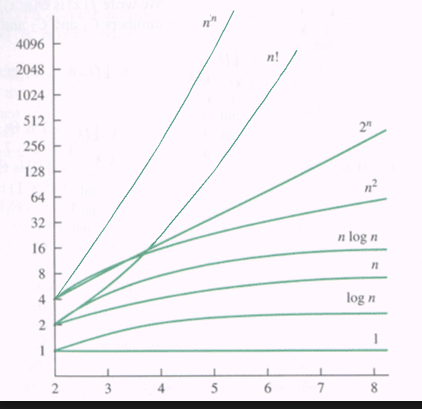
**Tower of Hanoi**

The **Tower of Hanoi**  is a [mathematical game](http://en.wikipedia.org/wiki/Mathematical_game) or [puzzle](http://en.wikipedia.org/wiki/Puzzle). It consists of three rods, and a number of disks of different sizes which can slide onto any rod. The puzzle starts with the disks in a neat stack in ascending order of size on one rod, the smallest at the top.

The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:

1. Only one disk can be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
3. No disk may be placed on top of a smaller disk.

With three disks, the puzzle can be solved in seven moves. The minimum number of moves required to solve a Tower of Hanoi puzzle is 2*n* - 1, where *n* is the number of disks.



BIG O : O(2^n)

1 void tower(int disks, char src, char dest, char aux){

2 if (disks==1){ //No Recursion

3 cout<<"move plate "<<src<<" to "<<dest;

4 }else{

5 tower(disks-1, src, aux, dest);

6 cout<<"move plate from "<<src<<" to "<<dest;

7 tower(disks-1, aux, dest, src);

8 }

9 }

Line 1: Identify the number of disks that are to be moved from the source to the destination using the auxiliary peg

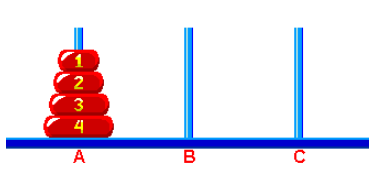
Line 2: If there is only one disk, no recursion is necessary. This is the base case.

Line 5: Move all the items except for the bottom most disk to the auxiliary. This will free up the bottom disk to be moved to the destination.

Line 6: Move the bottom disk, that is now free, to the destination.

Line 7: Move all the disks that was moved in line #5 from the auxiliary to the destination.

Summary: For each function call, we want to move the designated number of disks from the source to the destination. This is done through 3 steps. Move all the items except for the bottom disk to the auxiliary position. Then, move the bottom to the destination. Finally move all the items from the auxiliary position on top of the bottom most disk which is now sitting in its destination.



void tower(int disks, char src, char dest, char aux)

1st argument:

The 1st argument represents the number of disks that are to be moved.

For example 4 means that disks 1,2,3, and 4 will be moved. Disk 1 is the top most disk and disk 4 is the bottom most disk.

For example 3 means that disks 1, 2 and 3 will be moved. Disk 1 is the top most disk and 3 is the bottom most disk

2nd argument

This is the source. This is where the disks are sitting currently.

3rd argument

This is the destination. This is where the disks will end up.

4th argument

This refers the auxiliary peg

Example : 4 disks

Number of moves: 2^4 -1=15

Everything in Bold is the function header

Everything else is the body of the function body. In every case, except for the base case,

there is a function call, a move and then another function call. In the base case, there is only a

move. Move always moves the bottom most disk.

**Base case**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 disks are moved | 3 disks are moved | 2 disks are moved | 1 disk is moved |  |
| Box 1  **4,A, C, B**  3, A, B, C  mv 4 to C  3, B,C, A | Box 2  **3, A, B,C**  2, A, C, B  mv 3 to B  2, C, B,A | Box 3  **2, A,C, B**  1, A, B, C  mv 2 to C  1, B, C, A | Box 4  **1, A, B, C**  mv 1 to B | Aux has been left out of the visual on the left  tower(int disks, char src, char dest, char aux){  if (disks==1){ // base case  cout<<"move "<<src<<" to "<<dest;  }else{  tower(disks-1, src, aux, dest);  cout<<"move "<<src<<" to "<<dest;  tower(disks-1, aux, dest, src);  }  } |
| Box 5  **1, B, C, A**  mv 1 to C |
| Box 6  **2, C, B, A**  1, C, A, B  mv 2 to B  1, A, B, C | Box 7  **4, C, A, B**  mv 1 to A |
| Box 8  **4, A, B, C**  mv 1 to B |
| Box 9  **3, B, C, A**  2 B,A, C  mv 3 to C  2, A, C, B | Box 10  **2, B,A, C**  1, B, C, A  mv 2 to A  1, C, A, B | Box 11  **1, B, C, A**  mv 1 to C |
| Box 12  **1, C, A, B**  mv 1 to A |
| Box 13  **2, A,C, B**  1, A, B, C  mv 2 to C  1, B, C, A | Box 14  **1, A, B, C**  mv 1 to B |
| Box 15  **1, B, C, A**  mv 1 to C |

Move # Sequence of moves Identifies the red number within each box

|  |  |  |
| --- | --- | --- |
| 1 | mv 1 to B | 4 |
| 2 | mv 2 to C | 3 |
| 3 | mv 1 to C | 5 |
| 4 | mv 3 to B | 2 |
| 5 | mv 1 to A | 7 |
| 6 | mv 2 to B | 6 |
| 7 | mv 1 to B | 8 |
| 8 | mv 4 to C | 1 |
| 9 | mv 1 to C | 11 |
| 10 | mv 2 to A | 10 |
| 11 | mv 1 to A | 12 |
| 12 | mv 3 to C | 9 |
| 13 | mv 41 to B | 14 |
| 14 | mv 2 to C | 13 |
| 15 | mv 1 to C | 15 |

Everything in Red shows how all the disks (1,2,3 and 4) have been moved to the destination C.