

Recursion

Recursion

Recursion is a basic programming technique you can use in **Java**, in which a method calls itself to solve some problem. A method that uses this technique is **recursive**.

Many programming problems can be solved only by **recursion**, and some problems that can be solved by other techniques are better solved by **recursion**.

DEFINITION

Let's take **factorial** as an example:

```
static int factorial(int n){
    if (n == 1)
        return 1;
    else
        return
            (n * factorial(n-1));
}
```

Function

Actual program. Contains the base case and recursion case.

```
factorial(int n)
```

Check if

Finalization

Return base case, finishing the calculation.

Base Case

Required so that the program won't loop forever

```
if (n == 1)
    return 1;
```

Recursion Case

Cases that calls the function itself in order to finish the calculation

```
else
    return
        (n * factorial(n-1));
```

Recursion Cycle

factorial(n-1)

Call the function with the next parameter

DIAGRAM

Recursive

+ Pros:

- Reduce unnecessary calling of function.
- Recursion adds clarity and reduces the time needed to write and debug code.

- Cons:

- Difficulty to trace and debug
- Uses more memory and processing power
- No nested loops
- Bigger potential to overflow

Iterative

+ Pros:

- Easier to understand
- Nested loops
- Smaller potential to overflow

- Cons:

- Can't use the function itself as the base case.
- Difficulty in writing code if there's multiple cases.

COMPARISON

Recursion actually refers to the phenomenon of partially, self-repeating, which exists extensively in natural world.

The Ancient Greek philosopher **Democritus** proposed his atomic theory, which claims that matters can be repeatedly cut into parts until they become atoms, which means "uncuttable".

FUN FACT

StackOverflowError

It's one of the most common runtime errors we can encounter. Happens when we don't put base case in our recursion function, resulting in an infinite loop. **When a method is called, a new stack frame gets created on the call stack.**

If JVM encounters a situation where there is no space for a new stack frame to be created, it will throw a **StackOverflowError**.

ERROR