**RECURRSION**

Recursion is defined as a process which calls itself directly or indirectly and the corresponding function is called a recursive function.

**PROPERTIES OF RECURRSION**

1-The primary property of recursion is the ability to solve a problem by breaking it down into smaller sub-problems, each of which can be solved in the same way.

2-A recursive function must have a base case or stopping criteria to avoid infinite recursion.

3-Recursion involves calling the same function within itself, which leads to a call stack.

4-Recursive functions may be less efficient than iterative solutions in terms of memory and performance.

**TYPES OF RECURRSION**

1-Tail recursion,

2-Head recursion,

3-Tree recursion and

4-Nested recursion.

For further info about the types, check out this link :[TYPES OF RECURRSION](https://www.geeksforgeeks.org/types-of-recursions/)

**MATHEMATICAL INTREPRTATION:**

Used for repetitive computations in which each action is stated in terms of a previous result e.g.: fact(n) = n \* fact (n-1)

**STEPS:**

•For a problem to be written in recursive form, two conditions are to be satisfied

1-It should be possible to express the problem in recursive form, Solution of the problem in terms of solution of the same problem on smaller sized data i.e. try to find sub problem in the problem.

2- The problem statement must include a stopping condition

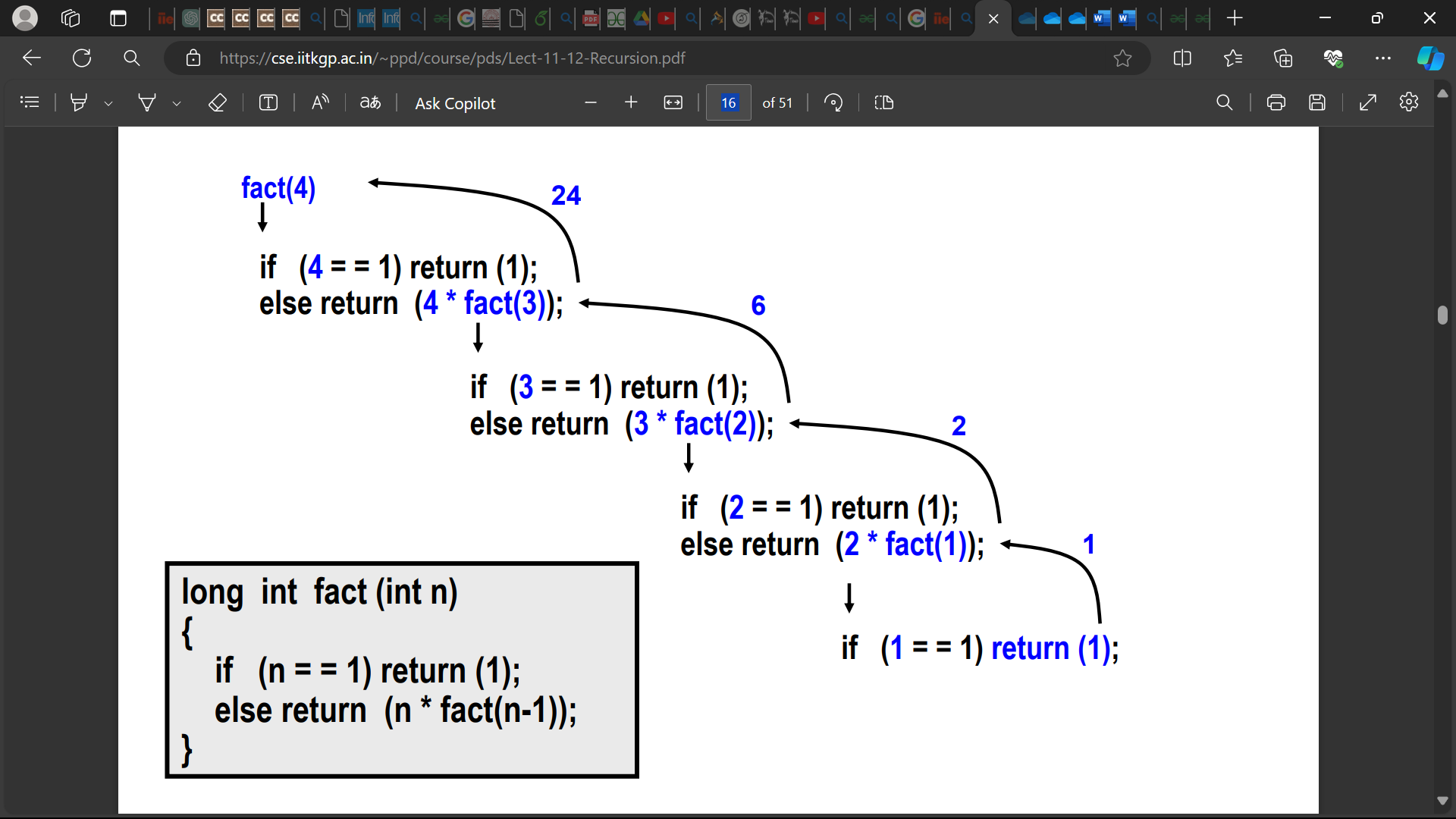
**EXAMPLE:**

Factorial(n) =1 if n=0 --------> **STOPPING CONDITION**

=n\*factorial(n-1) if n>0 --------->**RECURSIVE DEFINITION**

Let's understand it by executing a simple factorial code:

FOR BETTER UNDERSTANDING, LET’S SEE THE STEPS ONE BY ONE.

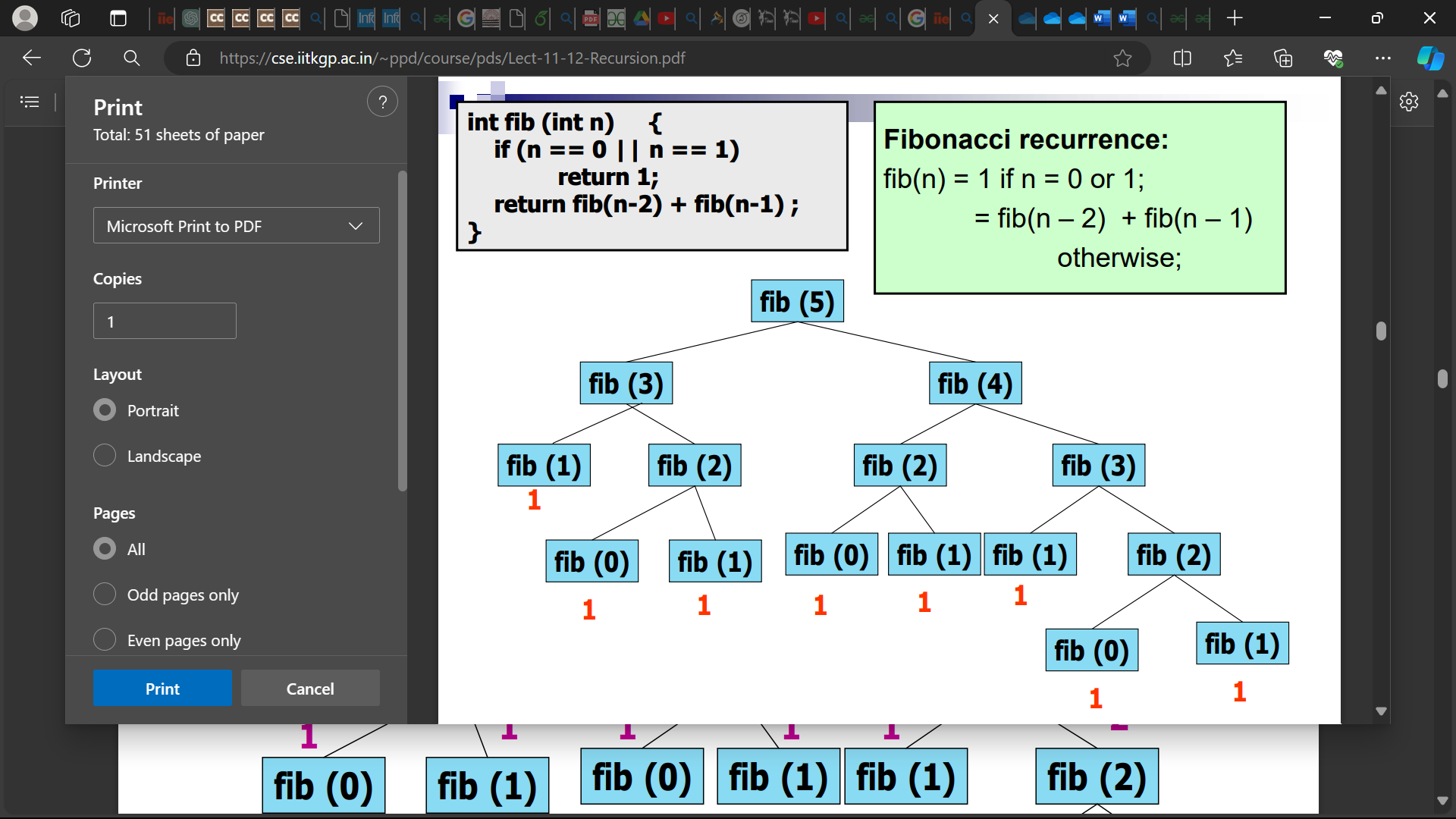


**EXAMPLE 2: FIBONACCI SERIES**

It is a series starting from 0 and 1 and the next element is the sum of the previous two elements.

By reading this statement, you should get the idea of the recurrence definition of the program

i.e



By the above two problems, you must have got the idea of recursion. for some recursion problem there is iterative problem possible.

**DRAWBACK OF RECURRSION OVER ITERATION**

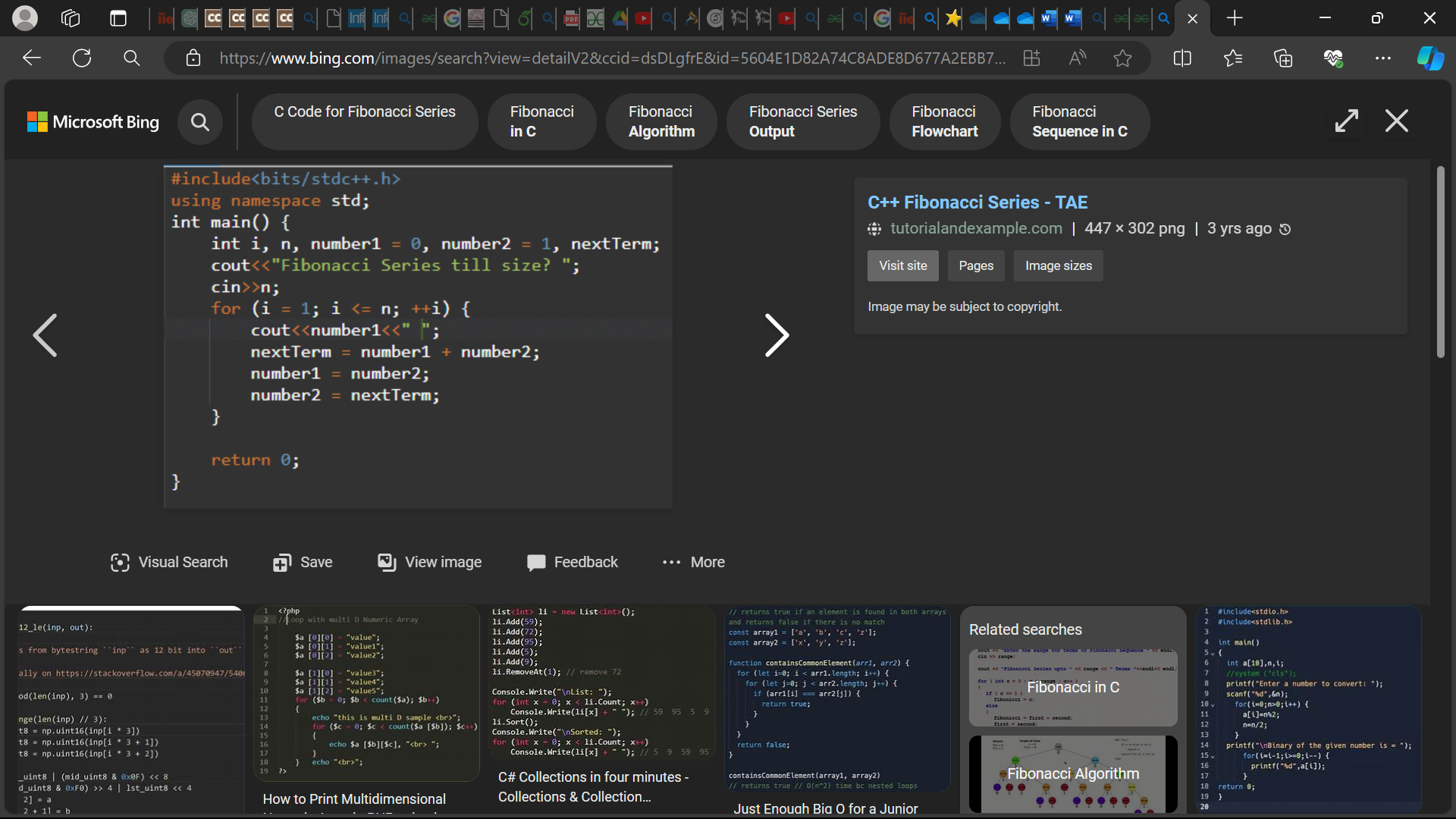
**Recursion uses a memory stack, and it stays inside the memory until it gets its return value. As** recursion call the function itself again, after reaching the stopping condition it returns the value, but till then the function calls remain in the memory stack.

Iteration solves this memory overhead problem.

**AN EFFICIENT ITERATIVE CODE FOR FIBONACCI SERIES**

We can use three variables: first term, second term and next term, with first term as 0 and second term as 1 and start the loop for size of series of we want. In each step next term can be calculated as first term + second term, then print this value, after that store the value of second term in first term variable, and next term value in second term variable and cycle goes on

FOR BETTER UNDERSTANDING REFER TO THE CODE BELOW:



Here there is no stack overhead **Space complexity: O (N)**

**Tower of Hanoi**

**Objective:** To solve the Tower of Hanoi puzzle that contains three disks. The stack of disks has to be shifted from Rod 1 to Rod 3 by abiding by the set of rules that has been mentioned above.

Follow the steps below to solve the problem:

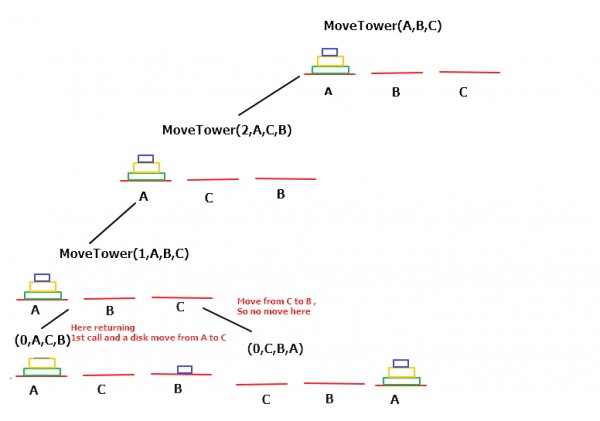
* Create a function **towerOfHanoi** where pass the **N** (current number of disk), **from\_rod**, **to\_rod**, **aux\_rod.**
* Make a function call for N – 1 th disk.
* Then print the current the disk along with **from\_rod** and **to\_rod**
* Again make a function call for N – 1 th disk.



TIME COMPLEXITY

The time complexity of the Tower of Hanoi solution using stacks is O(2^n), where n is the number of disks. This is because the number of moves required to solve the Tower of Hanoi problem grows exponentially with the number of disks. For each disk, there are two possible moves, leading to a total of 2^n moves.

RECURSIVE TREE



FLOWCHART

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  | **PRACTICE QUESTIONS**  1. <https://codeforces.com/group/MWSDmqGsZm/contest/223339/problem/B>  2. <https://codeforces.com/group/MWSDmqGsZm/contest/223339/problem/J>  3. <https://codeforces.com/group/MWSDmqGsZm/contest/223339/problem/I>  4. <https://codeforces.com/group/MWSDmqGsZm/contest/223339/problem/L>  5. <https://codeforces.com/group/MWSDmqGsZm/contest/223339/problem/L>  6. <https://codeforces.com/group/MWSDmqGsZm/contest/223339/problem/R>  7. [https://codeforces.com/group/MWSDmqGsZm/contest/223339/proble](https://codeforces.com/group/MWSDmqGsZm/contest/223339/problem/U)  [m/U](https://codeforces.com/group/MWSDmqGsZm/contest/223339/problem/U) |