Exposure CS 2022 for CS1

Chapter 8 Section 6 Slides

Combining Repetition with Traditional Graphics

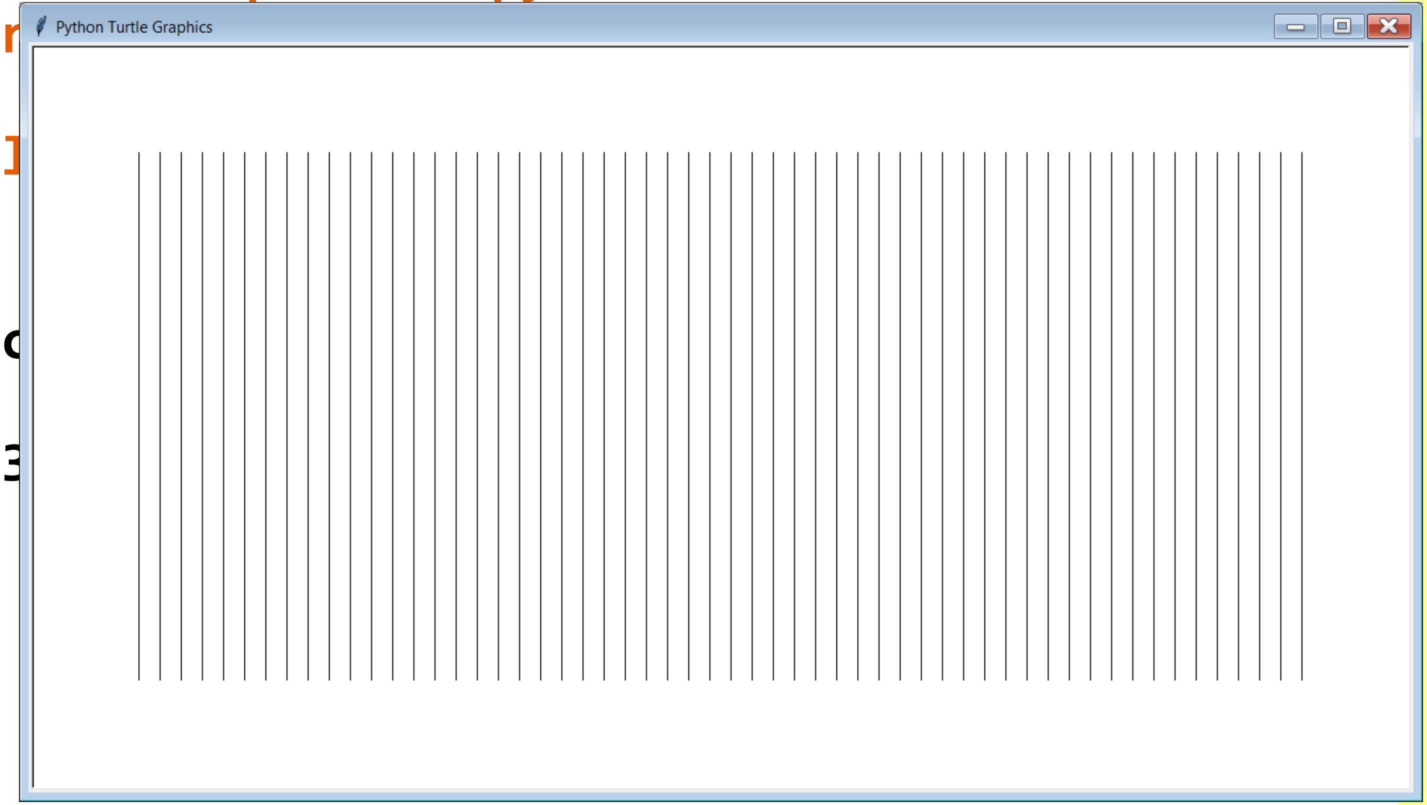
PowerPoint Presentation created by:
Mr. John L. M. Schram
and Mr. Leon Schram
Authors of Exposure
Computer Science

Section 8.6

Using Repetition with Traditional Graphics

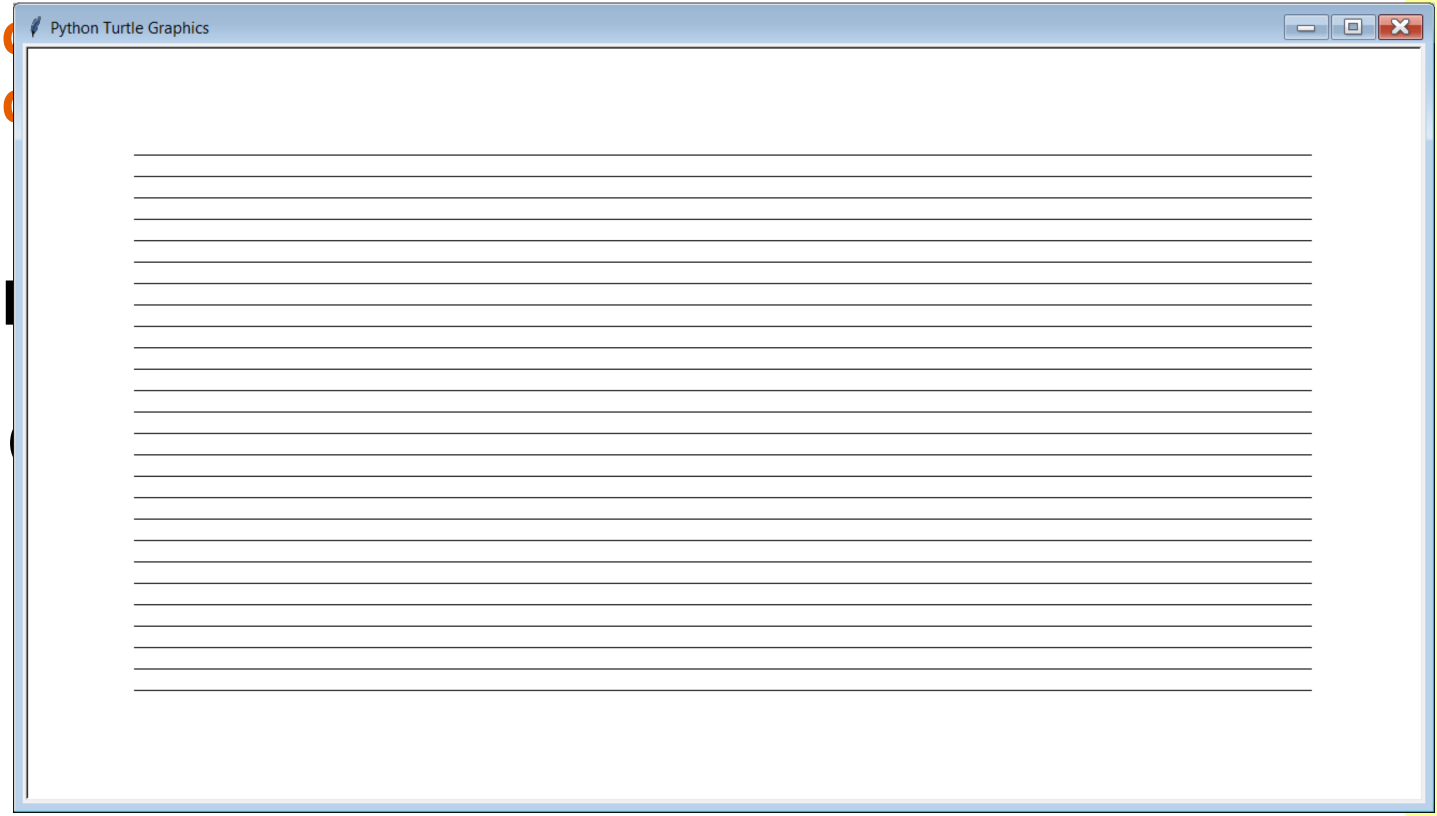
```python
# RepetitionWithGraphics04.py
# This program shows how a <for> loop can also be
# used with traditional graphics. This program draws
# vertical lines because <x1> and <x2> are always equal.


from Graphics import *

beginGrfx(1300,700)

x1 = 100
y1 = 100
x2 = 100
y2 = 600

for k in range(56):
    drawLine(x1,y1,x2,y2)
    x1 += 20
    x2 += 20

endGrfx()
```

```
# RepetitionWithGraphics04.py
# This progr
# used with
# vertical l


from Graphic

beginGrfx(13

x1 = 100
y1 = 100
x2 = 100
y2 = 600

for k in range(56):
    drawLine(x1,y1,x2,y2)
    x1 += 20
    x2 += 20

endGrfx()
```

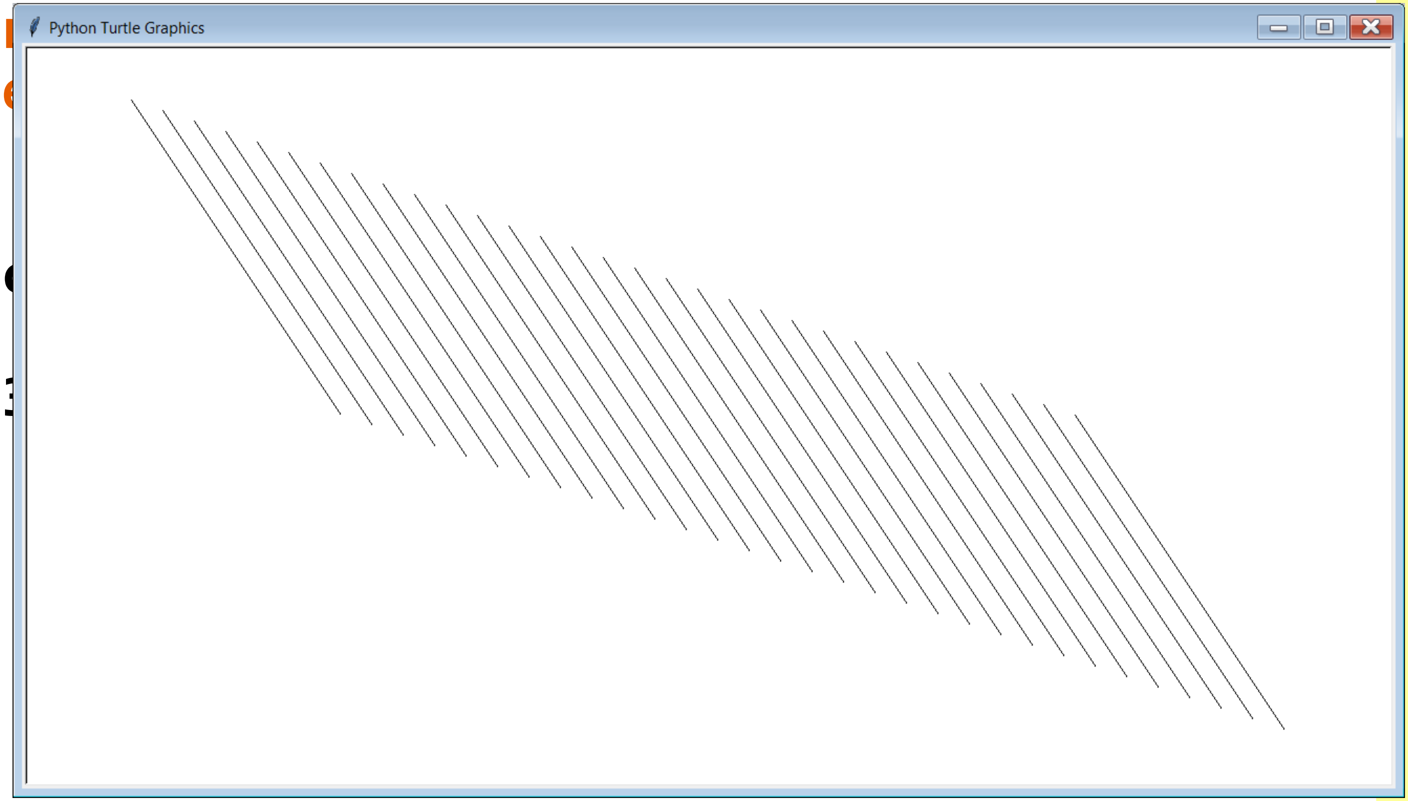


```python
# RepetitionWithGraphics05.py
# This program draws horizontal lines because
# <y1> and <y2> are always equal.


from Graphics import *

beginGrfx(1300,700)

x1 = 100
y1 = 100
x2 = 1200
y2 = 100

for k in range(26):
   drawLine(x1,y1,x2,y2)
   y1 += 20
   y2 += 20

endGrfx()
```

```
# RepetitionWithGraphics05.py
# This pr
# <y1> an


from Grap

beginGrfx(

x1 = 100
y1 = 100
x2 = 1200
y2 = 100

for k in range(26):
    drawLine(x1,y1,x2,y2)
    y1 += 20
    y2 += 20

endGrfx()
```



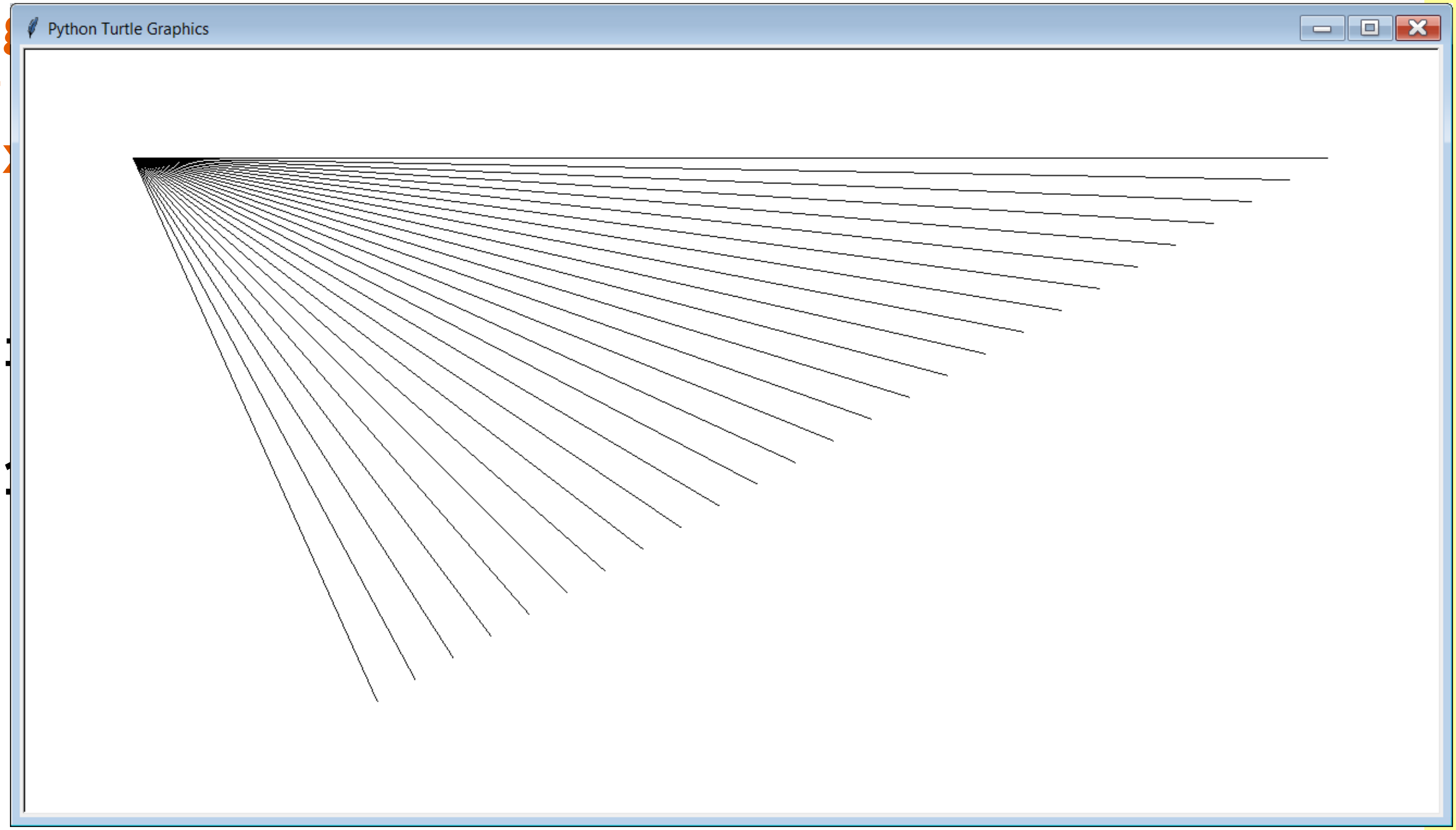
```python
# RepetitionWithGraphics06.py
# This program draws parallel diagonal lines
# and changes all 4 variables.


from Graphics import *

beginGrfx(1300,700)

x1 = 100
y1 = 50
x2 = 300
y2 = 350

for k in range(31):
   drawLine(x1,y1,x2,y2)
   x1 += 30
   x2 += 30
   y1 += 10
   y2 += 10

endGrfx()
```



```
# RepetitionWithGraphics06.py
# This progr
# and change


from Graphi

beginGrfx(13

x1 = 100
y1 = 50
x2 = 300
y2 = 350

for k in range(31):
    drawLine(x1,y1,x2,y2)
    x1 += 30
    x2 += 30
    y1 += 10
    y2 += 10

endGrfx()
```

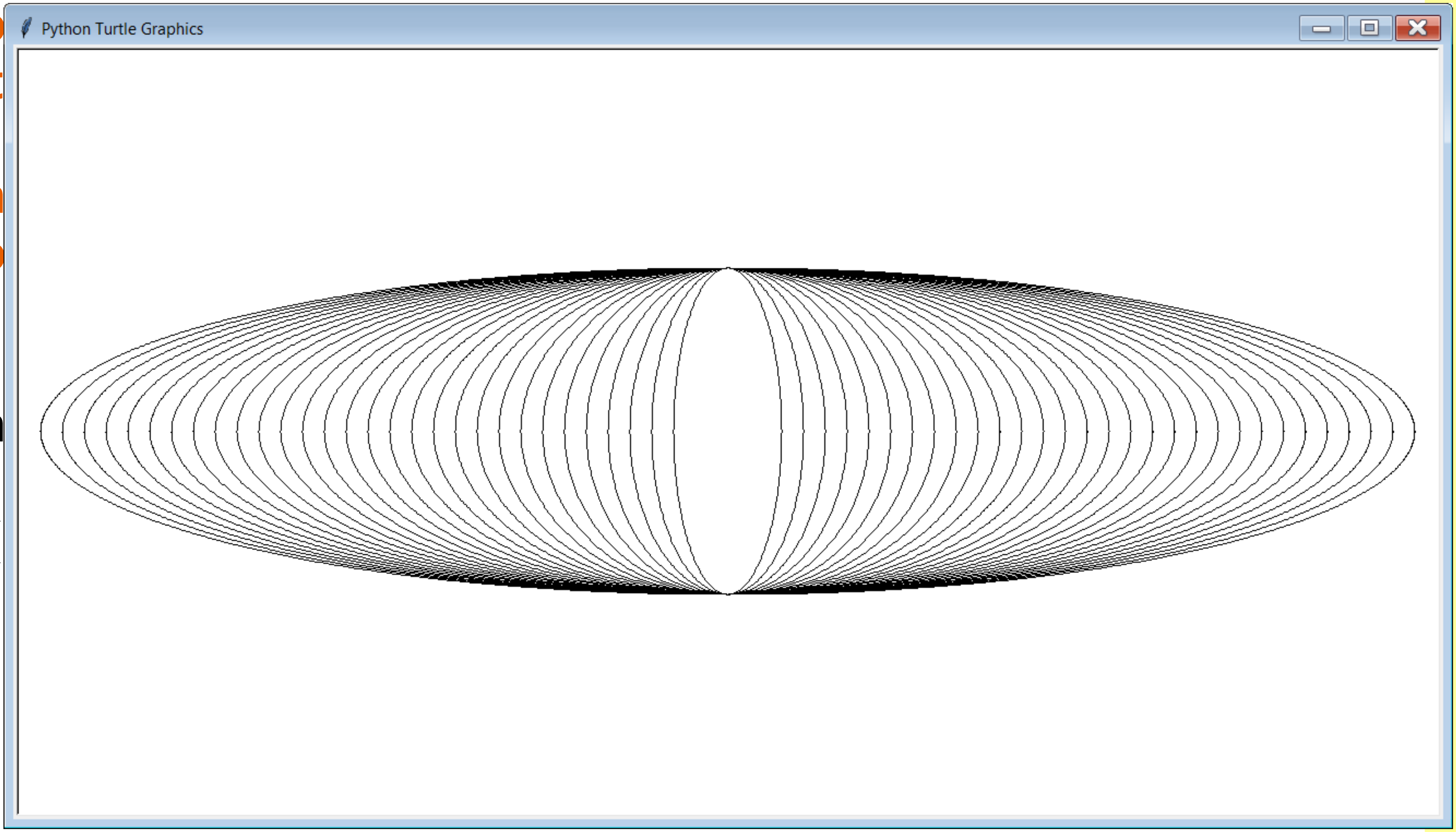
```python
# RepetitionWithGraphics07.py
# This program demonstrates several lines with the same
# starting point. In this case the (x1,y1) coordinate
# stays fixed while the (x2,y2) point changes.


from Graphics import *

beginGrfx(1300,700)

x1 = 100
y1 = 100
x2 = 1200
y2 = 100

for k in range(26):
   drawLine(x1,y1,x2,y2)
   x2 -= 35
   y2 += 20

endGrfx()
```

```
# RepetitionWithGraphics07.py
# This prog
# starting
# stays fi


from Graph

beginGrfx(

x1 = 100
y1 = 100
x2 = 1200
y2 = 100

for k in range(26):
    drawLine(x1,y1,x2,y2)
    x2 -= 35
    y2 += 20

endGrfx()
```



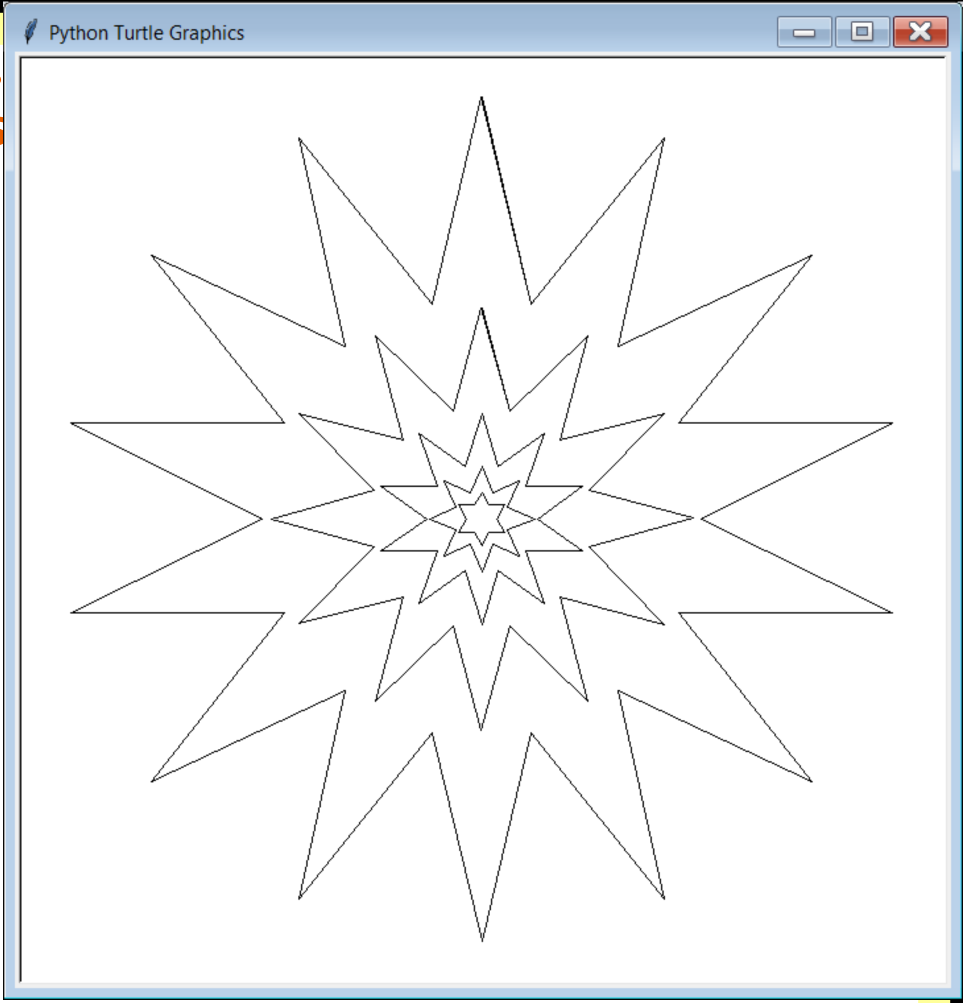
```
# RepetitionWithGraphics08.py
# This program demonstrates several ovals.
# All of the ovals have the same center and vertical
# radius.  The horizontal radius keeps growing.
# NOTE: There may be a little delay in the display
# of the output.


from Graphics import *

beginGrfx(1300,700)

x = 650
y = 350
hr = 50
vr = 150

for k in range(30):
    drawOval(x,y,hr,vr)
    hr += 20

endGrfx()
```

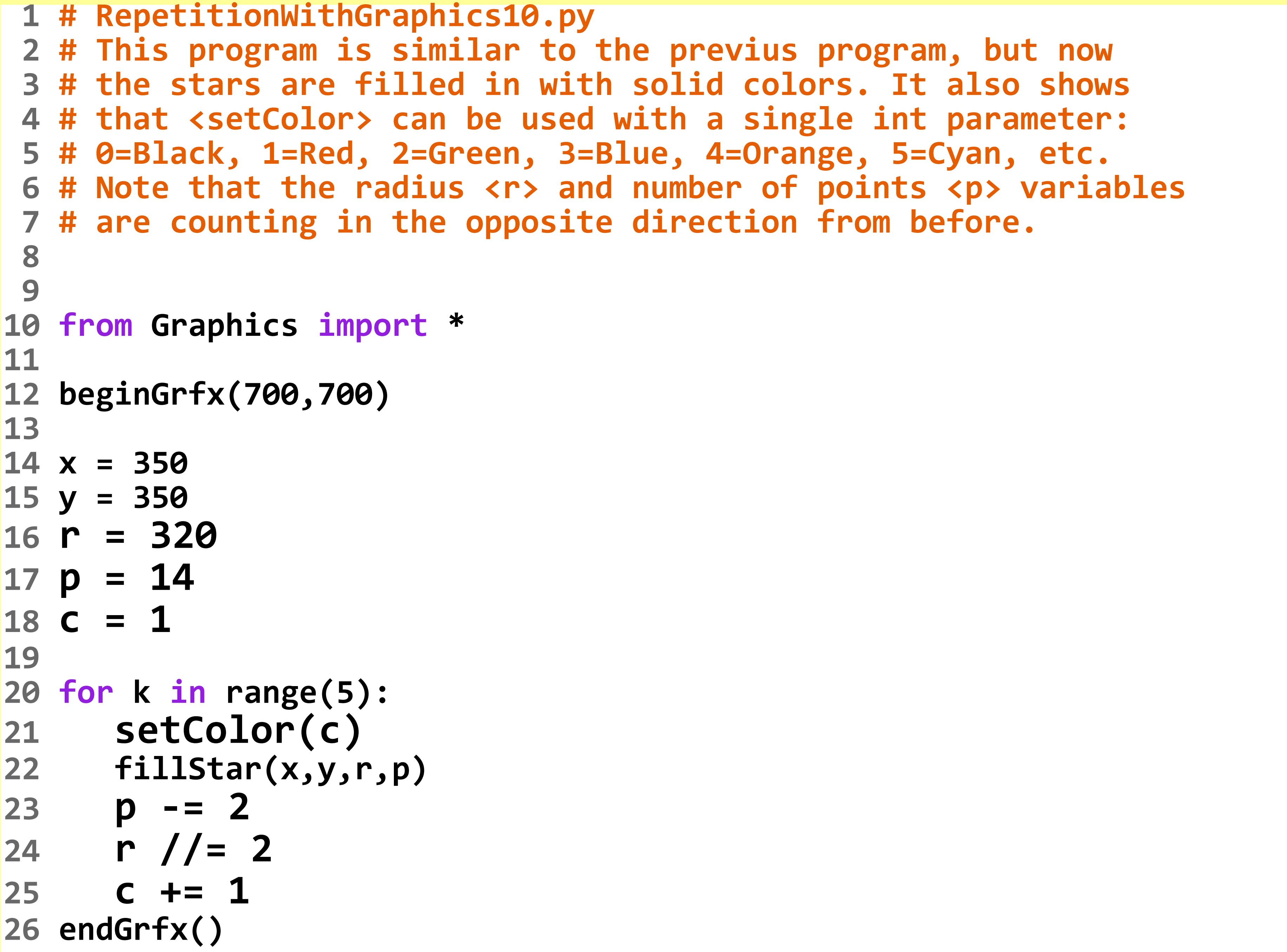
```
# RepetitionWithGraphics08.py
# This pro
# All of t
# radius.
# NOTE: Th
# of the o


from Graph

beginGrfx(

x = 650
y = 350
hr = 50
vr = 150

for k in range(30):
    drawOval(x,y,hr,vr)
    hr += 20

endGrfx()
```



```
# RepetitionWithGraphics09.py
# This program demonstrates several concentric stars.
# Each new star has 2 more points and double the radius
# of the previous star.


from Graphics import *

beginGrfx(700,700)

x = 350
y = 350
r = 20
p = 6

for k in range(5):
   drawStar(x,y,r,p)
   p += 2
   r *= 2

endGrfx()
```

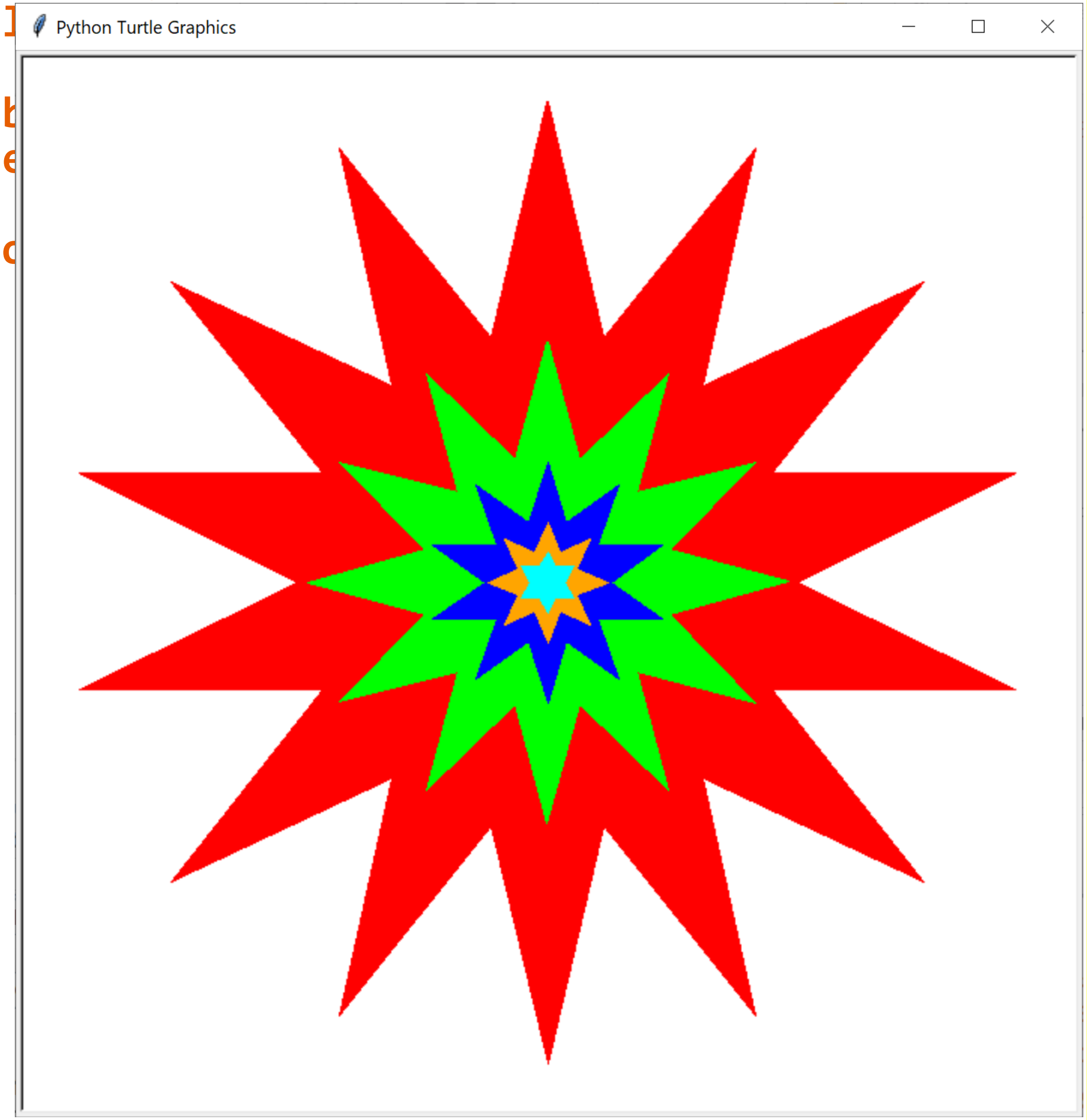
```
# RepetitionWithGraphics09.
# This program demonstrates
# Each new star has 2 more
# of the previous star.


from Graphics import *

beginGrfx(700,700)

x = 350
y = 350
r = 20
p = 6

for k in range(5):
    drawStar(x,y,r,p)
    p += 2
    r *= 2

endGrfx()
```





```python
# RepetitionWithGraphics10.py
# This program is similar to the previus program, but now
# the stars are filled in with solid colors. It also shows
# that <setColor> can be used with a single int parameter:
# 0=Black, 1=Red, 2=Green, 3=Blue, 4=Orange, 5=Cyan, etc.
# Note that the radius <r> and number of points <p> variables
# are counting in the opposite direction from before.


from Graphics import *

beginGrfx(700,700)

x = 350
y = 350
r = 320
p = 14
c = 1

for k in range(5):
   setColor(c)
   fillStar(x,y,r,p)
   p -= 2
   r //= 2
   c += 1
endGrfx()
```

```
# RepetitionWithGraphics10.py
# This program is simil
# the stars are filled
# that <setColor> can b
# 0=Black, 1=Red, 2=Gre
# Note that the radius
# are counting in the c


from Graphics import *

beginGrfx(700,700)

x = 350
y = 350
r = 320
p = 14
c = 1

for k in range(5):
   setColor(c)
   fillStar(x,y,r,p)
   p -= 2
   r //= 2
   c += 1
endGrfx()
```



Lab 8B

What you saw in the past 7 program examples relates directly to what you will be doing in Lab 8B.