NUMB3RS Activity: Turing Origami
Episode: "One Hour"

Topic: Computer Science, Logic **Grade Level:** 9 - 10
Objective: Learn to use a Turing Machine.
Time: 20 - 30 minutes
Materials: TI-83 Plus/TI-84 Plus graphing calculator

Introduction

In "One Hour," Amita and Charlie are quietly working on their own projects when Charlie, seeing a 2-D Turing Machine on Amita's laptop, comments, "I haven't seen an inductive Turing machine used quite like that before." Amita replies that she is "trying to figure out the finite state machines." Turing machines can be used to generate sequences. In this activity, students will determine the fold sequence generated by a Turing machine.

Discuss with Students

The activity includes a paper-folding activity. This can either be done individually by students or as a class demonstration. Emphasize to the students that all of the folds should be from right to left, and when we examine the sequence, we are reading the folds left to right. There are two types of folds: a valley fold and a mountain fold. (See student page 2 for a diagram showing both types of folds.) For notation, we use a V to represent a valley fold and a Λ to represent a mountain fold.

The purpose of the paper folding is to have a meaningful sequence to analyze, but the goal is to determine the fold sequence using the Turing machine. The algorithm for this particular sequence is to examine symbols one at a time and, based upon rules that students will identify, add two more symbols to the end of the sequence. This algorithm is known as a *Turing machine*, and the rules that determine which symbols should be attached to the end of the sequence are known as *finite state machines*. The symbol that is being read is the "state" of the Turing machine.

Below is a step-by-step solution of the fourth fold sequence generated in the activity.

Start with an initial number of 1.
First digit: state is 1, so applying the rules, add 3, 2
1, 3, 2
Next state is 3, so applying the rule, add 3, 1
1, **3**, 2, 3, 1
Next state is 2, so applying the rule, add 4, 2
1, 3, **2**, 3, 1, 4, 2
Next state is 3, so applying the rule, add 3, 1
1, 3, 2, **3**, 1, 4, 2, 3, 1
Next state is 1, so applying the rule, add 3, 2
1, 3, 2, 3, **1**, 4, 2, 3, 1, 3, 2
Next state is 4, so applying the rule, add 4, 1
1, 3, 2, 3, 1, **4**, 2, 3, 1, 3, 2, 4, 1
Next state is 2, so applying the rule add 4, 2
1, 3, 2, 3, 1, 4, **2**, 3, 1, 3, 2, 4, 1, 4, 2

Brock Wenciker, Shawnee Mission East High School, Overland Park, KS

When the students generate the Turing sequence, they need to take care that they examine one state, or number, at a time. Students may wish to mark which state they are on using checkmarks or underlines.

The extension leads the students to determine the summation that produces the number of folds for any folding. Depending on the level of your students, you may wish to review the summation sign and its use. The section of program provided in the extension is part of a larger program written specifically for this activity. The complete program file can be downloaded for free by going to **http://education.ti.com/exchange** and searching for "7900."

Student Page Answers:
1. *V V, V Λ, Λ V, and Λ Λ* **2.** *132314231324142* **3.** *VVΛVVΛΛVVVΛΛVΛΛ*
4. *Students will fold paper.* **5.** *1323142313241423132314241324142*
6. *VVΛVVΛΛVVVΛΛVΛΛVVVΛVVΛΛΛVVΛΛVΛΛ*

Extension Page Answers:
1. *(1, 1), (2, 3), (3, 7), (4, 15), (5, 31), (6, 63), (7, 127)* **2.** *2, 4, 8, 16, 32, 64* **3.** 2^1, 2^2, 2^3, 2^4, 2^5, 2^6
4. $\sum_{i=1}^{n} 2^i$
5.

```
:ClrHome
:"1"→Str1
:0→C
:Disp "ENTER THE NUMBER"
:Input "OF FOLDS ",N
:sum(seq(2^(X-1),X,1,N)→F      Sequence for the number of folds
:While length(Str1)<F
:C+1→C
:If sub(Str1,C,1)="1"
:Str1+"32"→Str1
:If sub(Str1,C,1)="2"
:Str1+"42"→Str1                Finite Machines from activity
:If sub(Str1,C,1)="3"
:Str1+"31"→Str1
:If sub(Str1,C,1)="4"
:Str1+"41"→Str1
:End
:ClrHome
:Disp "TURING SEQUENCE"
:Disp Str1
```


Brock Wenciker, Shawnee Mission East High School, Overland Park, KS

Name: _____ Date: _____

NUMB3RS Activity: Turing Origami

In "One Hour," Amita and Charlie are quietly working on their own projects when Charlie, seeing a 2-D Turing Machine on Amita's laptop, comments, "I haven't seen an inductive Turing machine used quite like that before." Amita replies that she is "trying to figure out the finite state machines." In this activity, you will determine the fold sequence generated by a Turing machine.

Before digital computers were invented, British mathematician Alan Turing proposed a thought experiment in which complex tasks could be accomplished by a series of conditional statements, which he called *finite state machines*. This thought experiment laid the foundation for computer science and many of the concepts used in computer programming. Today, we often think of these finite state machines as lines of computer code, in which each line completes some specific operation. Afterward, the program proceeds to the next line of code. Turing machines can be used to generate sequences. In Turing's thought experiment, these finite state machines can be either mechanical contraptions or people doing specific tasks.

To illustrate the process, imagine you enter the FBI office and Amita, Charlie, and Don are sitting at separate desks. Your task is to collect a series of signatures from people in the order that their names appear on a sheet of paper. The problem is that every time you get a signature from someone, that person adds two more names to the end of the list. A person cannot, however, add names to the list arbitrarily; the person must follow these rules:

- Amita adds Don and Charlie
- Charlie adds Don and Amita
- Don adds Megan and Amita

Your paper starts out with Amita's name, so you visit her desk first. What happens can be represented symbolically with the letters A, C, D, and M.

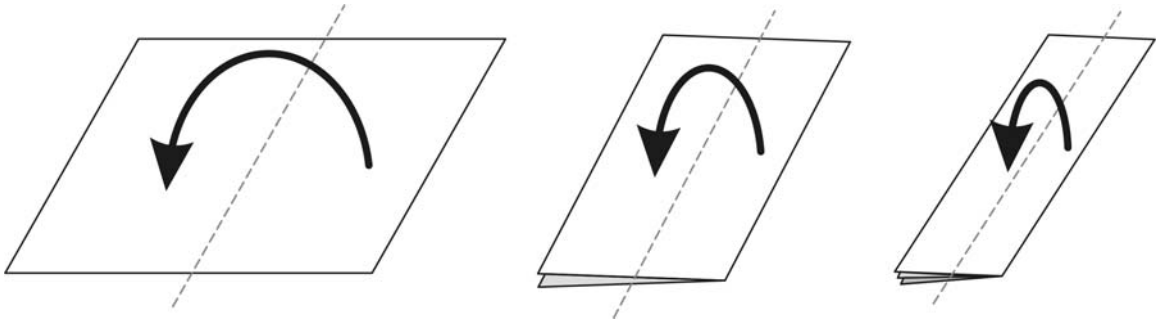
Description	Symbolic
You start with Amita, and she adds the names Don and Charlie to your list.	A adds DC to yield **A**DC
Next you go to Don, who adds Megan and Amita to your list.	A**D**CMA
Next you go to Charlie, who adds Don and Amita to your list.	AD**C**MADA
Next you are supposed to go to Megan, but she is not in the office, so you can stop.	STOP

The process of generating this list is the *Turing machine*; the rules specifying the names to be added are the *finite state machines*; and the desk you are currently visiting is the "state" of the Turing machine.

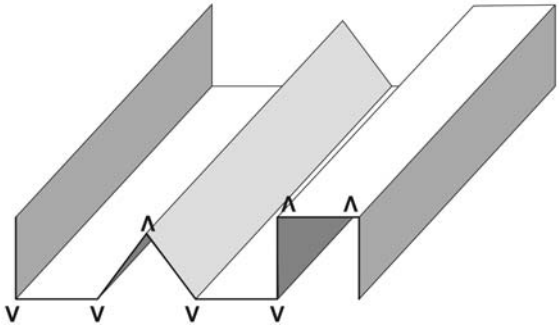
Brock Wenciker, Shawnee Mission East High School, Overland Park, KS

This is only one possible configuration. Other Turing machines may have different numbers of names that can be added; names could be erased; or any other variation that can possibly be imagined can be generated. All Turing machines have conditional statements (or the “finite state machines”).

If you repeatedly fold a piece of paper in half from right to left, you will create a series of seemingly random folds.



For instance, if you make the three folds shown above and then unfold the paper, the result will be the series of valley (**V**) and mountain (**Λ**) folds shown below:



The resulting sequence of folds is:

V V Λ V V Λ Λ

It is difficult to fold a piece of paper more than four times from right to left, but there is a Turing machine that will generate the sequence of folds using a series of finite state machines. This Turing machine, like our earlier example involving names, looks at one symbol at a time and, based on the finite state machines, adds two symbols to the end of the list.

1. Because our Turing machine adds two symbols to the end, what are the four possible combinations that can be added using the symbols V and Λ?

Because we read one of **two** symbols (i.e., “examine the state”), then add one of **four** combinations, we encounter a problem. We can see how this problem affects our sequence in the first several iterations:

Brock Wenciker, Shawnee Mission East High School, Overland Park, KS

Starting with an initial V, the state is V, and the pair combination to be added is V Λ.

Examining the second symbol V V Λ, the state is again V, but the pair combination to be added is V V.

The problem is in the fact that one finite state machine must be **"When the state is V, add V Λ to the end of the list,"** while another finite state machine is **"When the state is V, add V V to the end of the list."**

To overcome this contradiction, we need two variations for each symbol. One solution is to translate the symbols into numbers, in which the odd numbers 1 and 3 represent the two variations of V, and the even numbers 2 and 4 represent the two variations of Λ.

Using these numbers to represent the variations of symbols, we have the following rules:

- When the state is 1, add 3, 2 to the end of the list, then move one space to the right.
- When the state is 2, add 4, 2 to the end of the list, then move one space to the right.
- When the state is 3, add 3, 1 to the end of the list, then move one space to the right.
- When the state is 4, add 4, 1 to the end of the list, then move one space to the right.

With these finite state machines in place, the paper-folding sequence is generated as follows:

Start with an initial number of 1.

The first state is 1, so add 3, 2 to the end of the list: **1**, 3, 2

The next state is 3, so add 3, 1 to the end of the list: 1, **3**, 2, 3, 1

2. Continue until you have generated all 15 characters for the fourth fold sequence.

The final step is to translate this Turing sequence back into the paper-folding sequence with symbols of V and Λ.

Brock Wenciker, Shawnee Mission East High School, Overland Park, KS

3. Translate the sequence from numbers back into the fold sequence. (That is, replace each number with a V or Λ, as appropriate.)

4. Fold a piece of paper four times, and compare the results to the Turing sequence you created.

5. Using the same Turing machine, generate the 31 characters of the fifth fold sequence.

6. Translate the sequence from numbers back into the fold sequence.

Folding the piece of paper to check fold sequences becomes challenging (and impossible) as you increase the number of times the paper is folded. The Turing machine, however, can be used to generate the sequence for any theoretical number of folds.

Brock Wenciker, Shawnee Mission East High School, Overland Park, KS

The goal of this activity is to give your students a short and simple snapshot into a very extensive mathematical topic. TI and NCTM encourage you and your students to learn more about this topic using the extensions provided below and through your own independent research.

Extensions

1. Complete the table at the right to compare the number of foldings to the number of folds.

2. Find the differences between the numbers of folds.

3. Write these differences as powers of the same base.

Foldings	Number of Folds
1	1
2	3
3	7
4	
5	
6	
7	

4. Express these differences as a summation.

$$\sum_{i=1}^{n} \text{———}$$

Below is a section of a program for a graphing calculator that generates the Turing sequence and folding sequence for any number of folds. Identify the summation just described as well as the Finite Machines found in the activity.

```
:ClrHome
:"1"→Str1
:0→C
:Disp "ENTER THE NUMBER"
:Input "OF FOLDS ",N
:sum(seq(2^(X-1),X,1,N)→F
:While length(Str1)<F
:C+1→C
:If sub(Str1,C,1)="1"
:Str1+"32"→Str1
:If sub(Str1,C,1)="2"
:Str1+"42"→Str1
:If sub(Str1,C,1)="3"
:Str1+"31"→Str1
:If sub(Str1,C,1)="4"
:Str1+"41"→Str1
:End
:ClrHome
:Disp "TURING SEQUENCE"
:Disp Str1
```


Brock Wenciker, Shawnee Mission East High School, Overland Park, KS