Recursion and cases



- Every recursive algorithm involves at least 2 cases:
 - base case: A simple occurrence that can be answered directly.
 - recursive case: A more complex occurrence of the problem that cannot be directly answered, but can instead be described in terms of smaller occurrences of the same problem.
 - Key idea: In a recursive piece of code, you handle a small part of the overall task yourself, then make a recursive call to handle the rest.
 - Ask yourself, "How is this task self-similar?"
 - "How can I describe this algorithm in terms of a smaller or simpler version of itself?"

分器

"Recursion Zen"

• The real, even simpler, base case is an exp of 0, not 1:

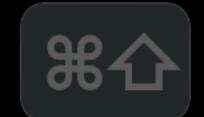
```
int power(int base, int exp) {
    if (exp == 0) {
        // base case; base^0 = 1
        return 1;
    } else {
        // recursive case: x^y = x * x^(y-1)
        return base * power(base, exp - 1);
    }
}
```

 Recursion Zen: The art of properly identifying the best set of cases for a recursive algorithm and expressing them elegantly.

(our informal term)

evaluate exercise





- Write a recursive function evaluate that accepts a string representing a math expression and computes its value.
 - The expression will be "fully parenthesized" and will consist of + and * on single-digit integers only.