Introduction

We have reached an important level in our path.

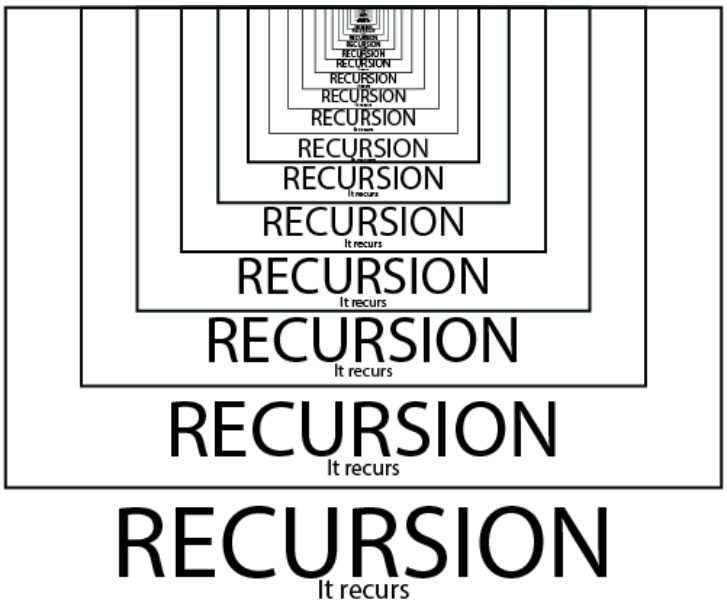
The next tool we need to learn in our journey is the recursive algorithms. During this super skill we are going to learn:

- What is the concept of recursivity?
- what are the rules of recursivity?
- What are the different types of a recursive algorithm?

Concept of Recursion

The process in which a function calls itself directly or indirectly, this allows the function to be repeated several times, since it calls itself during its execution. This act is called recursion and the corresponding function is called as recursive function. Using recursive algorithms, certain problems can be solved quite easily.

Recursion is often seen as an efficient method of programming since it requires the least amount of code to perform the necessary functions. However, recursion must be incorporated carefully, since it can lead to an infinite loop if no condition is met that will terminate the function.

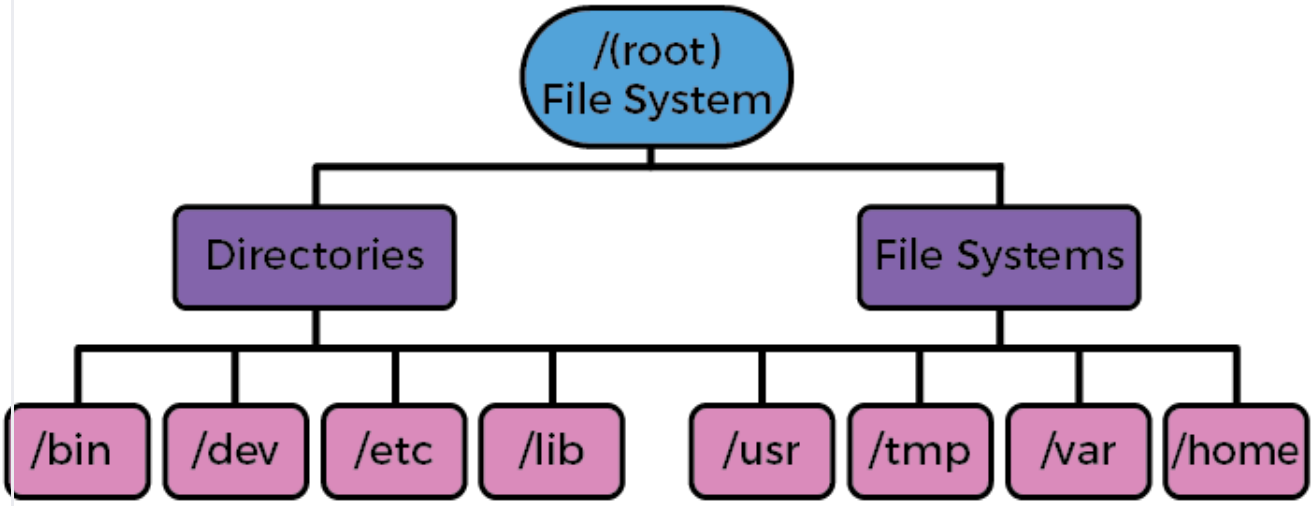


When do we use recursion?

Recursion is made for solving problems that can be broken down into smaller, repetitive problems. It is especially good for working on things that have many possible branches and are too complex for an iterative approach. We can divide this type of problem into two categories :

1. Problem P breaking down into 2 or more sub-problems of the same nature as P but of less complexity.
2. Recursive object / structure (definition of a mathematical function)

One good example of this would be searching through a file system. You could start at the root folder, and then you could search through all the files and folders within that one. After that, you would enter each folder and search through each folder inside of that.



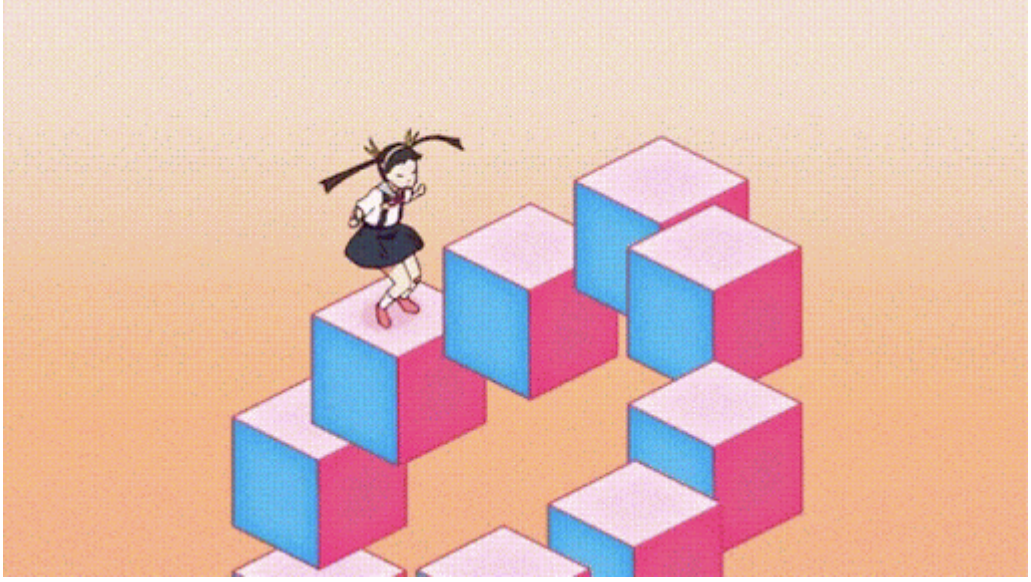
How does recursion work?

A big difference between recursion and iteration is the way that they end. While a loop executes the block of code, checking each time to see if it is at the end of the sequence, there is no such sequential end for recursive code. So, we stop a recursion in another way.

A recursive function consists of two parts: the recursive call and the base case (Stop condition).

The base case (or bases cases sometimes) is the condition that checks to see if we have gotten the information that we need out of our function. Every recursive function should have at least one base case, though there may be multiple.

Without the base case the recursion will be infinite. Like the horrifying loop of jumps of that little girl to the right, a recursive function could go on forever without a condition to put it out of its misery.



Previous next