LOGO COMO LINGUAGEM DE MACRO EM PROGRAMA DE GEOMETRIA DINÂMICA

Rafael Garcia Barbastefano
CEFET/RJ – MEPCM/DEPRO – rgb@cefet-rj.br

Alexandre Sardinha
DCC/LabMA – IM-UFRJ- alexandre.sardinha@gmail.com

Renato Campos Mauro
LabMa – IM-UFRJ – renato.mauro@gmail.com

Luiz Carlos Guimarães
LabMA – IM-UFRJ – lcg@labma.ufrj.br

1. Introdução

Macro-construções são elementos importantes em qualquer programa de geometria dinâmica. Através delas, é possível encapsular diversas etapas de uma construção em um único comando (Pratt, 1997), facilitando o processo de construções mais complexas e, por conseguinte, enriquecendo a lista de construções disponíveis aos usuários (Bellemain, 1992). Seus usos variam desde o ensino de figuras geométricas elementares para crianças (Ainley, 2006) até a ilustração de conceitos mais complexos como a geometria em espaços de Minkowski (Felsager, 2004). Alguns programas de geometria dinâmica como o Cabri possibilitam a mudança de elementos de interface através de macros, podendo-se restringir funcionalidades com fins didáticos específicos, como usado por Cabariti e Jahn (2006) no ensino de geometria hiperbólica.

Neste trabalho, apresentamos uma implementação de macro em um programa de geometria dinâmica na forma de programa na linguagem LOGO. Ao invés do registro e reutilização de pedaços de construções, elabora-se uma construção a partir de instruções elaboradas em uma linguagem de programação estruturada. Este tipo de implementação permite uma grande flexibilidade na elaboração de construções, além do

estabelecimento de estruturas de controle (if, while, for) e da utilização de elementos de interface gráfica com o usuário (caixas de diálogo, barras de rolagem, combos).

O presente artigo inicialmente apresenta o programa Tabulæ, em seguida, mostra-se como a implantação do TABULOGO (nome adotado) foi realizada em seguida, apresenta-se um estudo exploratório de possibilidades de construção com o software.

2. O Tabulæ

A geometria dinâmica é um conceito computacional que representa uma classe de programas usados como tecnologia educacional para o ensino de matemática (Schuman, 1989). O Tabulæ (Figura 1, Guimarães et al. 2001) é um software de Geometria Dinâmica plana que vem sendo desenvolvido e utilizado há pouco mais de seis anos, inicialmente no Projeto Enibam do IM/UFRJ, e hoje em projetos do LIMC (Laboratório de Pesquisa e Desenvolvimento no Ensino de Matemática e das Ciências). Estão envolvidos no projeto alunos de graduação dos cursos de engenharia, bacharelado em matemática, informática, licenciatura em matemática e desenho industrial, além de alunos de mestrado e doutorado. A versão atual do Tabulæ contém funcionalidades geométricas e vetoriais, além de calculadora. O objetivo principal do programa é proporcionar uma alternativa brasileira, comparável aos melhores softwares encontrados no mercado hoje em dia. É também um dos objetivos do projeto servir de plataforma para o desenvolvimento de funcionalidades e ferramentas que não são encontradas nos produtos existentes no mercado (ver [ICME 2006] e [9]).

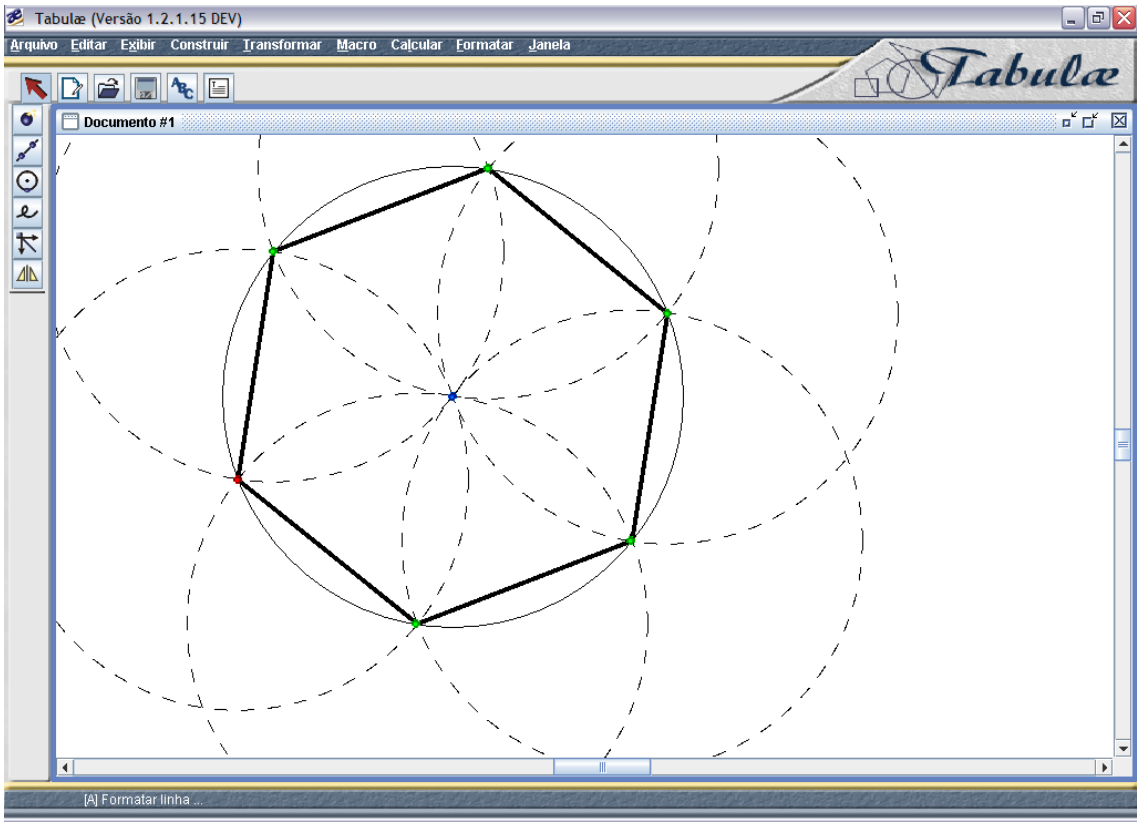

Figura 1. Um hexágono feito no Tabulæ.

Dentre as funcionalidades interessantes do programa, em relação a similares no mercado, podemos citar:

1. A interface gráfica permite a escolha entre os modos verbo-nome e nome-verbo.
2. Escrito em Java, o Tabulæ é compatível com diversas plataformas.
3. A programação é inteiramente orientada a objeto com o núcleo matemático e interface gráfica completamente separadas no programa.
4. O Tabulæ pode gerar código em Java, o que se torna útil na produção de hipertextos.
5. Algoritmos otimizados para lugares geométricos
6. O design de interface foi elaborado baseado em princípios ergonômicos.
7. Pode-se gerar relatórios detalhados de uso dos alunos.
8. Pode-se compartilhar construções através da Internet, facilitando a aprendizagem colaborativa.

3. A implantação da macro em LOGO

LOGO é uma linguagem de programação usada há algumas décadas - muitas crianças tiveram seu primeiro contato com um computador ou uma linguagem de programação através de alguma implementação da linguagem. Seu uso foi disseminado, com milhões de usuários por todo o mundo (Papert, 1993). Uma das principais aplicações didáticas do LOGO se dá justamente no ensino de geometria, através de um conceito computacional que Papert define como “Geometria da Tartaruga”, na qual o usuário comanda o computador para elaborar construções geométricas traduzidas em um programa da linguagem.

As construções geométricas elaboradas nas implementações da linguagem não possuem o caráter de manutenção de propriedades dos programas de geometria dinâmica. A “tartaruga” do LOGO faz, na verdade, um desenho e não uma construção, já que os objetos gerados não possuem propriedades geométricas associadas a eles nem tampouco essas propriedades se mantém quando os objetos são manipulados.

Usar o LOGO como linguagem de macro em um programa de geometria dinâmica abre a possibilidade de uma grande sinergia entre os potenciais didáticos das duas ferramentas. Para os usuários de LOGO traz a possibilidade de elaborar construções nas quais as propriedades geométricas se mantém quando elementos são manipulados, além de dar acesso uma enorme quantidade de métodos e comandos típicos da GD. Para os usuários de GD, por sua vez, o uso de uma linguagem de programação expande as possibilidades de criação de macros mais flexíveis, poderosas e com maior controle do usuário. Além disso, a reunião de duas poderosas ferramentas de ensino facilita a interação e troca de experiências didáticas entre profissionais que utilizem esta ou aquela ferramenta.

Para implementar a macro em LOGO, foi necessário, primeiramente, escolher quais primitivas e métodos do programa Tabulæ seriam utilizadas na macro. Foi feito um encapsulamento das principais funções básicas do software Tabulæ. Dessa forma, o programa poderia utilizar livremente estas funções já em funcionamento sem se preocupar com sua implementação. Por exemplo, não é necessário saber como o Tabulæ cria um ponto e sim saber qual função encapsulada que cria um ponto. Denominamos essa implantação de TABULOGO.

Depois disto feito, o próximo passo foi conceber como seria a linguagem dessa macro. Ela deveria obedecer aos princípios básicos do LOGO e primar pela simplicidade. Características como a tipagem fraca (isto é, não é necessário declarar explicitamente o tipo de uma variável e este tipo pode mudar durante o programa), foram mantidas (Sebesta, 2005). Além disso, foi preciso estender a linguagem para utilizar as possibilidades da geometria dinâmica. Essa é uma grande diferença de nosso trabalho, já que o LOGO tradicional trata os pontos e os objetos geométricos desenhados como apenas pixels. Na nossa implementação, os objetos geométricos preservam suas propriedades e podem ser manipulados através de nossa linguagem.

Para implementar uma linguagem de programação, como a do TABULOGO, é necessário descrever uma linguagem livre de contexto, ou seja, definida através de uma gramática, que descreve as regras sintáticas de como a linguagem se comporta (Sebesta, 2005). Em outras palavras, é a gramática que define a sintaxe dos comandos de uma linguagem de programação. Então essa gramática foi escrita baseada nas especificações da linguagem e nos comandos que gostaríamos de implementar.

Outra questão é, como a partir de um texto identificar quais são os comandos contidos neles para depois executá-los. Por exemplo, como reconhecer que a palavra Reta corresponde ao comando que cria uma reta? Um parser é um programa que tem justamente a finalidade de transformar essas palavras em objetos (ou entidades) que são entendidas pelo computador. Utilizamos um programa chamado Javacc, que a partir da nossa gramática gerou esse parser escrito na linguagem Java. Restava agora associar aos objetos às ações que gostaríamos que realizassem.

Freqüentemente em programação, utilizamos padrões de projeto. Padrões de projeto descrevem soluções clássicas para problemas usuais (Gamma et al., 1995). No nosso caso, o problema era trabalhar com objetos que eram comandos e para isso existe um padrão chamado Command. Este padrão sugere uma estrutura na qual cada objeto saiba se executar e por isso todos objetos de sua classe compartilham uma mesma função de execução. Este padrão ainda tem uma vantagem de ser de fácil integração com o programa Javacc e por isso foi utilizado com sucesso. Assim, finalizou-se o processo de implementação da linguagem.

Foram implementados todos elementos presentes em uma linguagem de programação, como comandos de controle (Se), repetição (Repita n Vezes) e chamadas de sub-rotinas. Também estão presentes os comandos clássicos de LOGO, como Direita, Esquerda, Anda, mantendo a metáfora da Tartaruga. Completam a lista os comandos derivados do Tabulæ, como Reta, Vetor, Translacao e etc. A entrada e saída de dados dos programas são feitas com componentes gráficos do próprio Java e podem ser vistas em LeNumero.

Posteriormente, foi acrescida à linguagem a possibilidade do uso de ouvintes. Eles são úteis em uma situação em que se escreve um programa e deseja-se que ele se comporte de maneiras que são sujeitas a condições. No entanto, como estamos lidando com geometria dinâmica, estas condições variam de acordo com a vontade do usuário, a qualquer momento que ele venha a manipular objetos na tela, e não somente no momento da execução do programa de macro. Então, se registra essa condição em um ouvinte e os comandos que devem ser executados caso ela seja verdadeira ou falsa. A cada mudança do estado do programa, testa-se novamente a condição e o ouvinte recebe o resultado. De acordo com este resultado comandos são executados ou não. Isto pode ser feito para uma ou mais condições.

4. Possibilidades de construção

Passamos a apresentar um estudo com três casos de construções possíveis de serem feitas com o TABULOGO:

1. Construção de uma tangente comum a dois círculos
2. Uma Curva de Koch
3. Uma Rosácea

Os programas se encontram no Anexo.

<u>Tangente comum a dois círculos</u>

A construção da tangente comum a dois círculos dados, C1 e C2 (figura 2) é muito difícil de ser executada com as macros usuais, já que são poucos os programas de geometria dinâmica apresentam a possibilidade de construções condicionais.. Isto ocorre em razão da construção demandar o conhecimento de qual dos dois círculos possui o menor diâmetro e esta informação poder mudar conforme o usuário manipule um dos dois círculos envolvidos.

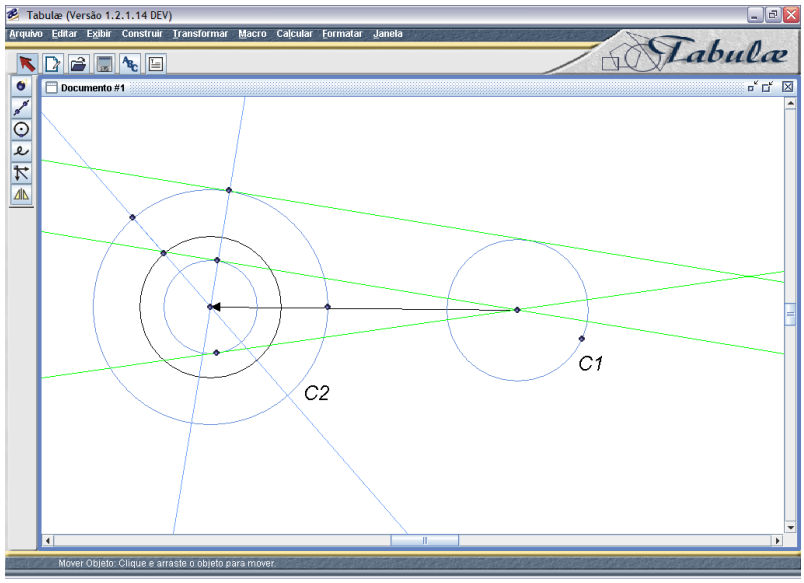

Figura 2: Uma tangente comum a dois círculos dados depende da determinação daquele que possui o menor raio.

Para resolver este problema, utiliza-se um ouvinte que fica monitorando os diâmetros dos dois círculos. Toda vez em que ocorre uma mudança no círculo de maior tamanho, o programa redesenha a construção[1].

Cuva de Koch

A Curva de Koch, concebida em 1904, é um exemplo de curva contínua sem derivada em nenhum ponto, que pode ser concebída de maneira elementar (Figura 3).

[1] Evidentemente, poderíamos, no caso deste problema, ter adotado uma alternativa de construção que não dependesse da escolha do círculo de maior raio. Usamos esta apenas como ilustração, e por ser a mais comum em livros texto.

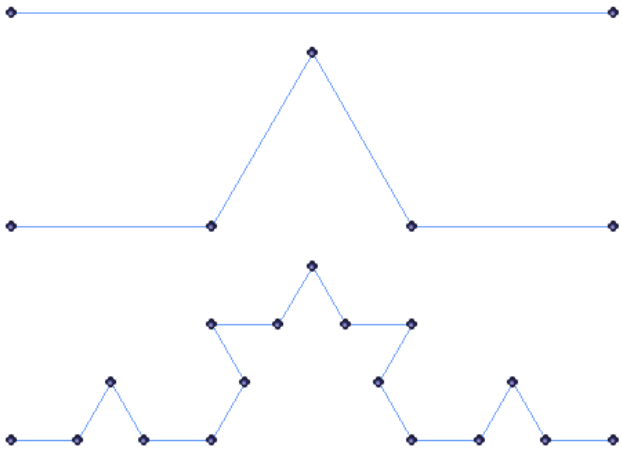

Figura 3: Os primeiros passos para elaboração da Curva de Koch

A sua elaboração depende do uso de ferramentas que possibilitem a construção recursiva de triângulos eqüiláteros no terço central de um lado construído na etapa anterior.

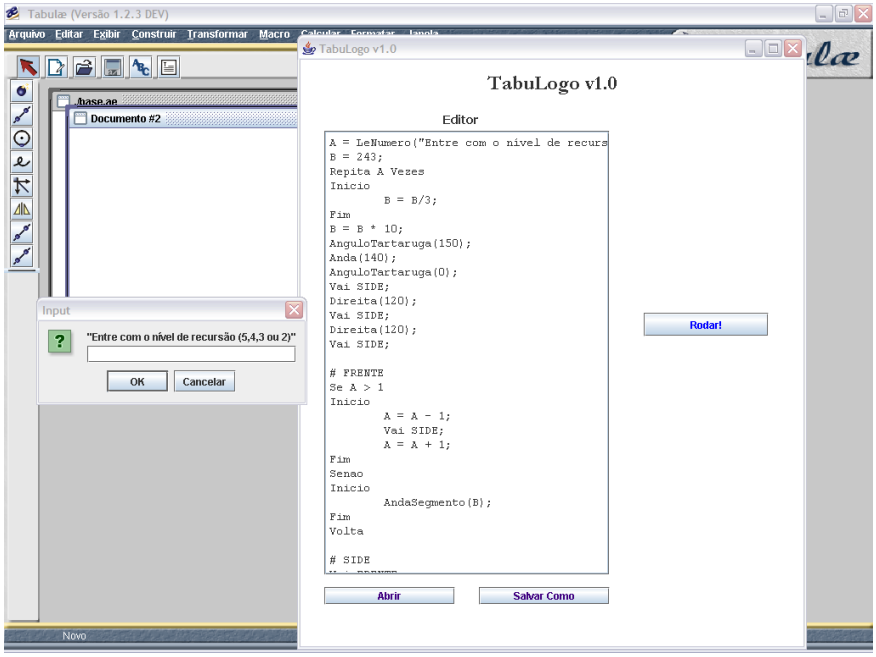

Figura 4: Editor LOGO com o programa gerador de uma Curva de Koch e uma caixa de diálogo pedindo o nível de recursão para geração da curva.

O processo de construção da Curva de Koch com o Tabulogo demanda apenas a inserção do nível de recursão por parte do usuário em uma caixa de diálogo (Figura 4). A partir daí, as iterações são controladas pelo próprio programa, gerando-se a curva que vemos na figura 5:

A utilização da ferramenta de geometria dinâmica permite expandir e contrair a figura se ela depender de um segmento, por exemplo. Além disso, a manipulação da construção mantém as propriedades da figura.

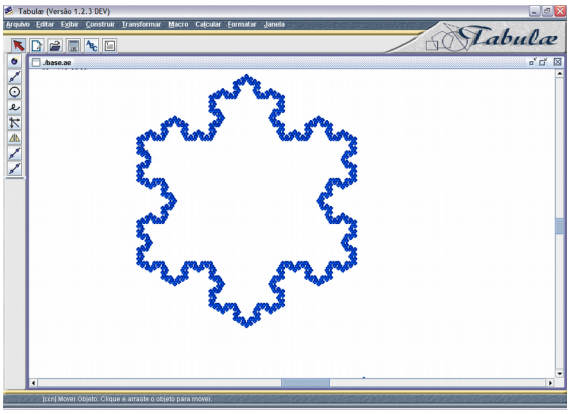

Figura 5: A Curva de Koch gerada pelo programa em LOGO.

Rosácea

Rosáceas e espirais são construções simples e belas de serem feitas em linguagem LOGO. Sua construção em programas de geometria dinâmica, por sua vez é muito complicada e exige o conhecimento de conceitos mais avançados como funções parametrizadas e coordenadas polares. Para geração da rosácea da figura 6, buscamos adaptar o código de exemplos de um concurso de figuras elaboradas na linguagem [2].

Da mesma forma que a Curva de Koch do exemplo anterior, a Rosácea da figura 6 também pode ser manipulada com manutenção das suas propriedades geométricas. Uma pequena modificação no programa permite que o comprimento de um segmento seja usado como parâmetro de tamanho da figura.

[2] No endereço: http://www.mathcats.com/gallery/15wordcontest.html

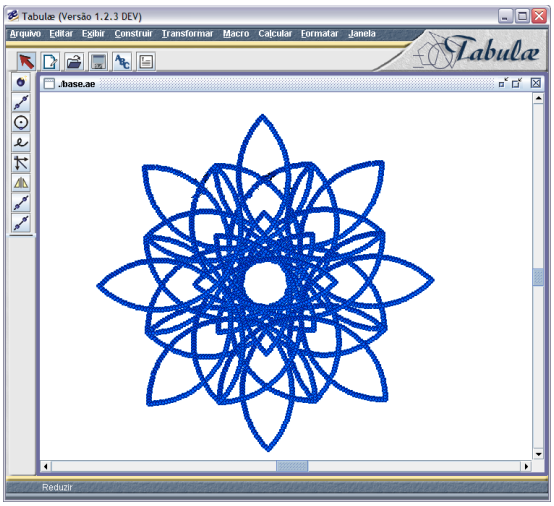

Figura 6: Rosácea

5. Conclusão

Apresentamos neste trabalho uma implementação de LOGO como linguagem de macro de um programa de geometria dinâmica. A utilização da linguagem permite a elaboração de macros mais flexíveis e de construções com maior complexidade. Por outro lado, os usuários da linguagem LOGO passam a poder gerar construções com dependência geométrica entre os objetos produzidos. No momento, estamos realizando outros estudos para determinar mais possibilidades didáticas de uso e fazendo testes com usuários.

Referências

Ainley, J., Pratt, D., Hansen, A. (2006). Connecting engagement and focus in pedagogic task design, *British Educational Research Journal*. 32.1, 21-36.

Bellemain, F. (1992) *Conception, réalisation et expérimentation d'un logiciel d'aide à l'enseignement de la géométrie Cabri-géomètre* Tese de Doutorado. Obtida em http://bibliotheque.imag.fr/publications/theses/1992/Bellemain.Franck/these.dir/these.pdf Acesso em 27/02/2007.

Cabariti, E., Jahn, A. P. (2006). A Geometria Hiperbólica na Formação Docente: possibilidades de uma proposta com o auxílio do Cabri-géomètre. *In: III*

Seminário Internacional de Pesquisa em Educação Matemática - SIPEM, 2006, Águas de Lindóia. Anais do III Seminário Internacional de Pesquisa em Educação Matemática. São Paulo : SBEM, v. 1. p. 18-28

Felsager, B., Gymnasium H., Denmark HF (2004), Introducing Minkowski-geometry using Dynamic Geometry Programs *In. ICME-10*, Copenhagen. Obtido em http://descartes.ajusco.upn.mx/varios/tsg10/articulos/Felsager_2_re_revised_paper__2_.doc Acesso em 27/02/2007.

Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. (1995) *Design Patterns: Elements of Reusable Object-Oriented Software* Addison-Wesley, New York.

Guimarães, L. C., Barbastefano, R. G., Carvalho, D.(2001) Tabulæ, Registro INPI n.0039192.

Papert, S. (1993) *Mindstorms: Children, Computers and Powerful Ideas 2ed.* Perseus Books, New York.

Pratt, D., Ainley, J. (1997). The Construction of Meanings for Geometric Construction : Two Contrasting Cases, *International Journal of Computers for Mathematical Learning*, 1.3, 293-322.

Sebesta, R.W. (2005) *Concepts of Programming Languages 7th edition*. Addison Wesley, New York.

Schuman, H. (1989). The influence of interactive tools in geometry learning. *In: Intelligent learning enviroments, the case of geometry*, Berlim, Springer-Verlag.

Anexo – Programas em LOGO utilizados

Tangente comum a dois círculos

```
c1 = RecuperaObjeto("C1");
c2 = RecuperaObjeto("C2");
```

```
r1 = Raio(c1);
r2 = Raio(c2);
Se r1 > r2
Inicio
        maior = c1;
        menor = c2;
    Ouvinte("Monitora Tangente 1");
Fim
Senao
Inicio
        maior = c2;
        menor = c1;
    Ouvinte("Monitora Tangente 2");
Fim

p1 = CentroCirculo(menor);
p2 = CentroCirculo(maior);

v1 = Vetor(p1,p2);

c3 = Translacao(v1,menor);
p3 = CentroCirculo(c3);
pm = CentroCirculo(menor);

rt =  RetaTangenteCirculo(c3,pm);
p4 = PontoTangente1(rt);
r3 = Reta(p3, p4) ;
r4  = RetaParalela(r3, pm);
irc1 = InserseccaoRetaCirculo(r3,maior);
irc2 = InterseccaoRetaCirculo(r4,menor);

retatangente = Reta(p6,p8);
```

```
# Monitora Tangente 1

c1 = RecuperaObjeto("C1");
c2 = RecuperaObjeto("C2");
r1 = Raio(c1);
r2 = Raio(c2);
Se r1 < r2
Inicio
        Chama DestroiTangente;
        Chama ConstroiTangente;
Fim

# Monitora Tangente 2

c1 = RecuperaObjeto("C1");
c2 = RecuperaObjeto("C2");
r1 = Raio(c1);
r2 = Raio(c2);
Se r1 > r2
Inicio
        Chama DestroiTangente;
        Chama ConstroiTangente;
Fim

Curva de Koch

A = LeNumero("Entre com o nível de recursão (5,4,3 ou 2)");
B = 243;
Repita A Vezes
Inicio
        B = B/3;
Fim
B = B * 10;
AnguloTartaruga(150);
```

```
Anda(140);
AnguloTartaruga(0);
Vai SIDE;
Direita(120);
Vai SIDE;
Direita(120);
Vai SIDE;

# FRENTE
Se A > 1
Inicio
        A = A - 1;
        Vai SIDE;
        A = A + 1;
Fim
Senao
Inicio
        AndaSegmento(B);
Fim
Volta

# SIDE
Vai FRENTE;
Esquerda(60);
Vai FRENTE;
Direita(120);
Vai FRENTE;
Esquerda(60);
Vai FRENTE;
Volta

Rosácea

Repita 8 Vezes
```

```
Inicio
Direita(45);
        Repita 6 Vezes
        Inicio
        Repita 90 Vezes
        Inicio
                AndaSegmento(10);
                Direita(2);
        Fim
        Direita(90);
        Fim
Fim
```