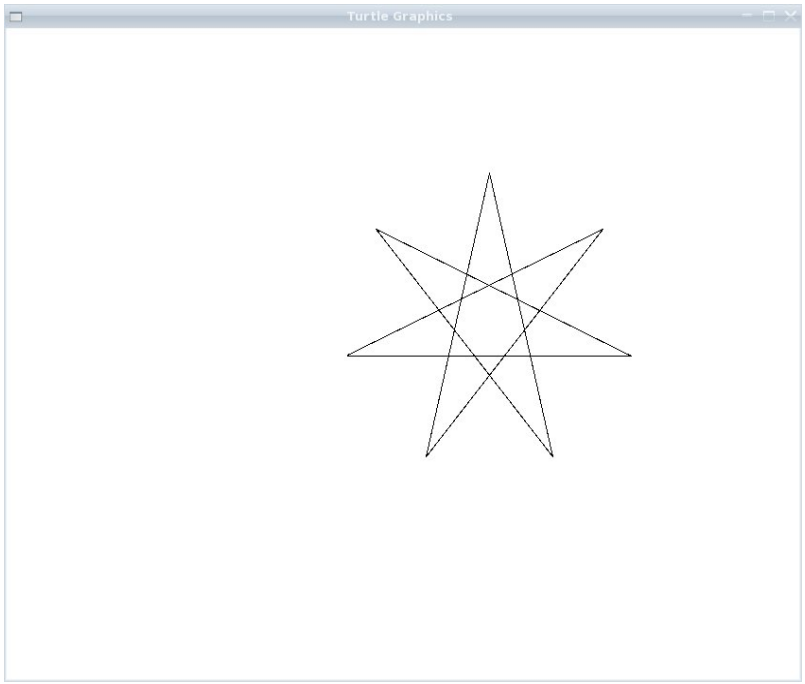


```
-------------------------------------------------------
Nesta aula: Uso a 2 módulos gráficos

  1) Módulo "turtle" (já usado em aulas anteriores)
  2) Módulo "Tkinter" - processamento de uma imagem a
     preto e branco
  3) Módulo "Tkinter" - processamento de uma imagem a
     cores
=======================================================
```

1 Estrelas desenhadas com o "turtle"

```
=====================================================

Problema: desenhar uma estrela com n lados,
  como na figura para n=7.


from turtle import *

speed("fastest")

def poli(n,a):
    """ desenhar uma estrela com n lados;
        o comprimento de cada lado é a
    """
    ang=180 - (360.0/n)/2
    down
    for i in range(n):
        forward(a)
        left(ang)
    up()
    forward(1000)


poli(7 ,300)
raw_input()


=====================================================
```

2 Desenho de imagens a preto e branco com o 'Tkinter' – Redução de contraste

```
====================================================

Possível representação de uma imagem
 ---------------------------------------------------------------
  - Uma "imagem" a preto e branco (PB) é uma lista de linhas,
    cada linha é uma lista de valores entre 0 (preto)
    e 255(branco)

    Uma "imagem" a cores é um tuplo
                    (r,g,b)
    onde r,g,b sao imagens a preto e branco
 ---------------------------------------------------------------
  Funcoes usadas:
  redc:      reduz o contraste de uma imagem a preto e branco
  pbimagem:  mostra uma imagem a preto e branco
  cimagem  - mostra uma imagem a cores
  grey, dig16, color: funcoes auxiliares
 ---------------------------------------------------------------

from Tkinter import *   # modulo grafico
from random import *
from copy import *

 ------------------------------------------------
    converter inteiro para string - dígito hexadecimal
 ------------------------------------------------
def dig16(n):
    "n de 0 a 15  ->  digito hexadecimal"
    return "%x"%(n)

 ------------------------------------------------
   gerar uma "imagem" aleatória
 ------------------------------------------------
#--- -> imagem aleatória lado * lado
def lrand(lado):
    return [[randint(0,255) for a in range(0,lado)] for b in range(0,lado)]




 ------------------------------------------------
  reduz o contraste de uma 'imag' quadrada
 ------------------------------------------------
```

```
def redc(im):
    n=len(im)
    c=deepcopy(a)
    for y in range(1,n-1):     # 1 a n-2
        for x in range(1,n-1): # 1 a n-2
            c[x][y]=(im[x][y]+im[x-1][y]+im[x+1][y]+im[x][y-1]+im[x][y+1])/5
    return c
 ------------------------------------------------------
          desenha uma imagem a preto e branco
  list: [[linha 0],[linha 1],...,[linha n]]
  ps: pixel size, cada ponto é representado por um quadrado
  de dimensoes ps*ps
 ------------------------------------------------------
def pbimagem(lista,ps=30):
    """ desenha o rectangulo composto de quadrados, 'lista';
        cada quadrado tem 'ps' de lado
    """
    nx=len(lista[0])      # n. de colunas
    ny=len(lista)         # n. de linhas
    dx=nx*ps              # lado horizontal (total dos quadrados)
    dy=ny*ps              # lado vertical (total dos quadrados)
    main=Tk()
    w=Canvas(main,width=dx,height=dy)
    w.pack()
    for y in range(ny):   # desenhar os quadrados
        y0=dy-y*ps-ps
        for x in range(nx):
            col=grey(lista[y][x])
            x0=x*ps
            w.create_rectangle(x0,y0,x0+ps,y0+ps,fill=col)
    mainloop()

a=lrand(3)
print a
a=[[6,0,3],[8,0,8],[6,1,5]]
print a
print redc(a)

a=lrand(50)

pbimagem(redc(a),20)
=====================================================
```

3 Desenho de imagens a cores com o 'Tkinter'

```
==================================================

Lembrando...
 ------------------------------------------------------------
  - Uma "imagem" a cores é formada por 3 listas de linhas,
    3 imagens,
    como nas imagens a preto e branco
    --- é da forma (r,g,b) onde r,g,b sao imagens a preto e branco
 ------------------------------------------------------------
from Tkinter import *   # modulo grafico
from random import *
from copy import *

 ----------------------------------------------
    converter inteiro para string - digito hexadecimal
 ----------------------------------------------

     <Igual ao programa "preto e branco">

 ----------------------------------------------
   gerar uma "imagem" aleatoria
 ----------------------------------------------

     <Igual ao programa "preto e branco">

 ----------------------------------------------
  --- exercicio das aulas TP
  reduz contraste, imag tem que ser quadrada
 ----------------------------------------------

     <Igual ao programa "preto e branco">



 ----------------------------------------------
  (r,g,b): R, g b - mesmo formato que a preto e branco
     podiamos ter usado uma lista de lista de trios...
 ----------------------------------------------
def cimagem((r,g,b),ps):
    nx=len(r[0])      # n. de colunas
    ny=len(r)         # n. de linhas
    dx=nx*ps          # lado horizontal
    dy=ny*ps          # lado vertical
    main=Tk()
    w=Canvas(main,width=dx,height=dy)
```

```
    w.pack()
    for y in range(ny):
        y0=dy-y*ps-ps
        for x in range(nx):
            col=color(r[y][x],g[y][x],b[y][x])
            x0=x*ps
            w.create_rectangle(x0,y0,x0+ps,y0+ps,fill=col)
    mainloop()

def color(r,g,b):
    """ r,g,b: entre 0 e 255
        retorna string com r, g, b em hexadecimal
        RRGGBB
        Exemplo, vermelho escuro: 880000
    """
    red  =dig16(r/16)+dig16(r%16)
    green=dig16(g/16)+dig16(g%16)
    blue =dig16(b/16)+dig16(b%16)
    return "#"+red+green+blue


=====================================================
```

4 Exemplos

```
=====================================================


 --- teste a cores

lado=32
r=lrand(lado)        # vermelhos: lado*lado aleatorio, cada de 0 a 255
g=lrand(lado)        # verdes: lado*lado aleatorio, cada de 0 a 255
b=lrand(lado)        # azuis: lado*lado aleatorio, cada de 0 a 255

imag = (r,g,b)       # tuplo rgb
cimagem(imag,30)

rimag = (redc(r),redc(g),redc(b))
cimagem(rimag,30)    # imagem com o contraste reduzido
```