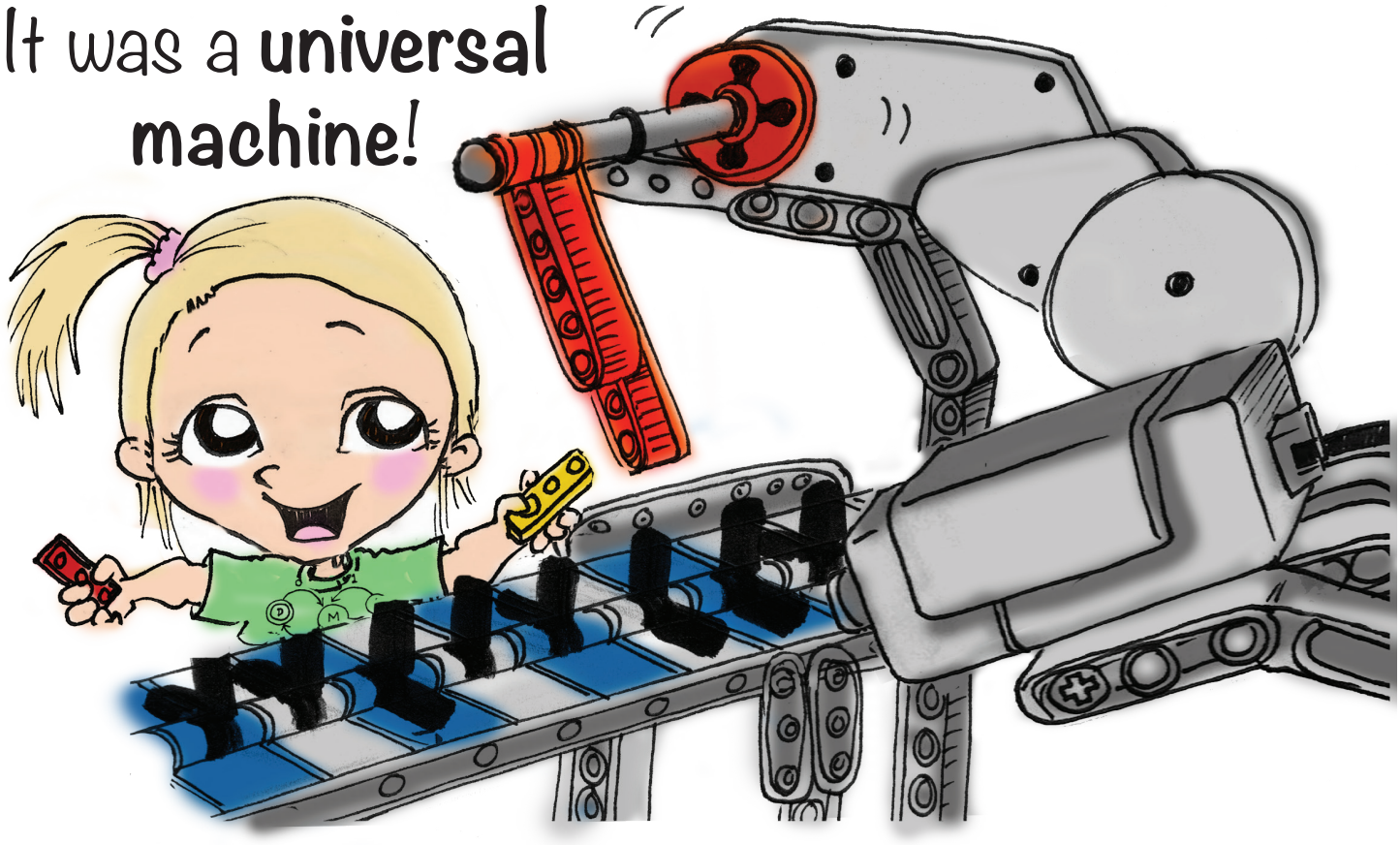
a way to make decisions,





and a way to keep
going and going...

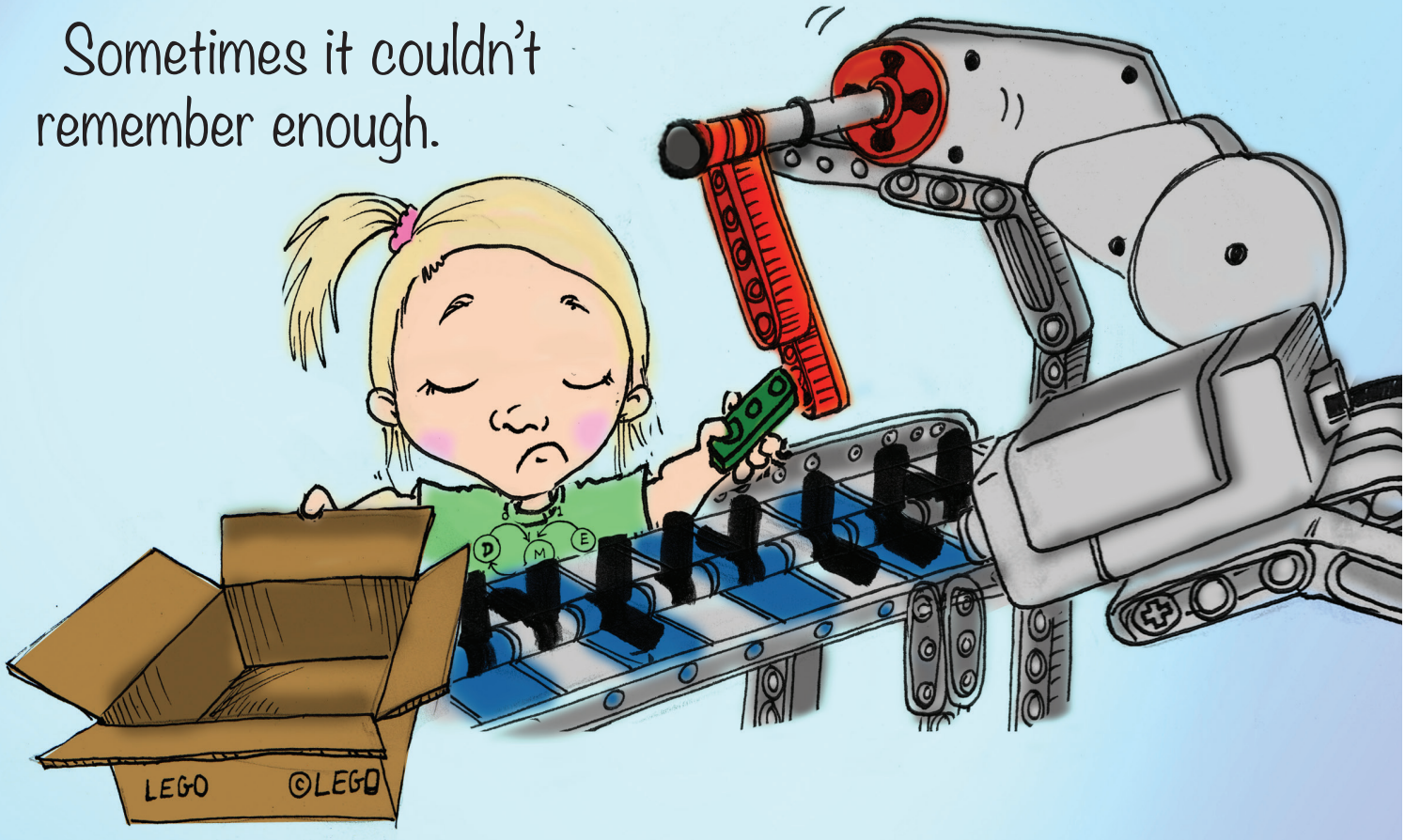
It was a **universal machine!**

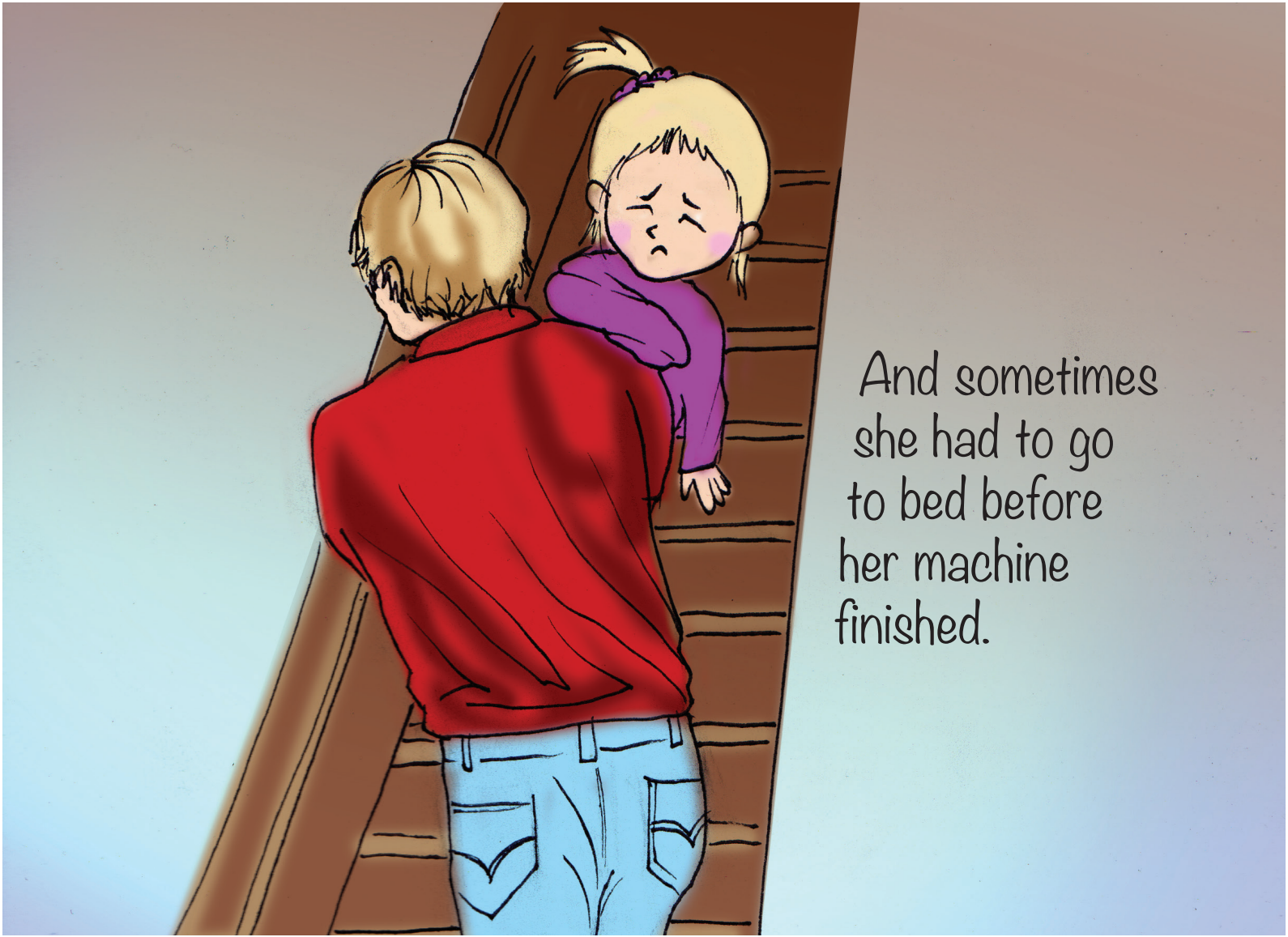


Dori could make it do anything she could imagine!

But, Dori's machine was not ideal.

Sometimes it couldn't remember enough.




And sometimes she had to go to bed before her machine finished.


But, the next day
she woke up and
got more bricks and
more time to play!
LEGO

Dori was very happy she would never run out of new things to do!

Notes

The **remote control toy** has 16 buttons to try. Adding more buttons will only make it a little bit more interesting. With are N buttons, there are N things to try.

The **farm mash-up toy** has five different animals. There are five choices for the front of the animal, and five choices for the back. This makes $5 \times 5 = 25$ different pairs to try. Adding one more animal adds lots more possibilities! We can try the new animal front with each of the old animals (as well as with its own back), and its back with each of the old animals also. With N animals, there are N *squared* ($N \times N$) pairs to try.

The **pop-up toy** has five different animals to pop-up. Each animal can be either up or down to make many different patterns. For example, there is one pattern where all the animals are down, one pattern where all the animals are up, and five patterns where just one animal is up. The easiest way to count all the patterns is to think about each animal being either up or down. We have two ways to set the first animal, and two ways to set the second, and so on. We can combine all the ways to set the first with all the ways to set all the other animals. This makes $2 \times 2 \times 2 \times 2 \times 2 = 32$ possible patterns. Multiplying a number by itself is called *exponentiation*. We can write $2 \times 2 \times 2 \times 2 \times 2$ as 2^5. Adding one more pop-up animal doubles the number of patterns since we can set the new animal both ways and combine it with all the ways to set all the other animals. With N pop-up animals, there are 2^N different patterns.

tttt

The **stacking toy** has five rings that can be arranged in any order. This means there are 5 choices for the first ring, 4 choices for the second ring, 3 choices for the third ring, 2 choices for the fourth ring, and only one ring left to use on top. The total number of orderings is $5 \times 4 \times 3 \times 2 \times 1 = 120$. This is called *factorial* and written as $5!$ With N rings, the number of orderings is $N!$ which grows very quickly! A 10-ring peg toy would have over 3 million orders to try. Dori would take six years to try them all if she can try one each minute and never goes to sleep.

A **universal machine** is a machine powerful enough to simulate every other machine. Alan Turing showed that a machine with a few very simple operations could simulate everything a person could do by following mechanical rules and using an unlimited amount of scratch paper. All you need is a way to remember (by putting the legos in one position or the other), a way to make decisions (reading the position of the current lego and taking a step based on what it is), and a way to keep going (keep making steps until the machine gets stuck). People have built such machines using legos (as well as with simple substitution rules, electronic circuits, DNA, and many other things).

Turing showed how to construct a machine that could simulate every other machine: **a universal machine!** Our universal machine is a mathematical abstraction. No real machine can be truly universal, since an ideal machine has an infinite amount of memory and can execute an unlimited number of steps. But, we can get pretty close by getting more memory when we run out, and by executing billions of instructions every second on a modern computer.

About Dori-Mic

Dorina Michelle (a.k.a., "Dori-Mic", which is pronounced "Dori-Meek" and means "Little Dori" in Dori's mother's mother tongue) lives with her parents in Charlottesville, Virginia. She is well-known for her previous work on recursive definitions, but this is her first book on computability theory. Dori's programs do not always terminate, but she blames her daddy for any mathematical errors in this book since she was not yet able to read when it was published.