The Kyoto College of Graduate Studies for Informatics

kcg.edu

コンピュータプログラミング概論

秋期第６回 eラーニング資料

安　平勲

h_an@kcg.ac.jp

標準ライブラリ　turtle

- **タートルグラフィックス**は，教育用プログラミング言語の「LOGO」を元に，画面上の亀をプログラミングで操作して描画するもの
- 1960 年代に開発され、**プログラム結果**を**視覚**ですぐに**確認（Graphical User Interface）**できることから、子供向けのプログラミング学習に用いられる
- Pythonのタートルグラフィックスは標準ライブラリ**turtle**モジュールで提供されている

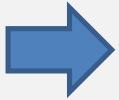
ファイル(.py) でturtleモジュールを試す

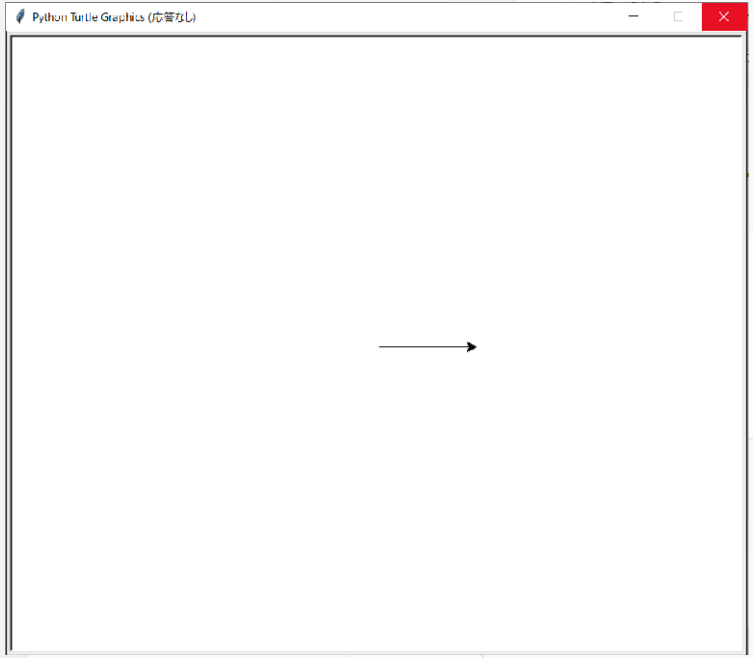
- Spyderでタートルグラフィックスに触れる

```
import turtle

turtle.forward(100)

turtle.done()
turtle.bye()
```





- 標準モジュールturtleをimport
- forward関数は直線を描く
- done関数とbye関数は最後に必ず書く

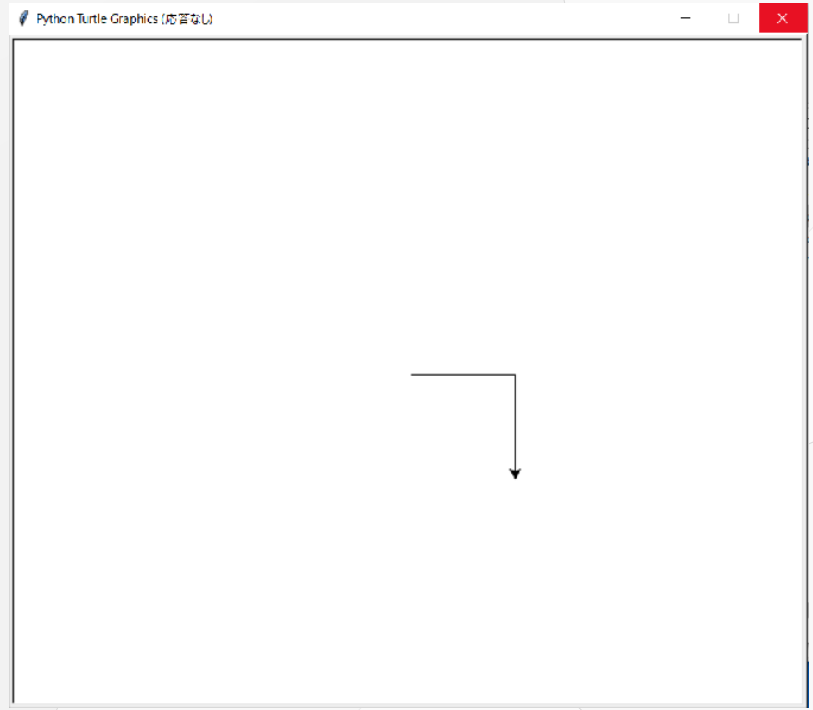
☞ファイル名を”turtle.py“に絶対にしない

turtleモジュールを試す（その２）

```
import turtle
turtle.forward(100)

turtle.right(90)
turtle.forward(100)

turtle.done()
```



- right関数で矢印を右に90度回転。そして100step進む

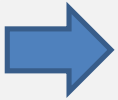
turtleモジュールを試す（その３）

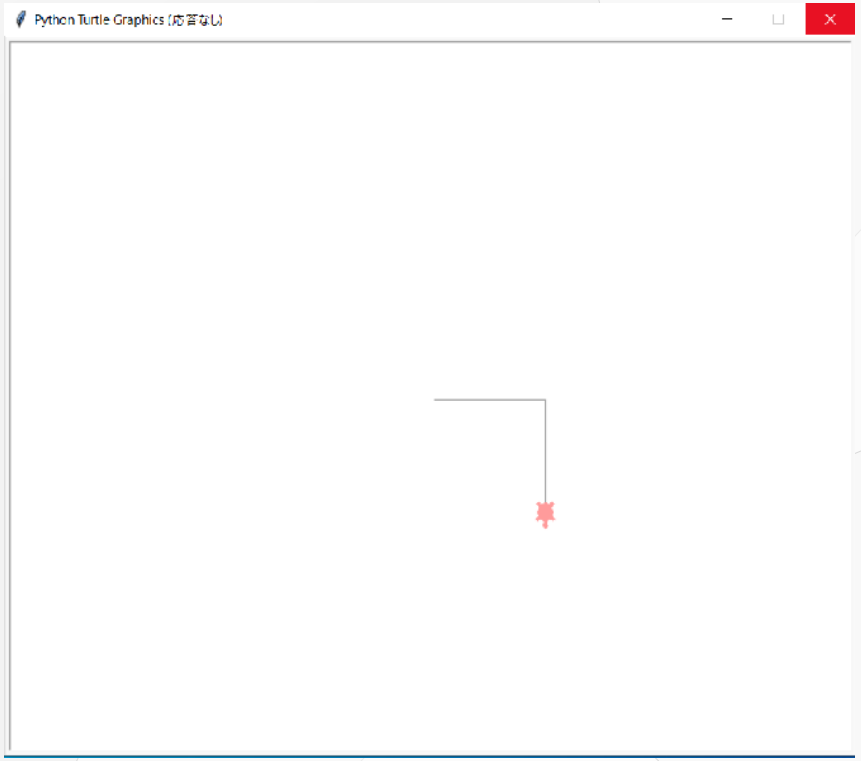
```
import turtle
turtle.forward(100)

turtle.right(90)
turtle.forward(100)

turtle.shape('turtle')
turtle.color('red')

turtle.done()
```





- 矢印の形を亀に，色を赤にする

turtleモジュールを試す（その４）

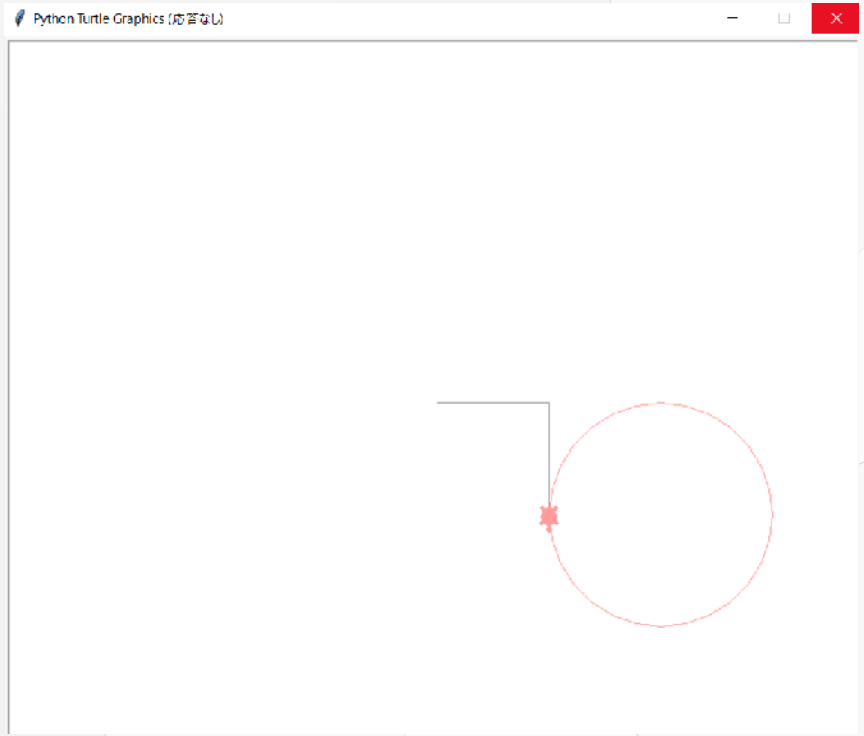
```
turtle.right(90)
turtle.forward(100)

turtle.shape('turtle')
turtle.color('red')

turtle.hideturtle()
turtle.showturtle()

turtle.circle(100)

turtle.done()
```



- hideturtlで亀を隠し、showturtle で再表示
- circleで円を描く（線の色も赤に）

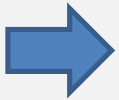
turtleモジュールを試す（その５）

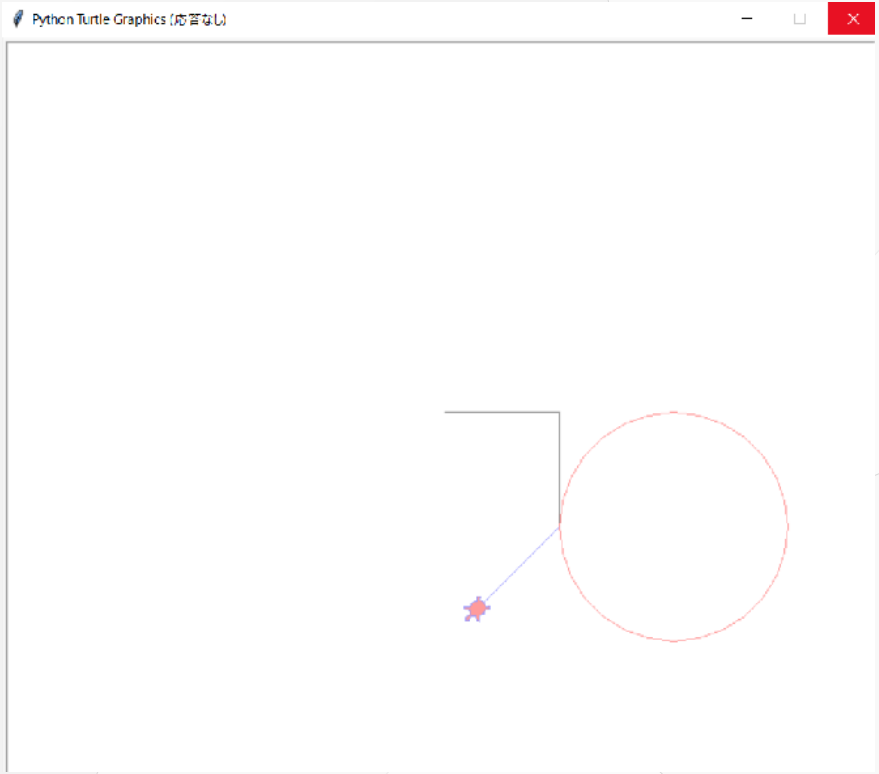
```
turtle.showturtle()

turtle.circle(100)

turtle.pencolor('blue')
turtle.right(45)
turtle.forward(100)

turtle.done()
```





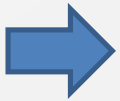
- 線の色を青に
- 右に45度回転して
- 100step進む

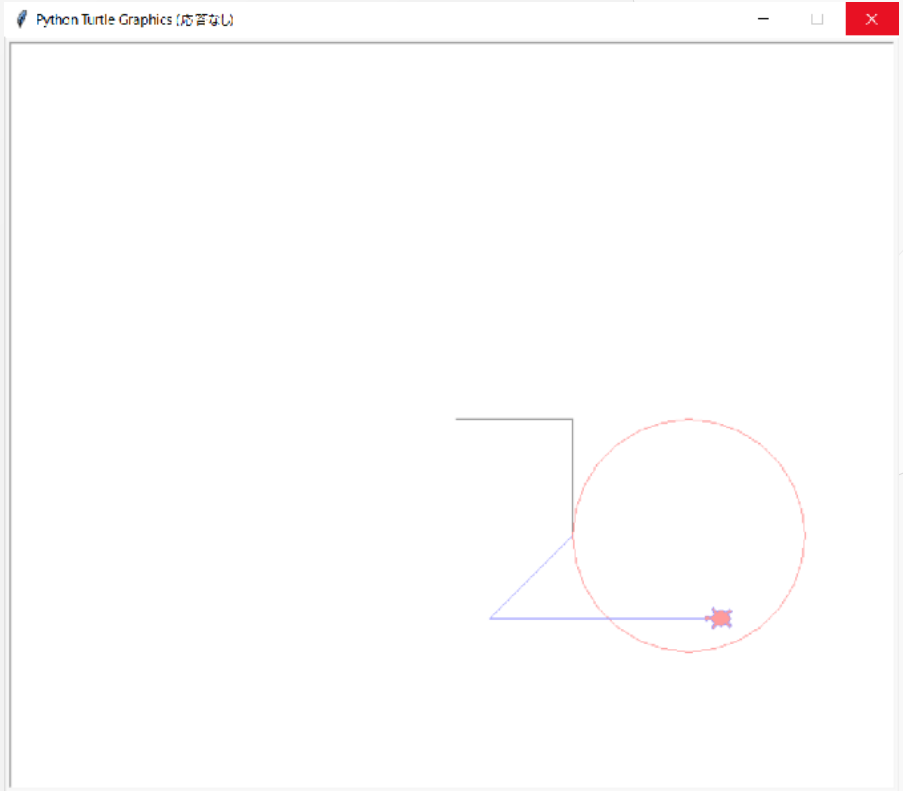
turtleモジュールを試す（その６）

```
turtle.pencolor('blue')
turtle.right(45)
turtle.forward(100)

turtle.right(45)
turtle.backward(200)

turtle.done()
```





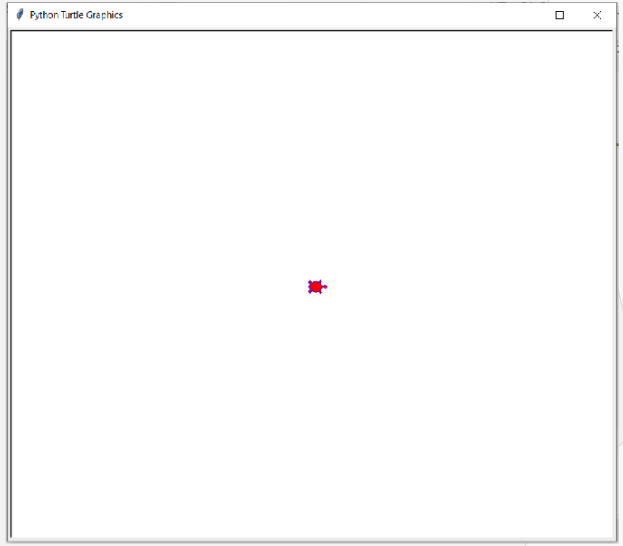
- さらに右に45度回転して,
- 200step後ろ向きに進む

turtleモジュールを試す（その７）

```
turtle.right(45)
turtle.backward(200)

turtle.home()
turtle.clear()

turtle.done()
```



- homeで一番最初の位置（ホームポジション）に戻る
- clearで描画がすべて消去される

クラスとオブジェクトとメソッド

- **turtle**モジュールは**Turtleクラス**と**メソッド**を提供する

1. **t1 = turtle.Turtle()** ➢ **turtle**モジュールが提供する**Turtleクラス**から**オブジェクトを作り**，それを**t1**と命名

使用方法
オブジェクト名 = モジュール名. クラス名（）

2. **t1.shape('turtle')** ➢ **オブジェクトt1**の矢印を亀形にする**メソッド**

使用方法
オブジェクト名. メソッド名（引数, 引数, , ）

☞ t2 = turtle.Turtle()で別の亀（**オブジェクトt2**）が作れる

オブジェクトのturtleを試す

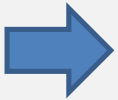
- Turtle**クラス**から**オブジェクト**を作る
- 回転角度が合計360°になる反復処理で正N角形が作れる

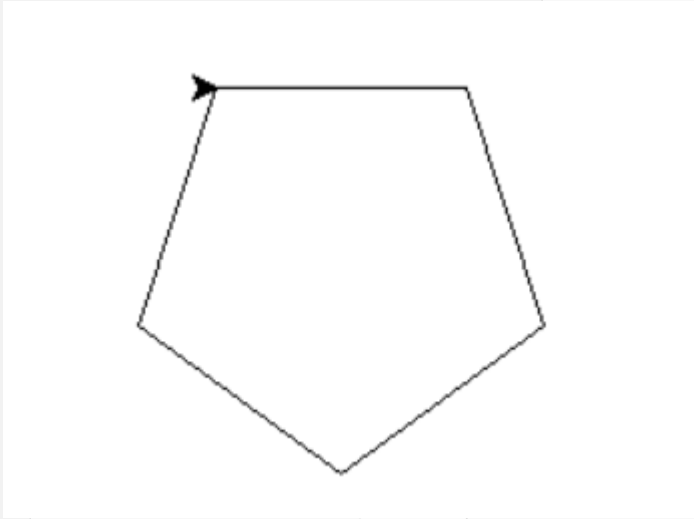
```
import turtle

t1=turtle.Turtle()

for i in range(5):
    t1.forward(100)
    t1.right(360 / 5 )

turtle.done()
```





☞ forwardもrightもturtleの標準メソッド。機能は標準関数と同じ（呼び出し方が違う）

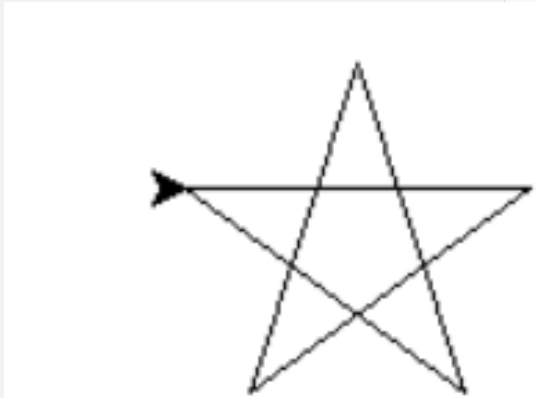
オブジェクトのturtleを試す

- さらに回転角度を変えると，星形を描ける

```
import turtle
t1=turtle.Turtle()

for i in range(5):
    t1.forward(100)
    t1.right(360 / 5 * 2)

turtle.done()
```



オブジェクトのturtleを試す

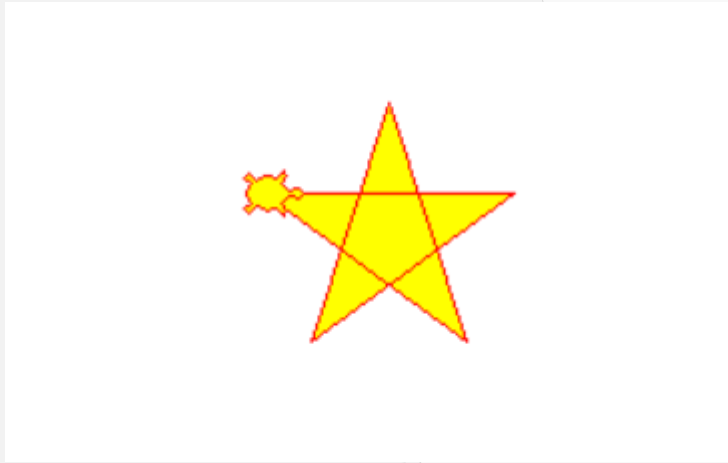
- 矢印を亀に、色もつけるメソッドも関数と同じ

```
import turtle
t1=turtle.Turtle()

t1.color('red', 'yellow')
t1.shape('turtle')

t1.begin_fill()
for i in range(5):
    t1.forward(100)
    t1.right(360 / 5 * 2)
t1.end_fill()

turtle.done()
```



☞ begin_fillとend_fillメソッドで図の塗りつぶしができる

オブジェクトのturtleを試す

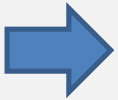
- 長さを少しずつ伸ばした星形を順に４つ描くと…

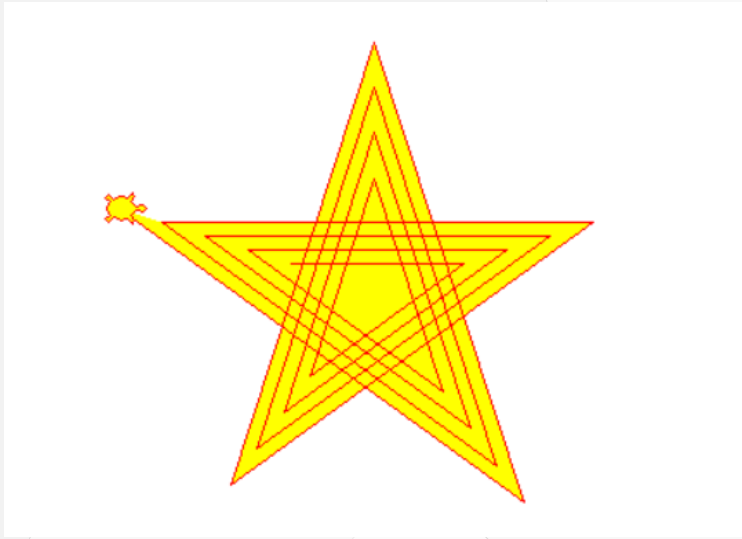
```
import turtle
t1=turtle.Turtle()

t1.color('red', 'yellow')
t1.shape('turtle')

t1.begin_fill()
for i in range(5 * 4):
    t1.forward(100 + i*10)
    t1.right(360 / 5 * 2)
t1.end_fill()

turtle.done()
```





2匹の亀を描画する

```
import turtle

t1 = turtle.Turtle()

t1.color('red')
t1.shape('turtle')
for i in range(5):
    t1.forward(100)
    t1.right(360 / 5 * 2)

t2 = turtle.Turtle()

t2.color('blue')
t2.shape('turtle')

t2.right(180)
for i in range(5):
    t2.forward(100)
    t2.left(360 / 5 * 2)

turtle.done()
```

- t2 = turtle.Turtle()で別の亀（**オブジェクト t2**）が作れる