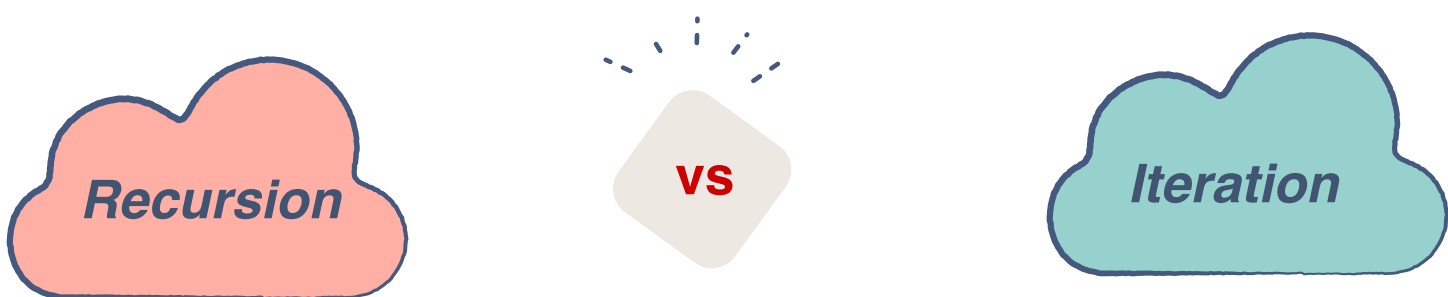


Is there a difference?#

Both recursion and iteration are used for executing instructions repeatedly until a condition is true. So, where does the difference actually lie between Iteration and Recursion? In this lesson, let’s discuss a few factors that differentiate the two methods.



Definition#

- Recursion refers to a situation where a method calls itself again and again until some base condition is not reached.
- Iteration refers to a situation where some statements are executed again and again using loops until some condition is true.

Application#

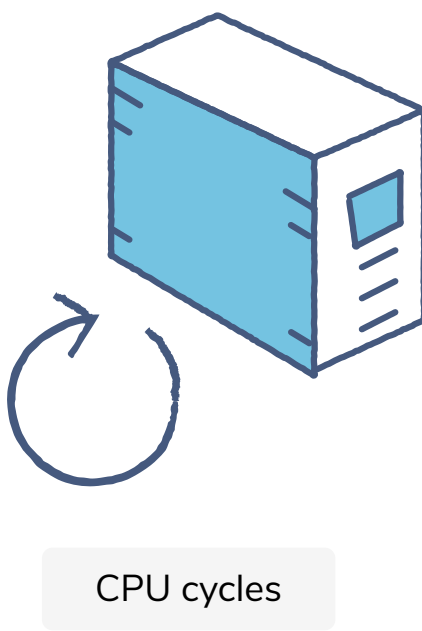
- Recursion is always called on a **method**, and therefore becomes a process.
- Iterative code is applied on **variables** and is a set of **instructions** that are called upon repeatedly.

Termination#

- Recursive code **terminates** on the base case condition.
- Iterative code either runs for a **particular number of loops** or until a specified **condition** is met.

Infinite case#

- An **infinite** recursion can lead to a code crash or stack overflow.
- An **infinite** iterative code, will consume more CPU cycles.

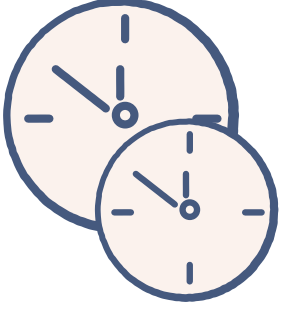


Code size#

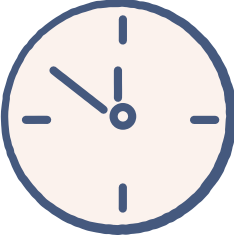
- Recursive code is **smaller and neater** in length.
- Iterative code is usually **extensive and cluttered**.

Overhead time#

- Recursive code has an **overhead** time for each recursive call it makes.
- Iterative code has no **overhead** time.



Takes more time

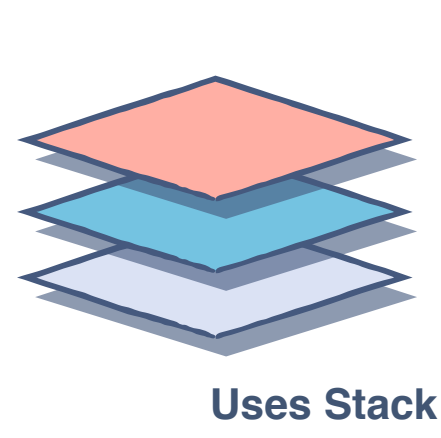


Takes lesser time

Speed#

- Recursive code is **slower** than the iterative code. It not only runs the program but also has to invoke the stack memory.
- Iterative code has a relatively faster runtime speed.

Stack utilization#



- Recursion uses the **stack** to store the variable changes for each recursive call.
- Iterative code does not use the stack.

Now that you are familiar with the basic differences between recursion and iteration, let’s move on to the next lesson and learn how to change an iterative code to a recursive code.