Faculdade de Design, Tecnologia e Comunicação
Universidade Europeia

Recursion

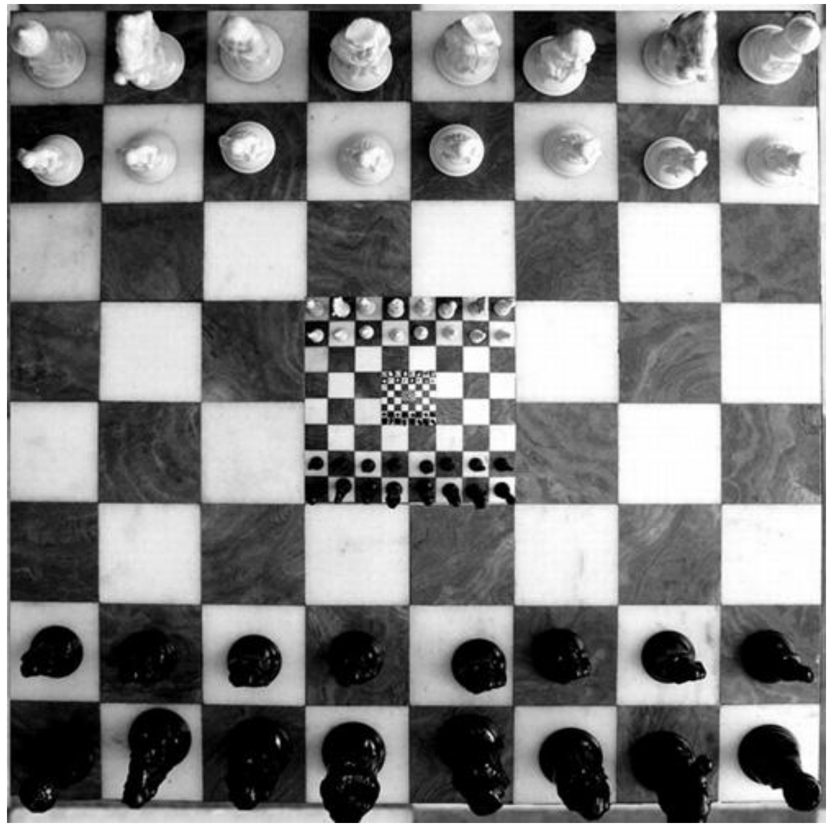
Data Structures

Estruturas de Dados

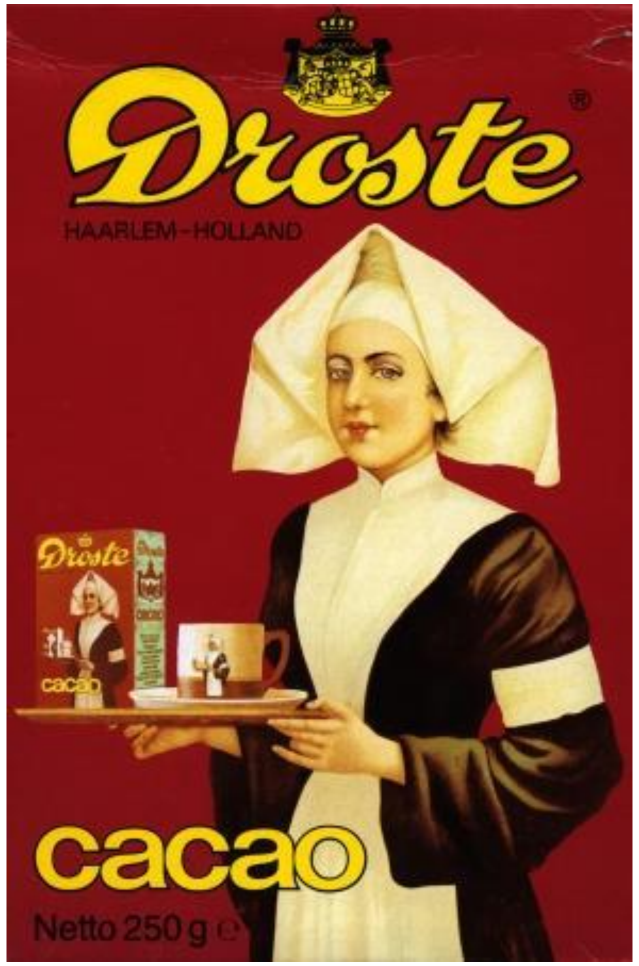
Fernando Marson

fernando.marson@universidadeeuropeia.pt

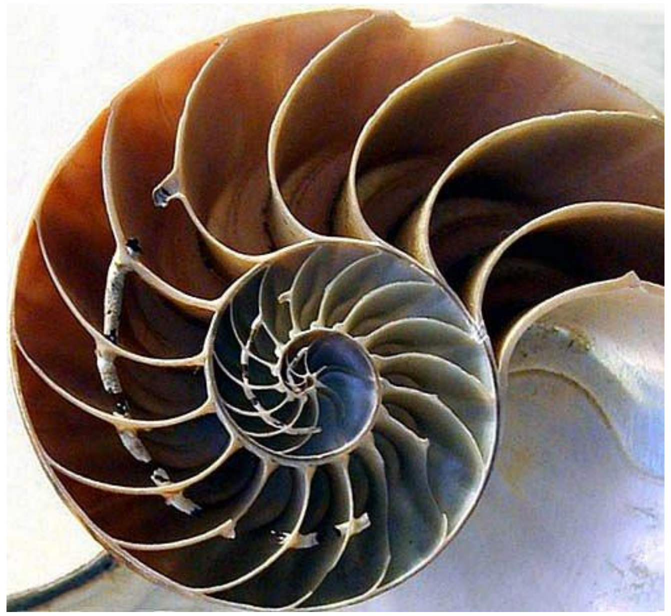
Recursion

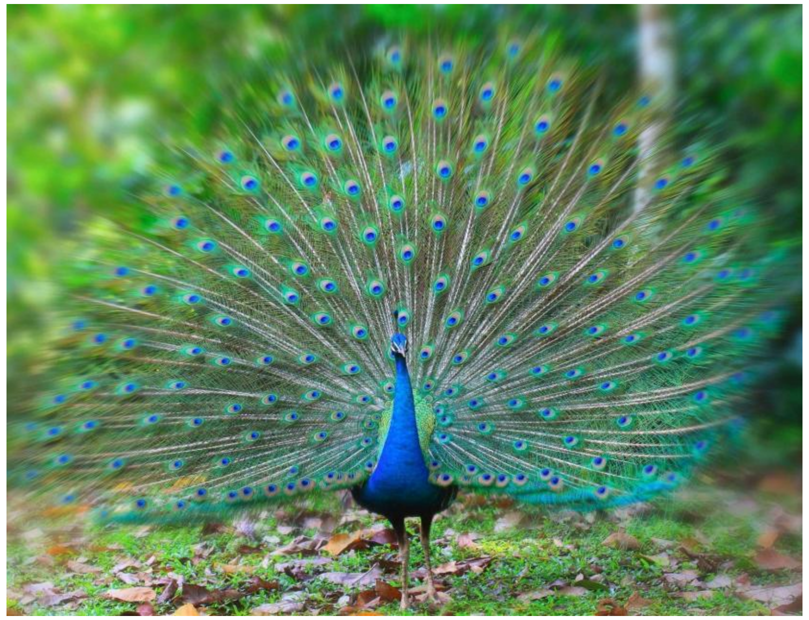


Recursion



Natural Recursion



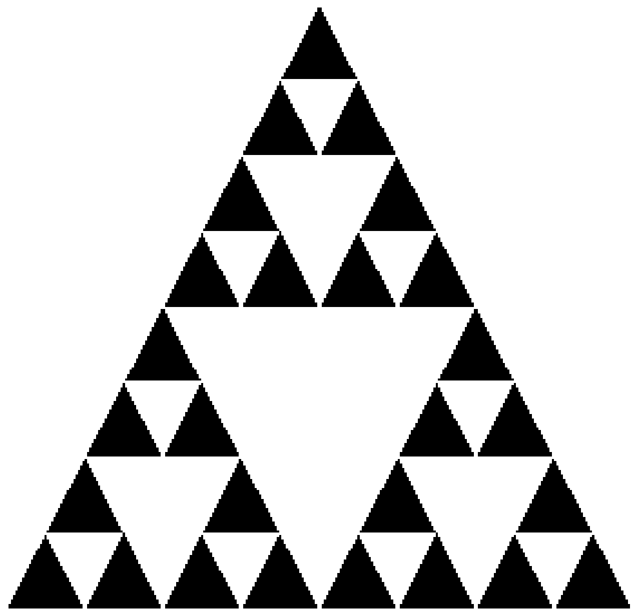


Recursion

- Property of that which can be repeated an indefinite number of times.

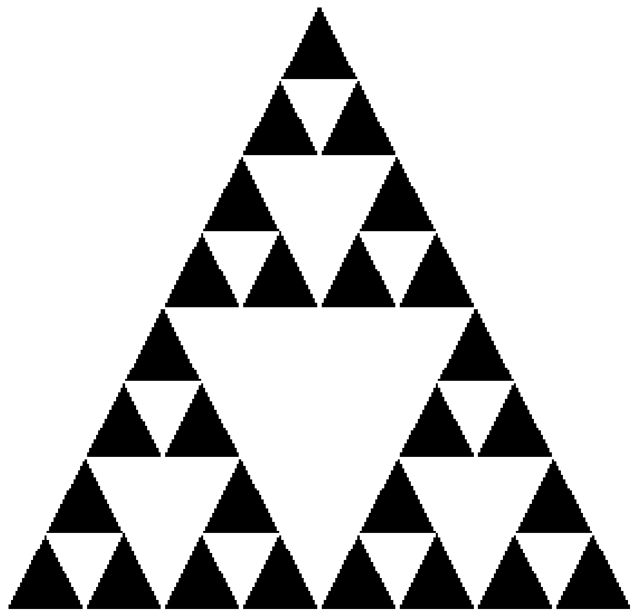
Recursion - Sierpinski triangle

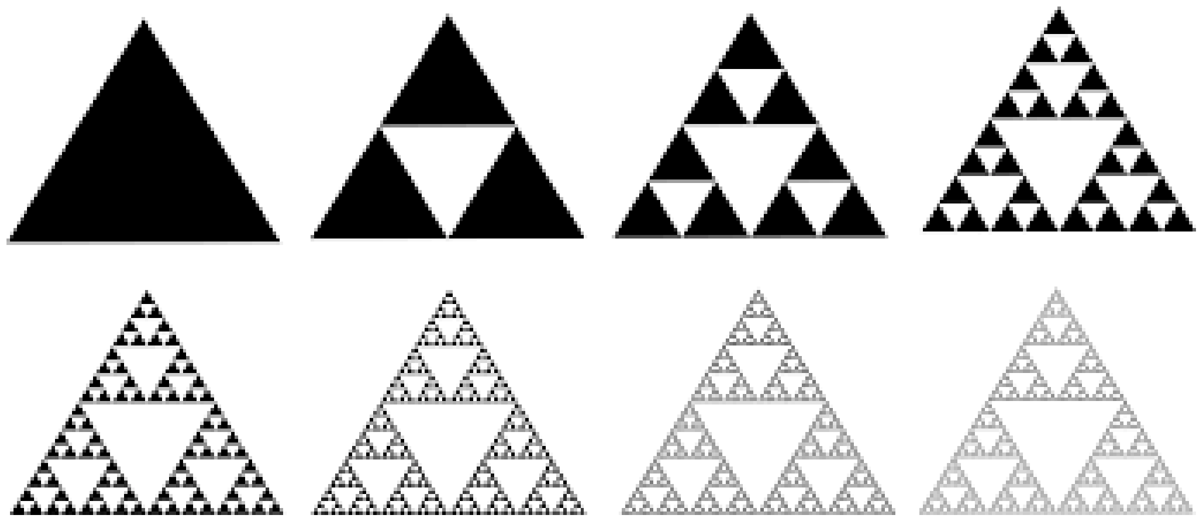
- Property of that which can be repeated an indefinite number of times.



Recursion - Sierpinski triangle

- Property of that which can be repeated an indefinite number of times.

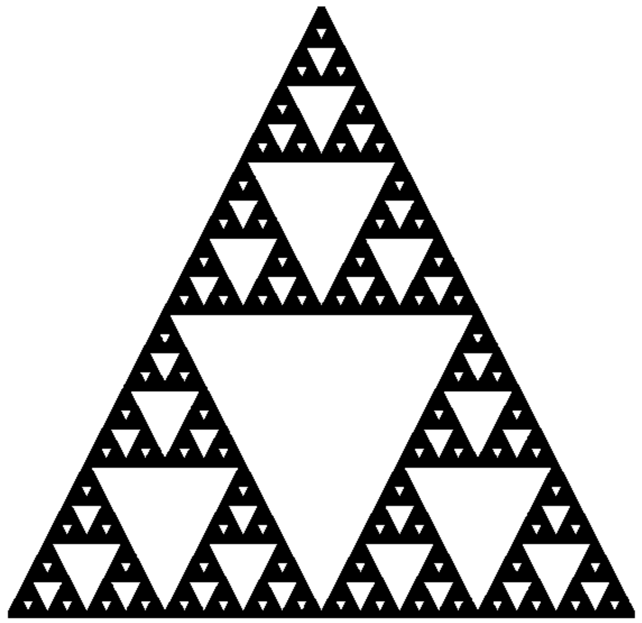


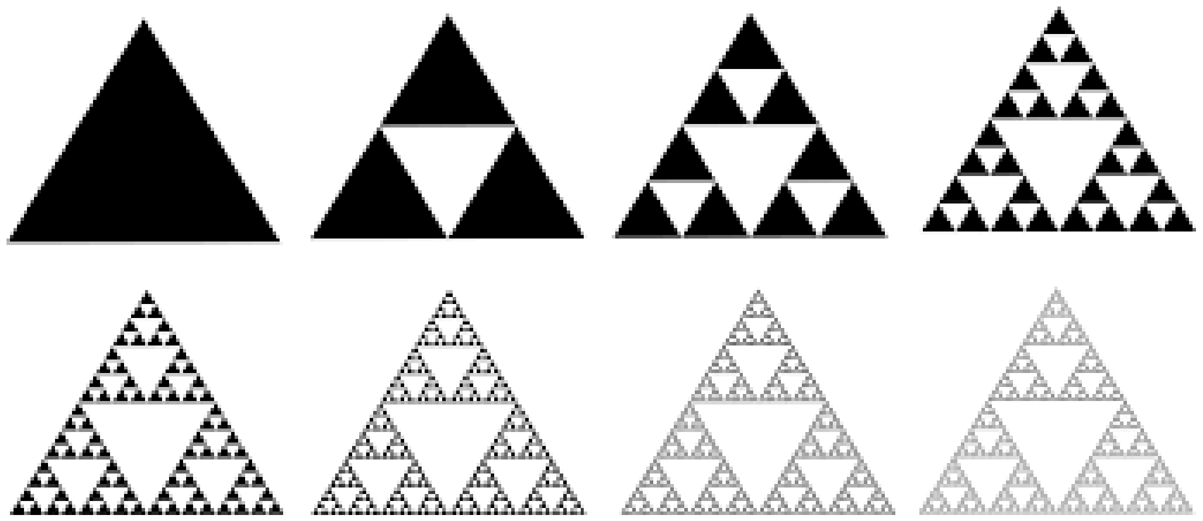


What is the rule in this case?

Recursion - Sierpinski triangle

- Property of that which can be repeated an indefinite number of times.



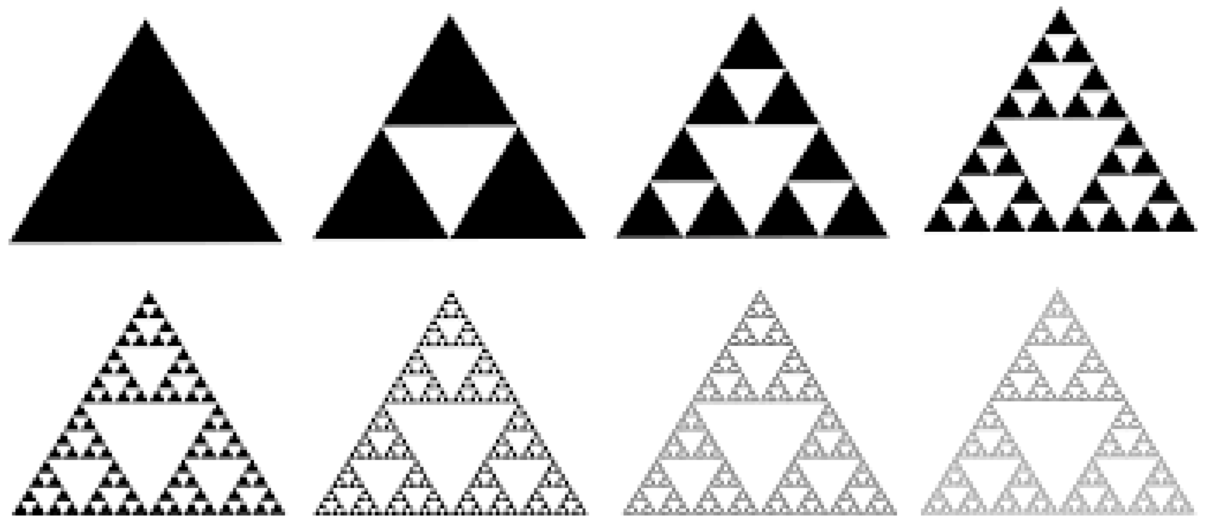


What is the rule in this case?

Recursion - Sierpinski triangle

- Property of that which can be repeated an indefinite number of times.

- Remove the middle rule
  - Start with a solid equilateral triangle.
  - Divide it into 4 smaller equilateral triangles (by connecting the midpoints of the sides).
  - Remove the middle triangle (the inverted one).
  - Repeat the process on each of the remaining triangles, indefinitely.



What is the rule in this case?

Faculdade de Design, Tecnologia e Comunicação
Universidade Europeia

Exercise 1A

- Write a program with a function that receives a number and prints that number minus 1 on the screen repeatedly until it reaches 0.

Exercise 1B

- Write a program with a function that receives a number and prints the number minus 1 on the screen until it reaches 0, without using loops.

Recursion

- Recursion is a programming technique in which a function calls itself to solve a problem. This approach breaks the problem into smaller, more manageable subproblems like the original

- A recursive function typically includes:
  - **Base case**: a condition that stops the recursion. This is essential to prevent infinite recursion and stack overflow; the base case directly solves the simplest instance(s) of the problem.

  - **Recursive Case:** the part where the function calls itself with a modified argument, progressively moving toward the base case.

Recursion

- Recursion is often used for problems that can be naturally divided into similar subproblems, such as mathematical computations (e.g., factorial, the Fibonacci sequence), data-structure traversal (e.g., trees, graphs), and algorithms (e.g., divide and conquer, dynamic programming).

Recursion vs Iteration

- Functions with the same purpose can be implemented either iteratively or recursively. Both approaches rely on control structures.
- The iterative form uses repetition constructs (for, while, do-while).
- The recursive form uses a selection construct (if–else).

Recursion vs Iteration

- Both involve repetition: iteration explicitly uses it, while recursion achieves the same behavior through repeated function calls.

- However, it is important to ensure that using recursion brings benefits to your code and to the application.

Recursion

- Pow calculation:
  - $3^0 = 1$
  - $3^1 = 3 * (3^0) \rightarrow 3$
  - $3^2 = 3 * (3 * (3^0)\ ) \rightarrow 9$
  - $3^3 = 3 * (3 * (3 * (3^0)\ )\ ) \rightarrow 27$

pow(y)	2^y	3^y	4^y
0	1	1	1
1	2	3	4
2	4	9	16
3	8	27	64

Recursion

- Pow calculation:
  - $x^0 = 1$
  - $x^1 = x * pow(x, 0)$
  - $x^2 = x * pow(x, 1) \rightarrow x * (x * pow(x, 0))$
  - $x^3 = x * pow(x, 2) \rightarrow x * (x * (x * pow(x,0)))$

pow(y)	2^y	3^y	4^y
0	1	1	1
1	2	3	4
2	4	9	16
3	8	27	64

Recursion

- Pow calculation:
  - If the power is 0 (zero), the result is 1.
  - This is the stopping criterion.

pow(y)	2^y	3^y	4^y
0	**1**	**1**	**1**
1	2	3	4
2	4	9	16
3	8	27	64

Recursion

- Pow calculation:
  - If the power is 0 (zero), the result is 1.
  - This is the stopping criterion.

```
int pow(int x, int y){
      if(y == 0)
      return 1; // stop
      else
      return  (x * pow(x, y - 1)); // recursive call
}
```

Recursion - Advantages

- **Simplicity**
  - Recursive solutions are often more elegant and easier to read than iterative ones.
- **Natural Fit**
  - Some problems are more naturally and intuitively solved using recursion.

Recursion - Disadvantages

- **Performance**
  - Recursive calls can be less efficient due to overhead from multiple function calls and the potential for stack overflow if not handled properly.
- **Memory Usage**
  - Each recursive call consumes stack space, which can become significant for deep recursion.

Recursion - Exercise

- Palindromes are words, phrases, or any type of unit that is the same when read from left to right as from right to left.

- Write a program to check if a word is a palindrome in a recursive way.

Faculdade de Design, Tecnologia e Comunicação
Universidade Europeia