

Recursion in Mathematics and Programming

Summation

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Compute summation

$$\sum_{k=1}^n = 1 + 2 + \dots + n.$$

It's recursive definition:

$$\text{sum}(0) = 0$$

$$\text{sum}(n) = \text{sum}(n-1) + n \quad (\text{for } n \geq 1)$$

Source code “*sum.jl*”.

Factorial

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Factorial function

$$\mathbf{fact}(n) = n! = 1 \cdot 2 \cdot \dots \cdot n.$$

It's recursive definition:

$$\mathbf{fact}(0) = 1$$

$$\mathbf{fact}(n) = \mathbf{fact}(n - 1) \cdot n \quad (\text{for } n \geq 1)$$

Source code “*fact.jl*”.

Factorial

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Evaluation of the expression **fact(3)**

$$\begin{aligned}\mathbf{fact(3)} &\Rightarrow \mathbf{fact(2) \cdot 3} \\ &\Rightarrow (\mathbf{fact(1) \cdot 2}) \cdot 3 \\ &\Rightarrow ((\mathbf{fact(0) \cdot 1}) \cdot 2) \cdot 3 \\ &\Rightarrow ((1 \cdot 1) \cdot 2) \cdot 3 \\ &\Rightarrow (1 \cdot 2) \cdot 3 \\ &\Rightarrow 2 \cdot 3 \\ &\Rightarrow 6\end{aligned}$$

Fibonacci numbers

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Recursive definition of the Fibonacci numbers:

$$\text{fib}(0) = 1$$

$$\text{fib}(1) = 1$$

$$\text{fib}(n) = \text{fib}(n-1) + \text{fib}(n-2) \quad (\text{for } n \geq 2)$$

Source code “*fib.jl*”.

Fibonacci numbers

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Evaluation of the expression **fib**(5)

$$\begin{aligned}\mathbf{fib}(5) &\Rightarrow \mathbf{fib}(3) + \mathbf{fib}(4) \\ &\Rightarrow (\mathbf{fib}(1) + \mathbf{fib}(2)) + (\mathbf{fib}(2) + \mathbf{fib}(3)) \\ &\Rightarrow (1 + (\mathbf{fib}(0) + \mathbf{fib}(1))) + ((\mathbf{fib}(0) + \mathbf{fib}(1)) + (\mathbf{fib}(1) + \mathbf{fib}(2))) \\ &\Rightarrow (1 + (1 + 1)) + ((1 + 1) + (1 + (\mathbf{fib}(0) + \mathbf{fib}(1)))) \\ &\Rightarrow (1 + 2) + (2 + (1 + (1 + 1))) \\ &\Rightarrow 3 + (2 + (1 + 2)) \\ &\Rightarrow 3 + (2 + 3) \\ &\Rightarrow 3 + 5 \\ &\Rightarrow 8\end{aligned}$$

Exponential running time, $O(2^n)$, too slow to be practical.

Improved recursive Fibonacci

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Evaluation of the expression **fib2**(5)

fib2(5) \Rightarrow **next**(1, 1, 2)
 \Rightarrow **next**(1, 2, 3)
 \Rightarrow **next**(2, 3, 4)
 \Rightarrow **next**(3, 5, 5)
 \Rightarrow 3 + 5
 \Rightarrow 8

Linear time, $O(n)$, this is a real improvement.

A simple calculator language

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Syntax

$\langle expr \rangle ::=$ *strings of digits*, represent integer numbers

| $(\langle expr \rangle + \langle expr \rangle)$

| $(\langle expr \rangle - \langle expr \rangle)$

| $(\langle expr \rangle \times \langle expr \rangle)$

Examples:

55

$(1 + 2)$

$((1 + 2) - (3 \times (4 \times 5)))$

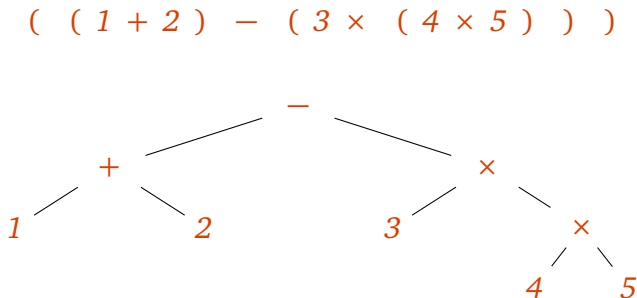
A simple calculator language

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems



The evaluating function looks at the root of the tree only, the operators are evaluated one at a time

$$\begin{aligned} & eval[(E_1 - E_2)] \\ \Rightarrow & eval[E_1] - eval[E_2] \end{aligned}$$

A simple calculator language

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Define the evaluation function:

Operators

$$\text{eval}[(E_1 + E_2)] = \text{eval}[E_1] + \text{eval}[E_2]$$

$$\text{eval}[(E_1 - E_2)] = \text{eval}[E_1] - \text{eval}[E_2]$$

$$\text{eval}[(E_1 \times E_2)] = \text{eval}[E_1] \times \text{eval}[E_2]$$

Numbers

$$\text{eval}[n] = n$$

A simple calculator language

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

$$\begin{aligned} & eval[((1 + 2) - (3 \times (4 \times 5)))] \\ \Rightarrow & eval[(1 + 2)] - eval[(3 \times (4 \times 5))] \\ \Rightarrow & (eval[1] + eval[2]) - (eval[3] \times eval[(4 \times 5)]) \\ \Rightarrow & (1 + 2) - (3 \times (eval[4] \times eval[5])) \\ \Rightarrow & 3 - (3 \times (4 \times 5)) \\ \Rightarrow & 3 - (3 \times 20) \\ \Rightarrow & 3 - 60 \\ \Rightarrow & -57 \end{aligned}$$

Recursion in Mathematics

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Some prominent examples of recursion

- *Euclid's algorithm*
a algorithm for computing the greatest common divisor.
- *Newton's method*
a recursive method for finding successively better approximations to the roots of a real-valued function.
- Complex objects such as fractals can be defined using recursive definition.

Define set M

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

$c \in M$ if the sequence

$$z_0 = 0$$

$$z_n = z_{n-1}^2 + c \quad \text{does not go to infinity.}$$

Define set M

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

$c \in M$ if the sequence

$$z_0 = 0$$

$$z_n = z_{n-1}^2 + c \quad \text{does not go to infinity.}$$

Numbers c and z_n are *complex numbers*. So, we need to quickly learn how to add and multiply them.

Quick intro to complex numbers

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

The set of complex numbers \mathbb{C} is an extension of the set of real numbers \mathbb{R} .

Any complex number can be represented by a pair of real numbers

$$(x, y) \quad x, y \in \mathbb{R}$$

x is the *real part*, and y is the *imaginary part*.

Alternative notation. You can the pair as a sum of its real and imaginary part

$$(x, y) = x + yi$$

$i = \sqrt{-1}$ is the *imaginary unit*.

Quick intro to complex numbers

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

Addition

$$(a, b) + (c, d) = (a + c, b + d)$$

Multiplication

$$(a, b) \cdot (c, d) = (ac - bd, bc + ad)$$

Particularly,

$$i^2 = i \cdot i = (0, 1) \cdot (0, 1) = (0 - 1, 0 + 0) = (-1, 0) = -1$$

Define set M

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

A complex number $c \in M$, if the sequence

$$z_0 = 0$$

$$z_n = z_{n-1}^2 + c \quad \text{does not go to infinity.}$$

$z_n = x_n + y_n i$ will go to infinity if one of its components gets large:

$$\sqrt{x_n^2 + y_n^2} \geq 2$$

This condition is used practically to compute the membership of c in the set.

Source code “*mset.jl*”.

L-systems

An L-system or Lindenmayer system is a parallel *rewriting system*.

It consists of

- an alphabet of symbols
- an initial string (called an axiom) to start construction
- a collection of production rules that expand each symbol into some larger string of symbols

Example:

<i>Alphabet:</i>	$\{A, B\}$
<i>Initial string:</i>	A
<i>Production rules:</i>	$A \rightarrow AB$
	$B \rightarrow A$

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

L-systems

Example:

Alphabet: $\{A, B\}$
Initial string: A
Production rules: $A \rightarrow AB$
 $B \rightarrow A$

Start rewriting:

$n = 0 : A$

$n = 1 : AB$

$n = 2 : ABA$

$n = 3 : ABAAB$

$n = 4 : ABAABABA$

$n = 5 : ABAABABAABAAB$

$n = 6 : ABAABABAABAABAABAABABA$

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems

L-systems

Koch curve:

Alphabet: $\{F, L, R\}$
Initial string: F
Production rules:
 $F \rightarrow F L F R F R F L F$
 $L \rightarrow L$
 $R \rightarrow R$

When F stands for moving forward, and L and R are the commands for turning left and right



Source code “*lsys.jl*”.

Simple examples

Evaluating
expressions

Recursion in
Mathematics

L-systems