Primeiro Exercício-Programa

Estrelas de Koch, H-trees e mais...

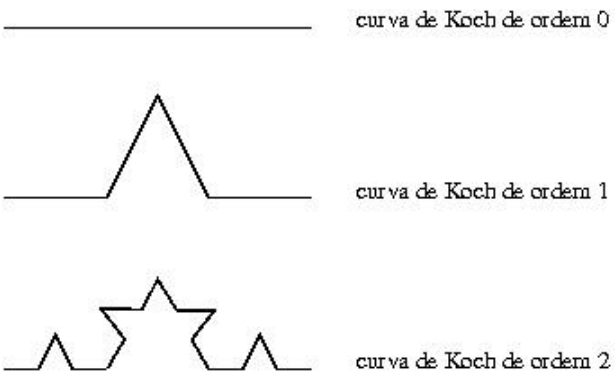
Entrega: 13 de setembro

O objetivo deste exercício-programa é utilizar recursão para gerar alguns fractais simples. Fractais são figuras geométricas que podem ser divididas em partes semelhantes à figura original. Por isso, a recursão é um recurso muito apropriado para gerar tais figuras.

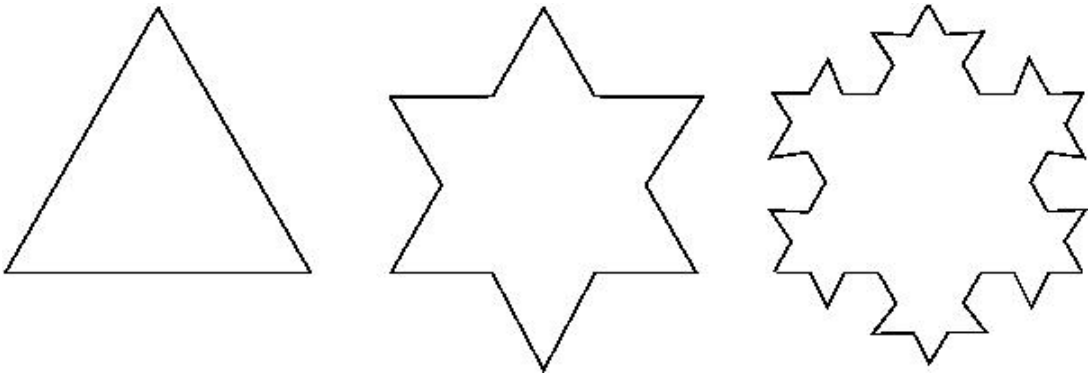
Começaremos apresentando dois fractais simples: as curvas de Koch e as árvores H (*H trees*). Você pode ler mais sobre eles na wikipedia:

- Curva de Koch
- H tree

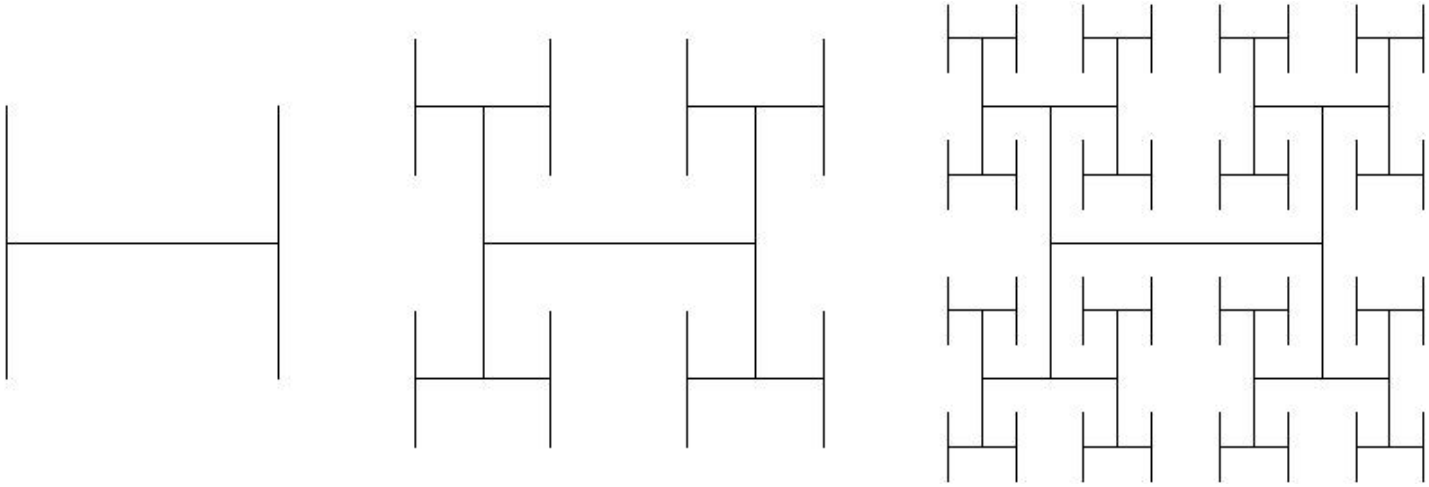
As curvas de Koch são definidas da seguinte maneira. A curva de Koch de ordem 0 consiste em um segmento de reta. A curva de Koch de ordem 1 é obtida a partir da curva de Koch de ordem 0 subdividindo o segmento em três segmentos de mesmo comprimento, e substituindo a parte do meio por dois outros segmentos do comprimento da parte do meio, formando um bico (como dois lados de um triângulo equilátero), como indicado pela figura abaixo. Esse processo se repete a cada segmento da curva de Koch de ordem 1 para obtermos a curva de Koch de ordem 2, etc.



Uma variante da curva de Koch é a chamada estrela de Koch, que é obtida de três curvas de Koch cada uma obtida a partir de um dos lados de um triângulo equilátero.



As árvores H por sua vez são definidas da seguinte forma. A árvore H de ordem 0 é mostrada na figura abaixo. A árvore H de ordem 1 é obtida da de ordem 0 adicionando-se uma árvore H de ordem 0 com a metade do seu tamanho, centrada em cada uma de suas folhas. Similarmente, uma árvore H de ordem 2 é obtida da de ordem 0 adicionando-se uma árvore H de ordem 1 com a metade do seu tamanho, centrada em cada uma de suas folhas, etc.



Estes são dois exemplos de construções recursivas. Você está convidado a pensar em uma construção sua, que dê origem a uma família de figuras semelhante a estas.

Entrada e saída do seu programa

Você deverá escrever um programa que:

1. Leia do teclado um número, onde 0 indica estrela de Koch, 1 indica árvore H e 2 indica o seu fractal.
2. Leia do teclado a ordem da figura a ser gerada.
3. Leia do teclado o nome de um arquivo de saída que conterá a imagem gerada.
4. Gere um arquivo postscript com a estrela de Koch, a árvore H ou o seu fractal, de acordo com a ordem dada. A imagem deve ter um tamanho semelhante às disponibilizadas abaixo.

Não modifique a ordem dos dados na entrada e nem faça seu programa ler algo além dos dados acima, pois o seu programa será submetido a um corretor automático. Haverá um desconto na nota do seu EP se for feita alguma modificação no formato de leitura.

Exemplos de saída

Aqui você encontra alguns arquivos pdf que foram obtidos dos arquivos ps gerados pelo meu programa.

Linguagem postscript

Postscript é uma linguagem completa cujos programas estão escritos em notação posfixa e são interpretados com a ajuda de uma pilha interna, exatamente como no cálculo de expressões aritméticas em notação posfixa. Cada operação obtém seus argumentos da pilha interna e deposita de volta nesta pilha seu valor de retorno.

Nesta seção, descreveremos apenas o necessário desta linguagem para que você possa implementar esse EP. Esta descrição baseia-se numa parte da seção 4.3 do livro do Sedgewick, Algorithms in C.

Os operadores aritméticos básicos em postscript são *add*, *sub*, *mul*, *div* (divisão em float) e *idiv* (divisão inteira). Um exemplo de programa em postscript mostra como se pode utilizar estes operadores:

5 9 8 add 4 6 mul mul 7 add mul

Sim, esta linha é um programa em postscript! Interprete a linha acima como uma expressão em notação posfixa e pense em qual é o valor desta expressão. Você deve obter o valor 2075. Confira! Assim, ao final desse programa em postscript, este valor é o que fica na pilha interna.

Além destes operadores aritméticos, você pode achar útil o operador *neg*, que reverte o sinal do número no topo da pilha.

A linguagem postscritp (ps) tem um número de funções primitivas que servem de instrução para um dispositivo abstrato para desenhar. A ação de cada função é em geral clara do seu nome. Estas funções são acionadas com argumentos que estejam na pilha. Por exemplo,

0 0 moveto 144 0 rlineto 60 rotate 144 0 rlineto stroke

corresponde à seguinte sequência de ações: chame *moveto* com argumentos 0 e 0, então chame *rlineto* com argumentos 144 e 0, depois *rotate* com argumento 60, etc. Algumas funções se referem diretamente à própria pilha. Por exemplo, o operador *dup* duplica a entrada do topo da pilha. Assim, por exemplo, o código

144 dup 0 rlineto 60 rotate dup 0 rlineto

corresponde a: chame *rlineto* com argumentos 144 e 0, depois *rotate* com argumento 60, então *rlineto* com argumentos 144 e 0. Um outro operador que atua na pilha diretamente é o *pop*, que remove o elemento do topo da pilha (desempilha).

A seguir, descrevemos com precisão a ação das funções mais básicas, que devem ser suficientes para você gerar as figuras pedidas neste EP. Antes disso, vamos descrever o dispositivo abstrato para desenhar. O dispositivo consiste de uma página virtual, inicialmente em branco, com um sistema de coordenadas. A origem deste sistema fica no canto inferior esquerdo da página, o eixo x é o horizontal e o eixo y, o vertical. Além da página, temos uma caneta virtual.

A operação *moveto* remove dois números da pilha e trata-os como as coordenadas x e y para onde move a caneta. A operação *rmoveto* é como o *moveto*, porém o movimento é relativo à posição corrente da caneta. Por exemplo,

100 50 moveto 10 20 rmoveto

faz com que a caneta primeiro mova para a posição (100,50), e depois para a posição (110,70). A operação *rotate* remove um número da pilha e executa uma rotação do sistema de coordenadas deste número de graus no sentido anti-horário. Assim, após

100 50 moveto 90 rotate 10 20 rmoveto

a caneta estará na posição (80,60).

Até agora, não desenhamos nada. Apenas movemos a caneta. O operador *lineto* remove dois números da pilha, tratando-os como as coordenadas x e y de um ponto, e registra um segmento de reta do ponto onde está a caneta para o ponto (x,y). A caneta termina no ponto (x,y). O operador *rlineto* é semelhante, mas interpreta os dois números desempilhados como a posição relativa à posição corrente. Vejamos um exemplo para entender melhor. Neste ponto, eu sugiro que você copie o programa ps abaixo em um arquivo texto de nome desenho.ps e visualize o resultado abrindo este arquivo, por exemplo, com o programa ghostview (gv) ou um equivalente.

150 250 moveto 200 400 lineto stroke

O operador *stroke* conclui e imprime o desenho. Experimente alternativamente

150 250 moveto 200 400 rlineto stroke

e perceba a diferença. No primeiro, o segmento desenhado e a caneta terminam na posição (200,400), enquanto que no segundo, terminam na posição (350,650).

Além destes, um comando que pode ser útil para manipular a pilha é o *roll*. Este comando remove dois números da pilha, digamos, a primeiro, depois b, e rotaciona circularmente a vezes os b últimos números da pilha. Por exemplo, ao final do programa

7 8 9 3 1 roll

a pilha está com **9 7 8**. Se a for negativo, a rotação é no sentido oposto. Assim, ao final de

7 8 9 3 -1 roll

a pilha está com **8 9 7**.

Para mais informações sobre a linguagem postscript, consulte o seu manual completo. No entanto, antes de usar no seu EP qualquer recurso da linguagem que não esteja entre os apresentados acima, por favor, me consulte.

Recomendações

Utilize recursão no seu programa para gerar o arquivo ps dos desenhos. Se você não utilizar recursão, o seu EP receberá nota zero.

Comece brincando um pouco com postscript. Escreva um programa ps que desenhe a curva de Koch de ordem 1. Uma vez que você tenha feito isso, tente escrever uma função recursiva em C que receba como parâmetro um inteiro $n \geq 0$ e imprima um arquivo ps que desenhe a curva de Koch de ordem n. Depois disso, será fácil obter a estrela de Koch.

Passe para a árvore H apenas depois de ter conseguido gerar o arquivo da estrela de Koch. A árvore H dá um pouco mais de trabalho.

Para o seu fractal, você está liberado para usar outras funções da linguagem postscript, como por exemplo funções que desenham arcos, ou círculos, etc. Divirta-se xeretando o manual.

Há recursão na linguagem postscript, mas não a use para desenhar a estrela de Koch ou a árvore H. Se quiser, experimente no seu fractal.

Divirta-se com o EP!

Entrega, prazos e observações

- O EP é estritamente individual.
- O EP é estritamente individual. Usaremos um sistema para detecão de EPs copiados.
- O EP é estritamente individual. Alunos envolvidos em cola serão reprovados na disciplina sem mais, e seus nomes serão encaminhados para a Comissão de Graduação para demais providências.
- Exercícios atrasados não serão aceitos.
- Capriche na documentação do seu programa.
- Coloque o cabeçalho usual em seu programa (como em MAC110), com nome, número USP, curso, data, nome do professor e identificação do EP (EP1)...
- A entrega do programa consiste da **entrega eletrônica** do programa. O seu `programa` deve ser entregue até dia 13 de setembro, inclusive.
