Marisangila Alves, MSc
marisangila.alves@catolicasc.org.br
marisangila.com.br

Católica de Santa Catarina

2025/1

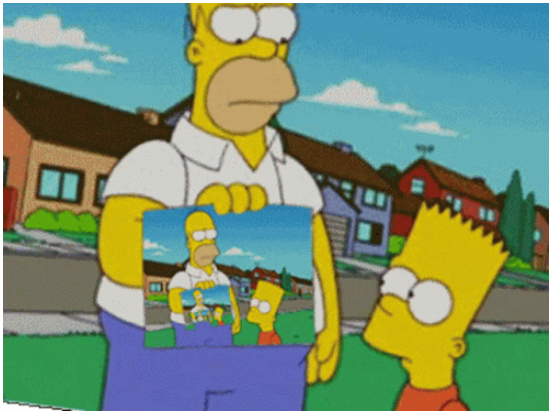
Estrutura de Dados

Estrutura de Dados
Recursividade

Sumário

Introdução à Recursão

Conceito de Recursão I



- Uma função recursiva é uma função que chama a si mesma.
- Utilizada para problemas que podem ser divididos em subproblemas semelhantes ao original.

- **Caso base**: quando o problema é simples o suficiente para ser resolvido diretamente.
- **Chamada recursiva**: divide o problema e chama a si mesma com uma versão menor.
- **Combinação de resultados**: as chamadas retornam valores que são combinados.

Fatorial

Definição Recursiva do Fatorial

- O fatorial de n é definido como:

$$n! = \begin{cases} 0 & \text{se } n \leq 1 \\ n \times (n-1)! & \text{caso contrário} \end{cases}$$

- Exemplo: $5! = 5 \times 4 \times 3 \times 2 \times 1 = 120$

```
1  int fatorial(int n){
2      if(n == 0)
3          return 1;
4      int fatorial = 1;
5      while(n > 1){
6          fatorial *= n;
7          n--;
8      };
9      return fatorial;
10 }
```

Código 1: Cálculo iterativo do fatorial.

```
1 int fatorial(int n){
2     if(n == 0)
3         return 1;
4     return n *= fatorial(n - 1);
5 }
```

Código 2: Cálculo recursivo do fatorial.

Fluxo de Chamadas Recursivas

Chamadas:

$$
\begin{aligned}
fatorial(5) &= 5 \times fatorial(4) \\
fatorial(4) &= 4 \times fatorial(3) \\
&\vdots \\
fatorial(1) &= 1
\end{aligned}
$$

Retornos:

$$
\begin{aligned}
fatorial(1) &= 1 \\
fatorial(2) &= 2 \times 1 = 2 \\
fatorial(3) &= 3 \times 2 = 6 \\
&\vdots \\
fatorial(5) &= 5 \times 24 = 120
\end{aligned}
$$

Versão Iterativa

```
long fatorial(long n) {
    long resultado = 1;
    for (long i = n; i >= 1; i--)
        resultado *= i;
    return resultado;
}
```

Versão Recursiva

```
long fatorial(long n) {
    if (n <= 1)
        return 1;
    else
        return n * fatorial(n - 1);
}
```

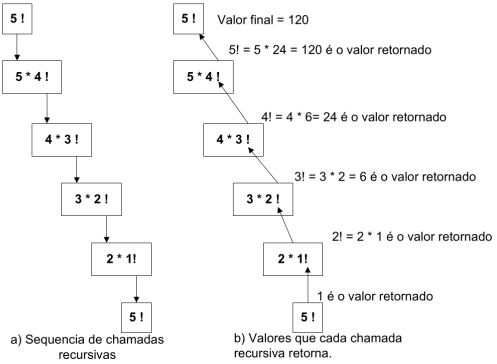


Figura 1: Figura: Chamadas e retornos de `fatorial(5)` (Deitel; Deitel, 2011).

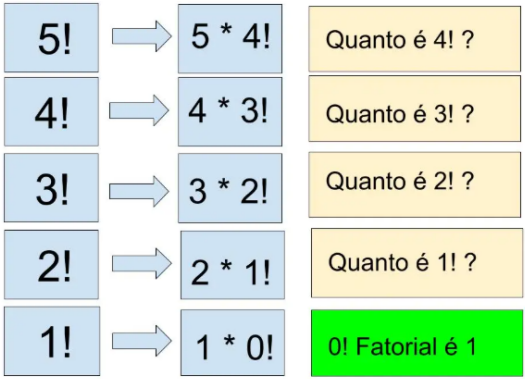


Figura 2: Pilha de chamada das Funções.

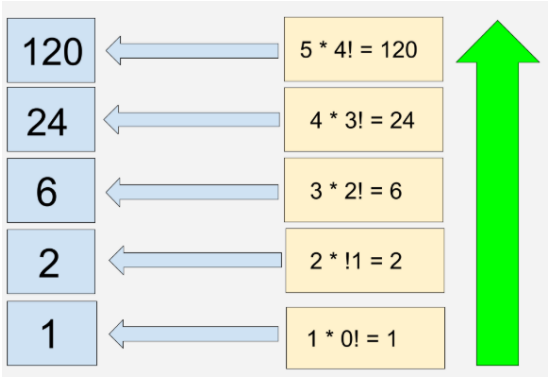


Figura 3: Pilha de retorno das Funções.

Potência

```c
int potencia(int base, int expoente) {
    int resultado = 1;
    if (expoente == 0)
        return 1;
    for(int i = 0;i<expoente;i++)
        resultado *= base;
    return resultado;
}
```

Código 3: Cálculo iterativo de potência.

```c
int potencia(int base, int expoente) {
    if (expoente == 0)
        return 1;
    return base * potencia(base, expoente - 1);
}
```

Código 4: Cálculo recursivo de potência.

Multiplicação

```c
int multiplicar(int n, int m)
{
    int resultado = 0;
    for (int i = 0; i < m; i++)
    {
        resultado += n;
    }
    return resultado;
}
```

Código 5: Cálculo iterativo de multiplicação.

```c
int multiplicar(int n, int m)
{
    if(m == 1)
        return n;
    return n + multiplicar(n,m-1);
}
```

Código 6: Cálculo recursivo de multiplicação.

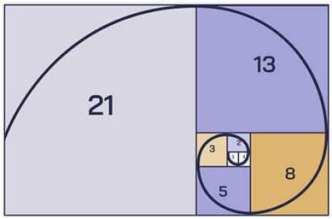
Fibonacci

THE FIBONACCI SEQUENCE

Each number is the sum of the two that precede it.

0 1 1 2 3 5 8 13 21

0 + 1 = 1
1 + 1 = 2
1 + 2 = 3
2 + 3 = 5
3 + 5 = 8
5 + 8 = 13
8 + 13 = 21

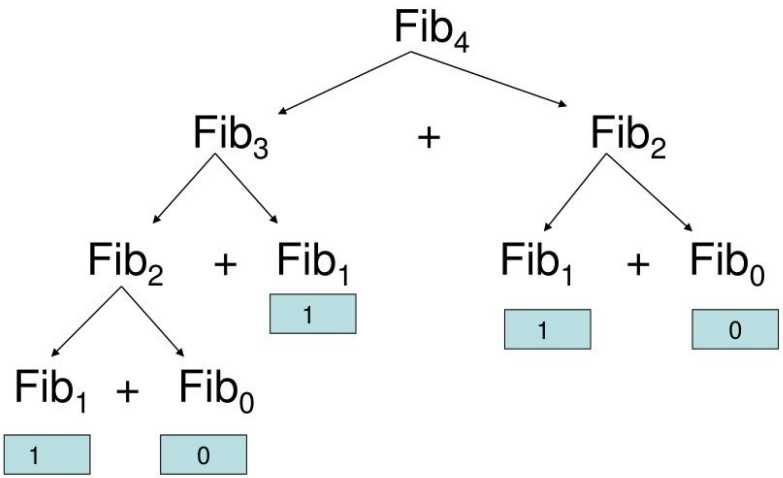


```c
void fibonacci(int n) {
    int anterior = 1, posterior = 1, aux;
    if (n >= 1)
        printf("1 ");
    if (n >= 2)
        printf("1 ");
    for (int i = 3; i <= n; i++) {
        aux = anterior + posterior;
        printf("%d ", aux);
        anterior = posterior;
        posterior = aux;
    }
}
```

Código 7: Cálculo iterativo do Fibonacci.

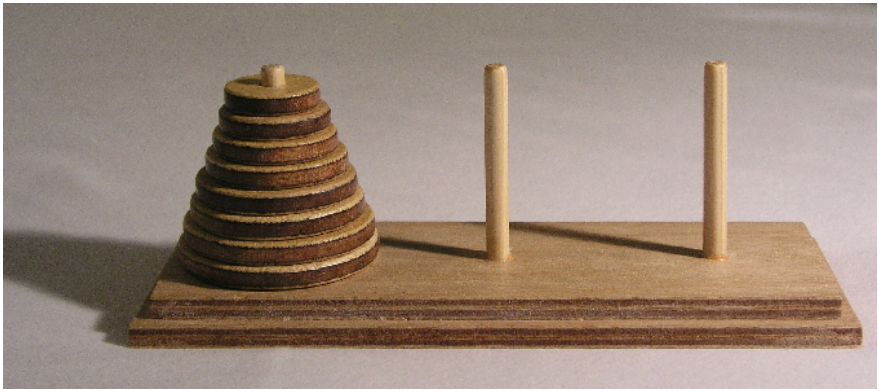
```c
int fibonacci(int n) {
    if (n == 0)
        return 0;
    else if (n == 1)
        return 1;
    else
        return fibonacci(n - 1) + fibonacci(n - 2);
}
```

Código 8: Cálculo recursivo do Fibonacci.


Fib4
Fib3
+
Fib2
Fib2
+
Fib1
Fib1
+
Fib0
1
1
0
Fib1
+
Fib0
1
0

Torre de Hanoi

Recursão e Torre de Hanoi I



Recursão e Torre de Hanoi II

- Existem 3 hastes: **A, B, C**.
- N discos de tamanhos diferentes empilhados na haste A.
- Objetivo: mover todos os discos de A para C, seguindo as regras:
  1. Mover apenas 1 disco por vez.
  2. Disco maior nunca pode ficar sobre disco menor.

Recursão e Torre de Hanoi III

Jogar

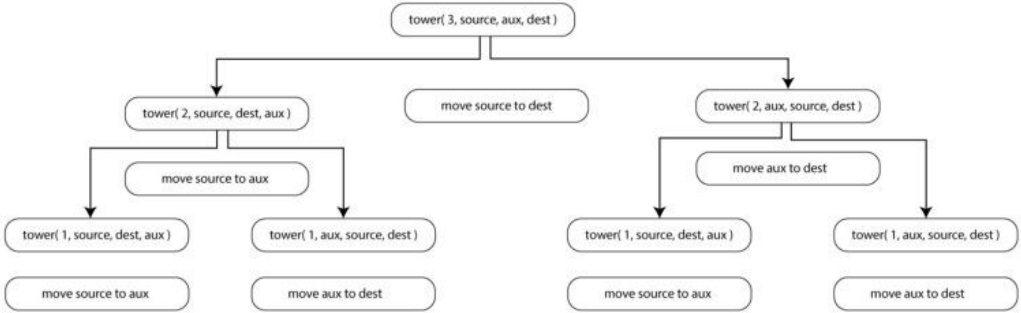
Clique aqui para jogar:

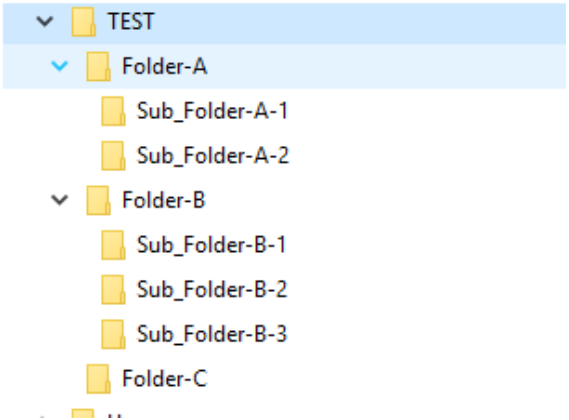
Recursão e Torre de Hanoi IV

```c
#include <stdio.h>
void hanoi(int n, char origem, char auxiliar, char destino) {
    if (n == 1) {
        printf("Mover disco 1 de %c para %c\n", origem, destino);
        return;
    }
    // Mover n-1 discos de origem para auxiliar
    hanoi(n-1, origem, destino, auxiliar);
    // Mover o disco n de origem para destino
    printf("Mover disco %d de %c para %c\n", n, origem, destino);
    // Mover n-1 discos do auxiliar para destino
    hanoi(n-1, auxiliar, origem, destino);
}
```

Código 9: Algoritmo Torre de Hanoi.

Recursão e Torre de Hanoi V

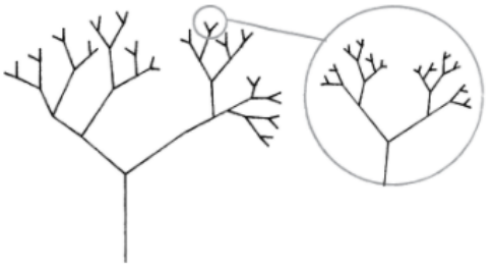


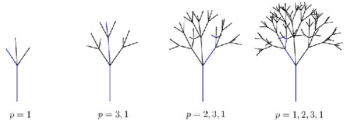

TEST
Folder-A
Sub_Folder-A-1
Sub_Folder-A-2
Folder-B
Sub_Folder-B-1
Sub_Folder-B-2
Sub_Folder-B-3
Folder-C

Extra - Fractais

?

O QUE UM BRÓCOLIS E FUNÇÕES RECURSIVAS TÊM EM COMUM?





Clique aqui para assista ao vídeo

Fractais

Fractal (do latim *fractu*: fração, quebrado) é uma figura da geometria não clássica muito encontrada na natureza, isto é, um objeto em que suas partes separadas repetem os traços (a aparência) do todo completo (padrão repetitivo).

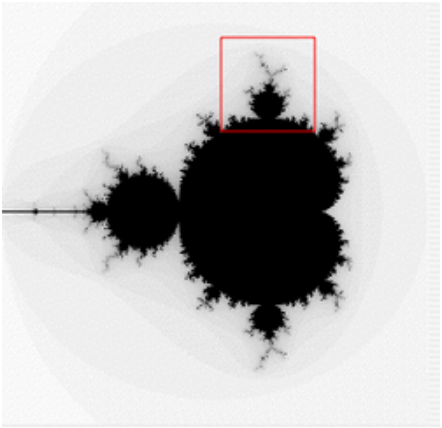
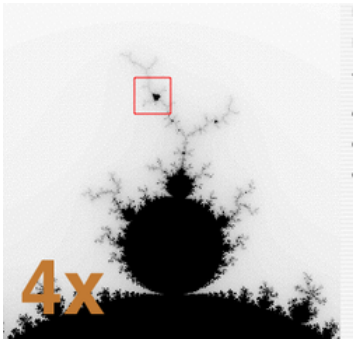
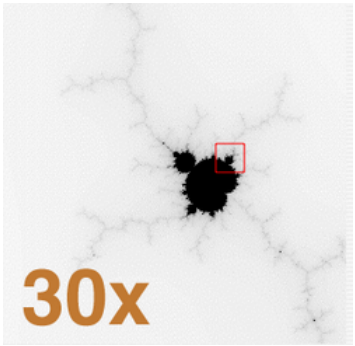
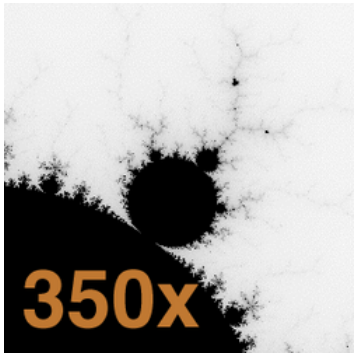
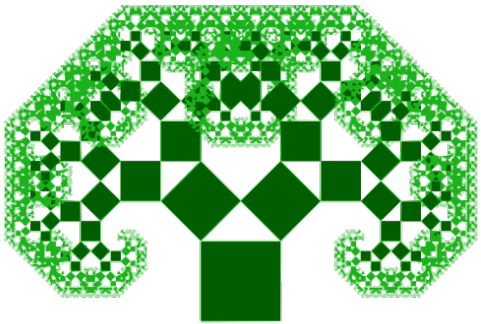


Figura 4: Conjunto de Mandelbrot.

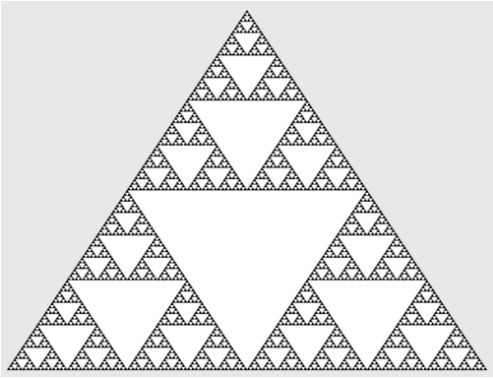

4x


30x


350x

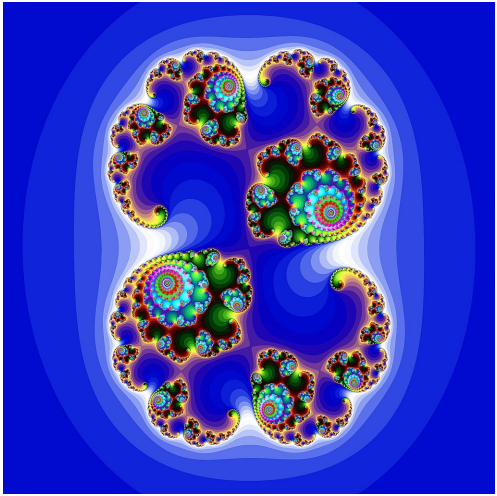


Árvore de Pitágoras

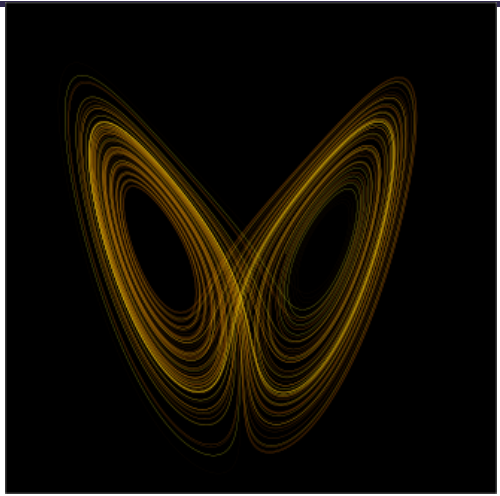


Triângulo de Sierpiński

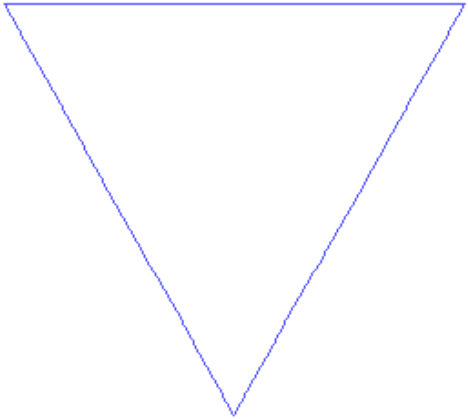
Fractais V



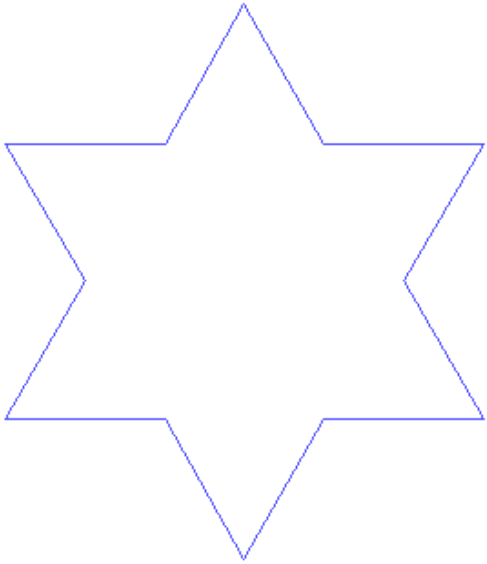
Conjunto de Julia



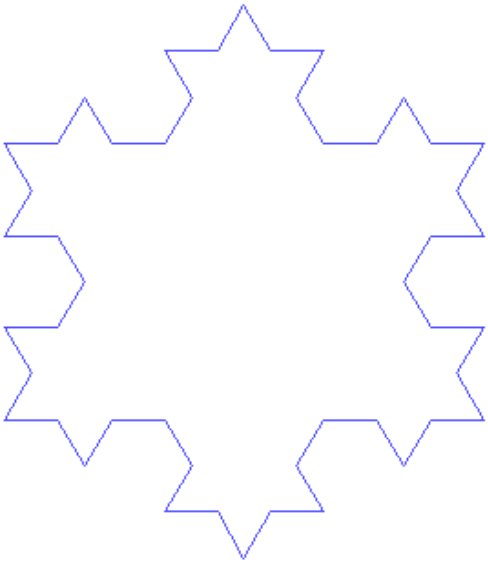
Atrator de Lorenz



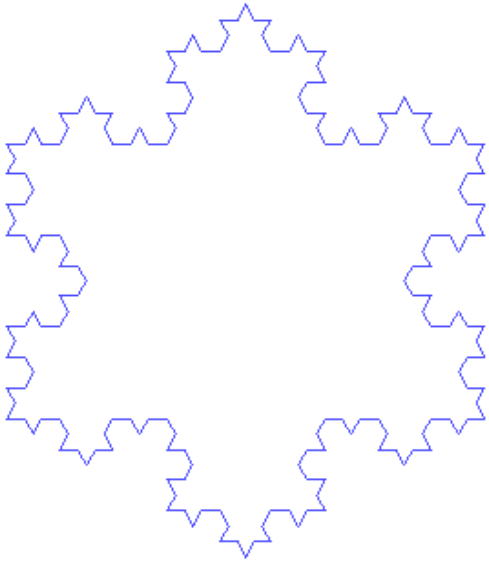
Curva de Von Koch



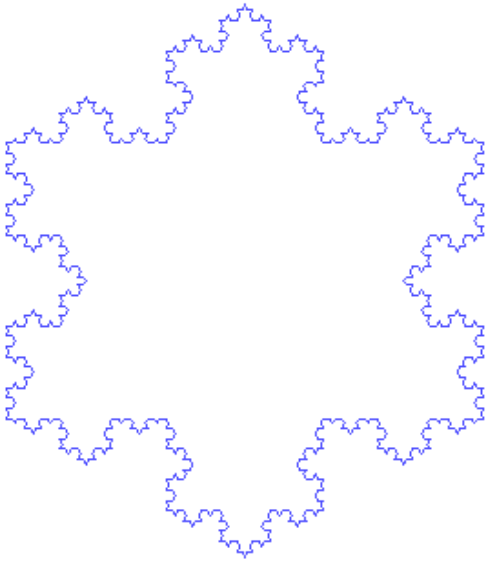
Curva de Von Koch



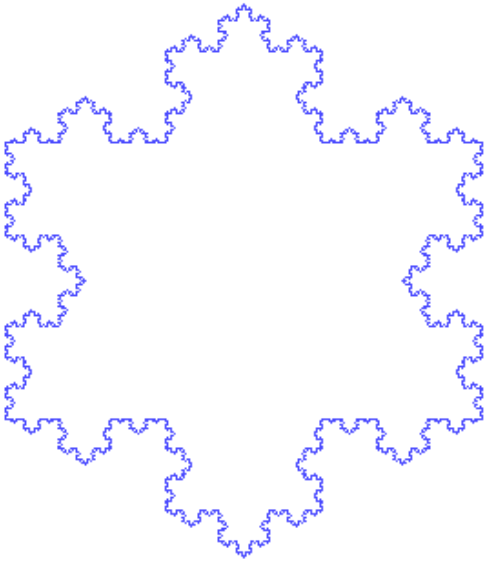
Curva de Von Koch



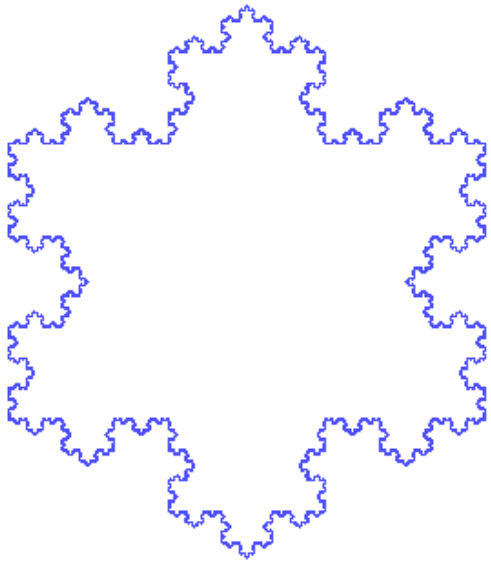
Curva de Von Koch



Curva de Von Koch



Curva de Von Koch



(Deitel; Deitel, 2011) - Capítulo/Seção 5.13

CELES, Waldemar; CERQUEIRA, Renato; RANGEL, José Lucas. **Introdução a estruturas de dados: com técnicas de programação em C**. Rio de Janeiro: Elsevier, 2004.

DEITEL, Paul; DEITEL, Harvey. **C: Como Programar**. 6. ed. São Paulo: Pearson Universidades, 2011.

SCHILDT, Herbert. **C Completo e Total: O Guia Definitivo para Programação em C**. 3. ed. São Paulo: Makron Books, 1996. ISBN 978-8534606928.

DEKKING, Michel *et al.* (ed.). **Fractals: Theory and Applications in Engineering**. London: Springer, 2011. Edição em Inglês, capa comum. ISBN 978-1-4471-0872-6. Disponível em: https://link.springer.com/book/10.1007/978-1-4471-0873-3.

Estes slides estão protegidos por uma licença Creative Commons

Este modelo foi adaptado de Maxime Chupin.

Marisangila Alves, MSc
marisangila.alves@catolicasc.org.br
marisangila.com.br

Católica de Santa Catarina

2025/1

Estrutura de Dados

Estrutura de Dados
Recursividade