Tower-of-Hanoi

May 18, 2020

0.0.1 Problem Statement

The Tower of Hanoi is a puzzle where we have three rods and `n` unique sized disks. The three rods are - source, destination, and auxiliary as shown in the figure below. Initally, all the `n` disks are present on the source rod. The final objective of the puzzle is to move all disks from the source rod to the destination rod using the auxiliary rod. **However, there are some rules applicable to all rods:** 1. Only one disk can be moved at a time. 2. A disk can be moved only if it is on the top of a rod. 3. No disk can be placed on the top of a smaller disk.

You will be given the number of disks `num_disks` as the input parameter. Write a **recursive function** `tower_of_Hanoi()` that prints the "move" steps in order to move `num_disks` number of disks from Source to Destination using the help of Auxiliary rod.

0.0.2 The Idea

Assume you are writing a function that accepts the following arguments: 1. arg1 - number of disks 2. arg2 - rod A - this rod acts as the source (at the time of calling the function) 2. arg3 - rod B - this rod acts as the auxiliary 2. arg4 - rod C - this rod acts as the destination

Follow the steps below: 1. Given the `num_disks` disks on the source, along with auxiliary and destination rods 2. Check if `num_disks == 1`. This must be the termination condition, therefore use recursion to reach at this moment. - If yes, move disk from source to destination. (Termination condition) 3. For `num_disks > 1`, just think of your FIRST set of steps. You want to pick the bottom most disk on the source, to be transferred to the destination. For this reason, you will will perform the steps below: - Step 1: Move `num_disks - 1` from source to auxiliary - Step 2: Now you are left with only the largest disk at source. Move the only leftover disk from source to destination - Step 3: You had `num_disks - 1` disks available on the auxiliary, as a result of Step 1. Move `num_disks - 1` from auxiliary to destination

---

0.0.3 Exercise - Write the function definition here

```
In [4]: def tower_of_Hanoi(num_disks):
            """
            :param: num_disks - number of disks
            TODO: print the steps required to move all disks from source to destination
            """
            pass
```

Hide Solution

```
In [ ]: # Solution
        def tower_of_Hanoi_soln(num_disks, source, auxiliary, destination):

            if num_disks == 0:
                return

            if num_disks == 1:
                print("{} {}".format(source, destination))
                return

            tower_of_Hanoi_soln(num_disks - 1, source, destination, auxiliary)
            print("{} {}".format(source, destination))
            tower_of_Hanoi_soln(num_disks - 1, auxiliary, source, destination)

        def tower_of_Hanoi(num_disks):
            tower_of_Hanoi_soln(num_disks, 'S', 'A', 'D')
```

Compare your results with the following test cases

- num_disks = 2

```
solution
        S A
        S D
        A D
```

- num_disks = 3

```
solution
        S D
        S A
        D A
        S D
        A S
        A D
        S D
```

- num_disks = 4

```
solution
        S A
        S D
        A D
        S A
        D S
        D A
        S A
```

```
S D
A D
A S
D S
A D
S A
S D
A D
```

In [ ]: