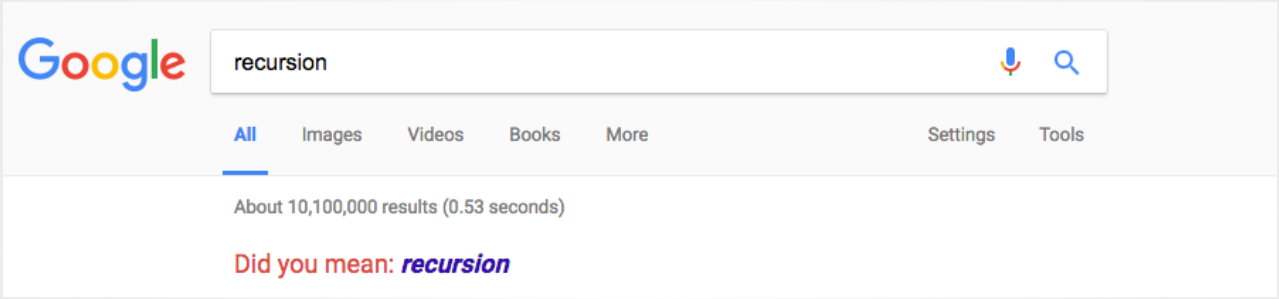
Recursion


Google
recursion
All
Images
Videos
Books
More
Settings
Tools
About 10,100,000 results (0.53 seconds)
Did you mean: recursion

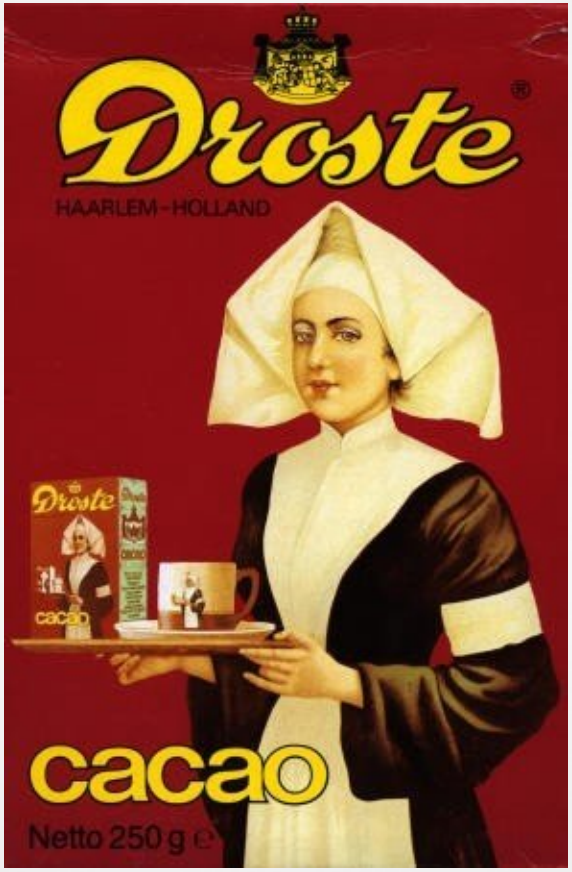
Recursion

Recursion is a programming technique in which a function calls itself.

The idea of recursion is that a problem is solved by finding solutions to smaller instances of the same problem.

Used as an alternative to iteration.

recursion.jpg



How to Solve A Problem Recursively:

- Find a solution for the trivial case or cases
  - These are known as the basis cases
  - Usually only 1 or 2 basis cases
- Breakup non-trivial inputs into smaller inputs until you reach the basis cases

In order to understand recursion, you must first understand recursion.

Factorial

The factorial of a number if is calculated using the formula

n! = 1 * 2 * 3 * 4 * ... * n

where n is a positive integer. By definition 0! is 1. Factorial is not defined for negative numbers.

Iterative Factorial

```
function factorial(n) {

  if (n < 0) {
    return -1;
  }

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

  var fact = n;

  while (n-- > 2) {
    fact *= n;
  }

  return fact;
}

console.log(factorial(8));
```

Recursive Factorial

```
function factorial(n) {

  if (n < 0) {
    return -1;
  }

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

  return (n * factorial(n - 1));
}

console.log(factorial(8));
```

Fibonacci

An integer sequence characterized by every number in the sequence being the sum of two preceding integers:

1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 ...

$F_n = F_{n-1} + F_{n-2}$

Iterative Fibonacci

```
function fibonacci(n) {
  var a = 0;
  var b = 1;
  var f = 1;

  for(var i = 2; i <= n; i++) {
    f = a + b;
    a = b;
    b = f;
  }

  return f;
}

console.log(fibonacci(12));
```

Recursive Fibonacci

```
function fibonacci(n) {
  if (n <= 2) {
    return 1;
  } else {
    return fibonacci(n - 2) + fibonacci(n - 1);
  }
}

console.log(fibonacci(12));
```

Recursion vs Iteration

Neither is inherently better, but there are known trade-offs:

- Recursion solutions are usually shorter
- Recursion is usually more difficult to design and test
- Recursion is more mathematical, usually considered more elegant
- Recursion can be more expensive than iteration
- Iteration feels less "magical"
- Iterative solutions are usually easier, or at least faster, to understand

Problems with Recursion:

infinite function calls infinite function calls infinite function calls

Recursion Exercise

What is the Output?

```
function gcd(a, b) {
  if (b === 0) {
    return a;
  } else {
    return gcd(b, a % b);
  }
}

console.log(gcd(20, 12));
```