Chapter 9

What is Recursion?

The process in which a function calls itself directly or indirectly is called recursion and the corresponding function is called a recursive function. Using a recursive algorithm, certain problems can be solved quite easily. Examples of such problems are Towers of Hanoi (TOH), Inorder/Preorder/Postorder Tree Traversals, DFS of Graph, etc. A recursive function solves a particular problem by calling a copy of itself and solving smaller subproblems of the original problems. Many more recursive calls can be generated as and when required. It is essential to know that we should provide a certain case in order to terminate this recursion process. So we can say that every time the function calls itself with a simpler version of the original problem.

What is the difference between direct and indirect recursion?

A function fun is called direct recursive if it calls the same function fun. A function fun is called indirect recursive if it calls another function say fun_new and fun_new calls fun directly or indirectly. The difference between direct and indirect recursion has been illustrated in Table 1.

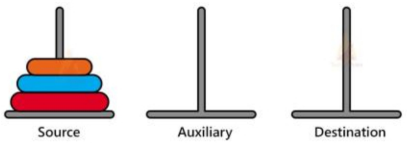
```
// An example of direct recursion
void directRecFun()
{
   // Some code....

   directRecFun();

   // Some code...
}
// An example of indirect recursion
void indirectRecFun1()
{
   // Some code...

   indirectRecFun2();

   // Some code...
}
void indirectRecFun2()
{
   // Some code...

   indirectRecFun1();

   // Some code...
}
```

```
Python code to implement Fibonacci series

# Function for fibonacci
def fib(n):

   # Stop condition
   if (n == 0):
      return 0

   # Stop condition
   if (n == 1 or n == 2):
      return 1

   # Recursion function
   else:
      return (fib(n - 1) + fib(n - 2))
```

```
n = 5;
print("Fibonacci series of 5 numbers is :",end=" ")

# for loop to print the fibonacci series.
for i in range(0,n):
        print(fib(i),end=" ")
```

Tower of Hanoi

The Tower of Hanoi is a mathematical puzzle containing **3 pillars/towers** with **n disks** each of a different size/diameter. These disks can slide onto any pillar. The following diagram shows the initial state of the problem.



Guidelines of Tower of Hanoi Puzzle

The Tower of Hanoi puzzle is cracked by utilising a bunch of directions, and they are as follows:

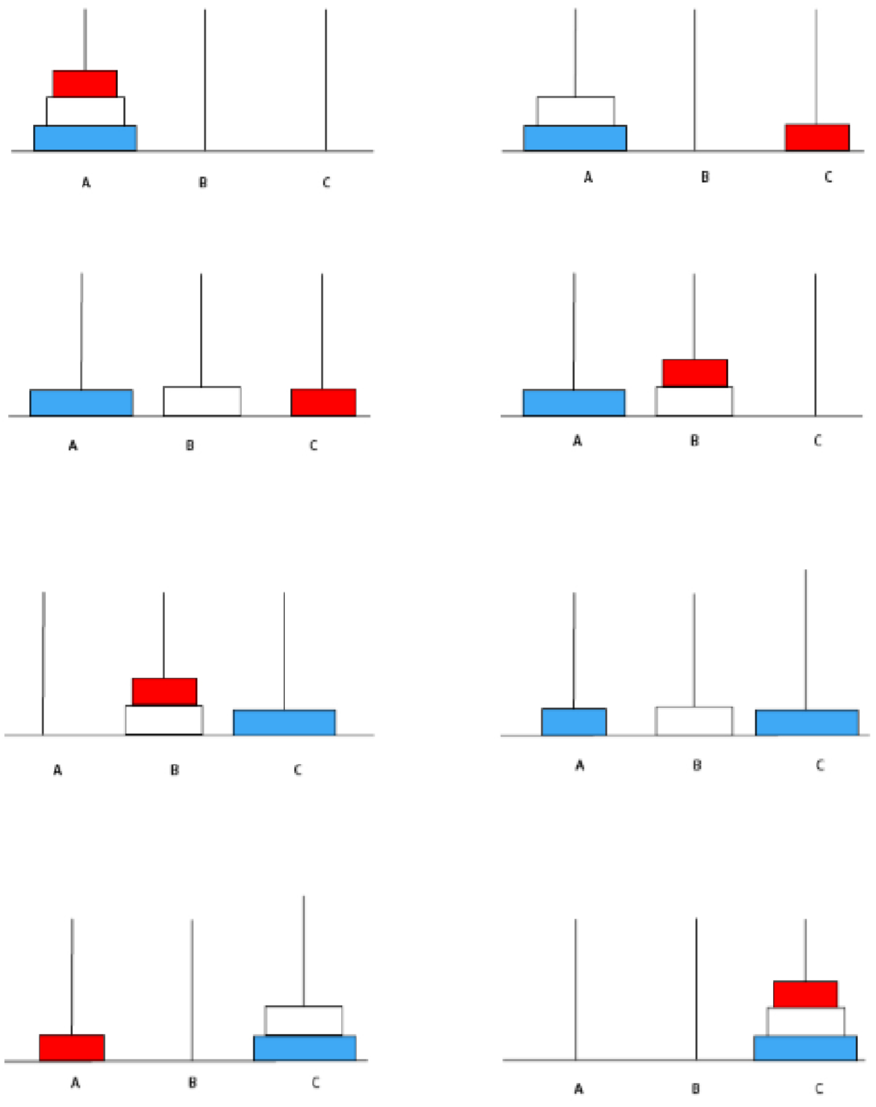
- We are allowed to move only one disc at a time.
- We can only transfer the top disc of one stack to another bar which is empty.
- We cannot place the larger disc on the smaller disc.

The number of alternative moves can be used to estimate the complexity. The least moves required to decode the Tower of Hanoi puzzle with n discs are 2n-1.

Approach:

Tower of Hanoi is a recursion based puzzle and thus, we will follow a recursive approach to solve it. Consider a puzzle with 3 pillars and 3 disks as shown:

Let us we have three disks stacked on a peg



Algorithm

To write an algorithm for Tower of Hanoi, first we need to learn how to solve this problem with lesser amount of disks, say → 1 or 2. We mark three towers with name, **source**, **destination** and **aux** (only to help moving the disks). If we have only one disk, then it can easily be moved from source to destination peg.

If we have 2 disks −

- First, we move the smaller (top) disk to aux peg.
- Then, we move the larger (bottom) disk to destination peg.
- And finally, we move the smaller disk from aux to destination peg.