The Towers of Hanoi

Introduction

The Towers of Hanoi is a puzzle that has been studied by mathematicians and computer scientists alike for many years. It was popularized by the western mathematician Edouard Lucas in 1883.

The puzzle originates with a legend. This legend comes in various forms, so you may encounter a slightly different version if you were to research the Towers of Hanoi puzzle. The legend states that there is a secret room in a temple containing three large posts. The first post contains 64 golden disks stacked upon it, starting with the largest disk on the bottom, and decreasing in size to the smallest disk on top of the stack. Monks (or priests) have been been moving the disks continuously since the beginning of time. The goal is to recreate the 64 disk tower on the third post. The monks must move the disks according to two rules:

1. The monks can only move one disk at a time.
2. The monks can only place smaller disks on top of larger disks.

Once the priests make the final move and complete the puzzle, the world will end.

How long until the world ends?

Activity 1: Hanoi Races

Your first task is to complete the puzzle for 6 disks. Work with your group to complete the puzzle faster than the other groups.

Note here any strategies you used or other ideas that you found helpful in solving the puzzle. You will want to refer back to these later.

Activity 2: Optimal Strategies

It is hard to get a good estimate on how long it would take the monks to finish the puzzle, especially if these monks are not very good with mathematics. They will probably make mistakes, and thus fail to complete the puzzle in the most efficient way. In order to see how long it will take for the world to end, we need to figure out an efficient and systematic way to complete the puzzle. We'd like to find an *optimal* solution. In this case, our optimal solution would be the minimum number of moves required to solve the puzzle.

An *algorithm* is a set of rules that precisely defines a set of operations or calculations. A carefully described algorithm will allow us to consistently solve the puzzle in the most optimal manner.

Work with your group to minimize the moves you need to make to solve the puzzle. Complete the chart below for $n = 1$, 2, 3, and 4 disks.

number of disks n	minimum number of moves
1	
2	
3	
4	

Once you and your team think you know the minimum number of moves to solve the puzzle, create an algorithm that explains how to solve the puzzle for the $n = 4$ box (disk) problem. Use the space on the next page. You must be clear, concise, and complete in your algorithm description. The algorithm must explain how to start, when to finish, and what to do at each step.

Algorithm Development

Activity 3: A Step-By-Step Strategy

Binary Numbers

To help us understand some strategies of the Towers of Hanoi puzzle, we first need to explore the *Binary Number System*.

We are familiar working with numbers in base-10 notation. We use 10 digits (0-9) to form the numbers in our system. Each place value of a number represents a power of 10. For example, 125 is the same as $(1 \times 10^2) + (2 \times 10^1) + (5 \times 10^0)$. Computers use a base-2 number system, or a Binary number system. This means that only the digits 0 and 1 are used to represent numbers, and each place value of a number represents a power of 2. For example, the binary number 101 is the same as $(1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) = 5$. We can easily convert back and forth between base-10 and base-2 representations of numbers using some simple steps.

- **Base-2 $\rightarrow$ Base-10**
  We read the place values from right to left, as increasing powers of 2. We evaluate 11001 as $(1 \times 2^0) + (0 \times 2^1) + (0 \times 2^2) + (1 \times 2^3) + (1 \times 2^4) = 1 + 8 + 16 = 25.$

- **Base-10 $\rightarrow$ Base-2**
  We start with the highest power of 2 less than our number, and mark a 1 in the space if our number contains that power of 2, and mark a 0 if it does not.

  For example, let us find the base-2 representation of 13.

  – First, we find the largest power of 2 that is less than 13. This is 3, since $2^3 = 8$ and $2^4 = 16$. So, we will have a 1 in the 2^3-place. Our base-2 expansion will look like $1\_\_\ \_\_\ \_\_$.
  – We subtract 2^3 from our original value of 13, and we have 5 leftover. The next smallest power of 2 is $2^2 = 4$. This is less than 5, so we have enough space for 2^2. We put a 1 in the 2^2-place, and our expansion is now $11\_\_\ \_\_$.
  – We subtract 2^2 from 5, and now we are left with 1. The next smallest power of 2 is $2^1 = 2$, and this is greater than what we have left, so we don't have room for 2^1. Thus, we put a 0 in that spot, and our base-2 expansion is now $110\_\_$.
  – We have 1 left, which is precisely 2^0. Thus we put a 1 in the last spot and we are done. 13 in base-10 is equivalent to 1101 in base-2, or $13_{10} = 1101_2$.

Practice converting the following base-10 numbers to their binary equivalent:

1. 8

2. 15

3. 26

4. 37

Puzzle Strategy

We can look at the Towers of Hanoi puzzle from a “binary" perspective in many ways. Each time we need to move a disk, we have to choose between two options. For each turn, we can think of each disk as either being “moved" or “not moved" (i.e. “1" or “0"). You may have also noticed that whether or not a disk is “odd" or “even" (or “blue" or “"red") can also help in solving the puzzle. In this next section, we will use the Binary number system to tell us which moves to make in order to efficiently solve the puzzle.

We define k to be the step or move number. Each step, we first write out k in its binary form. Starting from the right, we associate each place values with a disk numbers, starting with D1 representing the smallest (topmost) disk.

	2^3	2^2	2^1	2^0
k	D4	D3	D2	D1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0

$\vdots$

We interpret the moves as follows: For each step k, the algorithm is as follows:

1. Locate the rightmost 1 in the binary expansion. We'll call the corresponding disk the source disk.
   (a) If there are no other 1's in the binary expansion, then we move the source disk to an empty space. (Ex: 010 means that we move Disk 2 to an empty space).
   (b) On the first move, you have two empty spaces. This choice is important. Use your notes from the previous activities to help here.
2. Locate the second 1 from the right, and call the corresponding disk the location disk.
   (a) If there are NO zero's *OR* an EVEN number of zero's between the source and location, then we move the source disk on top of the location disk. (Ex: 011 means move disk 1 on top of disk 2, and 01001 means move disk 1 on top of disk 4).
   (b) If there are an ODD number of zero's between the source and the location, then we move the source disk to the space that DOES NOT contain the location disk. (Ex: 101 means move disk 1 to the space that DOES NOT contain disk 3).

Practice these steps, and fill in the charts below for the $n = 2$, 3, and 4 disk puzzles. Record any patterns you notice or other observations you made while completing the charts. (For example: when is the puzzle complete? Does this match your observations in Activity 2?)

$n = 2$	2^3	2^2	2^1	2^0	
k			D2	D1	move
0	0	0	0	0	
1	0	0	0	1	move D1 to empty
2	0	0	1	0	
3					

$n = 3$	2^3	2^2	2^1	2^0	
k		D3	D2	D1	move
0	0	0	0	0	
1	0	0	0	1	move D1 to empty
2	0	0	1	0	
3					

$n = 4$	2^3	2^2	2^1	2^0	
k	D4	D3	D2	D1	move
0	0	0	0	0	
1	0	0	0	1	move D1 to empty
2	0	0	1	0	
3					

Use the space below for any notes, observations, ideas, etc.

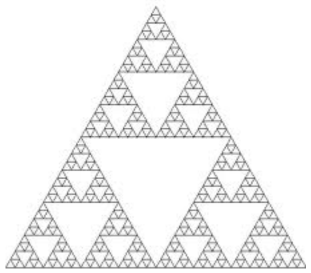
Activity 4: Recursion: See Recursion

We still need to figure out how long it will be until the world ends. Instead of the iterative, step-by-step method of solving the puzzle, let's look at the Towers of Hanoi from yet another perspective.

We can represent a sequence of numbers as a function f whose domain is the natural numbers. For $n = 0, 1, 2, 3\ldots$ our sequence can be defined as $f : \mathbb{N} \to \mathbb{R}$. We denote the value of the function as $f(n)$ or f_n.

For example, the Fibbonacci Sequence is the sequence of numbers 1, 1, 2, 3, 5, 8, 13 ... where each number in the sequence is the sum of the two previous numbers. In function notation, we have $f(0) = 1$, $f(1) = 1$, $f(2) = 2$, $f(3) = 3$, $f(4) = 5$, ... We know that $f(5) = 8$, and also that $f(5) = f(4) + f(3)$, since $8 = 5 + 3$.

We can define this function or sequence *recursively*. Generally, recursion is the process of repeating something in a self-similar way. The pictures below represent the idea of recursion, or a self-repeating pattern.



Mathematically, recursive functions or sequences are those in which successive values are defined in terms of previous function values, or sequence terms. New terms of the sequence depend on older terms. Each time we evaluate the function f, we recursively evaluate the same function f, at previous points.
In order for our function to be recursive, we need two parts:

1. A base case, or a starting point. Depending on the problem, we may not include 0 in the domain.

   Ex: If $n = 0$ or $n = 1$, then $f(n) = 1$.

2. A recursive relation, or how we define the successive terms of the sequence. These rules should reduce our problem down to the base case(s).

   Ex: If $n > 1$, then $f(n) = f(n-1) + f(n-2)$

Problems

1. Evaluate $f(4)$ of the Fibonacci sequence recursively.

2. Write the function $g(n) = n!$ recursively. Evaluate $g(5)$.

3. Use your notes from Activities 1, 2, and 3 together with the puzzle to represent the Towers of Hanoi puzzle recursively. Let n be the number of disks, and let $f(n)$ be the minimum number of moves it takes to complete an n-disk puzzle. Construct a recursive formula for $f(n)$.

Activity 5: The End of the World

How long until the world ends? First we need to know how many steps are required to solve the puzzle, so we need to evaluate $f(64)$. Since we only have a recursive formula right now, this means we also need to evaluate $f(63)$, which means we also need to evaluate $f(62)$, etc. This might take a while...

An explicit formula would be more useful for this question. We'd like a formula that only depends on n, the number of disks in the puzzle. We want to be able to plug in n, and output the minimum number of steps needed to solve the n-disk puzzle.

1. Try working backwards from $f(n)$ with the recursive definition to find an explicit formula. You might start by writing out the formula for $f(n)$, and then replacing the occurrence(s) of $f(n-1)$ with its own recurisve definition, and so forth until you notice a pattern. Use your charts and notes from Activities 1-4 to help you find the patterns and determine the explicit formula for $f(n)$. Check your formula against the charts you made in Activities 2 and 3.

2. Once you have your explicit formula for $f(n)$, evaluate it for $n = 64$, to get the minimum number of moves required to complete the legendary puzzle. Assume that each move takes 1 second to complete (these are fast-moving monks). How many years until the world ends??