Recursion vs Iteration

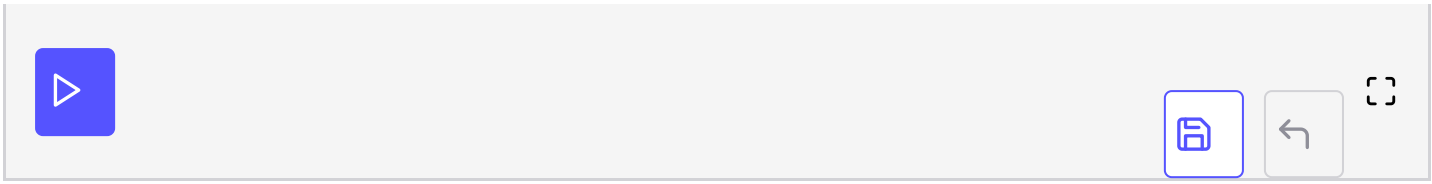
In this lesson, we will see how to calculate the factorial of a number iteratively.

We'll cover the following

Recursive solution

We can implement a recursive solution iteratively. Let's write a program to calculate the factorial of a number using a loop. In the iterative solution, we are not calling the same problem with a simpler version; instead, we are using the `counter` variable and keep incrementing it until the given condition is true.

```
#include <iostream>

using namespace std;

// Iterative factorial function
int factorial(int n) {
  int fact = 1;

  if (n == 0) {
    fact = 1;
  }

  for (int counter = 1; counter <= n; counter++) {
    fact = fact * counter;

  }
  return fact;
}

// main function
int main() {
  int n = 5;
  int result;
  // Call factorial function in main and store the returned value in result
  result = factorial(n);
  // Prints value of result
  cout << "Factorial of " << n << " = " << result;
  return 0;
}
```



Differences

Let’s see the difference between recursion and iteration.

- In the computer language, **iteration** allows you to repeat a particular set of instructions until the specified condition is met. Whereas, the **recursive function** allows you to keep calling itself in the function body until some condition is met.
- The sole purpose of iteration and recursion is to achieve repetition. Loops achieve repetition through the repetitive structure, whereas recursion achieves repetition through repetitive function calls.
- When the loop condition fails, iteration terminates. On the other hand, recursion terminates when our base condition evaluates to true.
- Iteration happens inside the same function; therefore, it takes less memory. Whereas, in the recursive function, there is the overhead of function calls that make our program slow, and it consumes more memory since each function call calls another copy of the function.
- In iteration, our code size is very large. Meanwhile, recursion helps us to write shorter code.
- Iterative code is faster than the recursive code.
- Infinite loops will stop the further execution of the program but do not lead to any system crash. Whereas, infinite recursive calls will result in a CPU crash because of memory overflow.

Why recursion?

It depends upon the requirements of your problem. The problems that can be defined in terms of itself are the best candidates for recursion.

Use recursion when it seems more natural to divide the problem into the simpler versions of itself. Therefore, recursion is best for cases where you can divide your

problem into branches. For example, consider the example of a file searching

system. In the file searching system, you start with the main folder and then go through each subfolder to find a particular file.

For such a type of problem, use recursion. The recursive solution helps you write shorter and more understandable code that makes debugging easier.

However, if performance is your main concern, then write the iterative solution since it takes less time and memory for its execution (Recursive calls take more time and extra memory.).

But, it's totally your choice.

Quiz

Which of the following statements are true?

(You can select multiple correct options)

Retake Quiz

Let’s solve some challenges related to recursion in the upcoming lessons.