Python 程序设计

第 2 章 基本图形绘制

Python基本语法元素

- 缩进、注释、命名、变量、保留字
- 数据类型、字符串、 整数、浮点数、列表
- 赋值语句、分支语句、函数
- input()、 print()、 eval()、 print()格式化

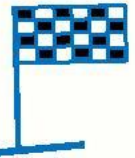
CC BY-NC-SA 4.0 嵩天

and	elif	import	raise	global
as	else	in	return	nonlocal
assert	except	is	try	True
break	finally	lambda	while	False
class	for	not	with	None
continue	from	or	yield	
def	if	pass	del	

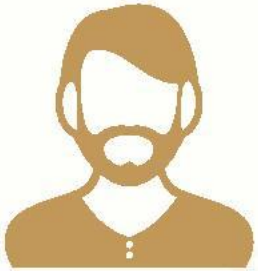
保留字

```
#TempConvert.py
TempStr = input("请输入带有符号的温度值: ")
if TempStr[-1] in ['F', 'f']:
    C = (eval(TempStr[0:-1]) - 32)/1.8
    print("转换后的温度是{:.2f}C".format(C))
elif TempStr[-1] in ['C', 'c']:
    F = 1.8*eval(TempStr[0:-1]) + 32
    print("转换后的温度是{:.2f}F".format(F))
else:
    print("输入格式错误")
```

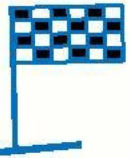
温度转换



第2章 Python基本图形绘制



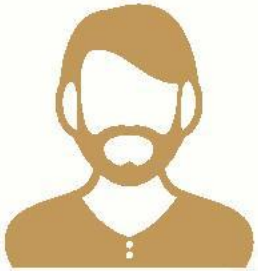
- 2.1 深入理解Python语言
- 2.2 实例2: Python蟒蛇绘制
- 2.3 模块1: turtle库的使用
- 2.4 turtle程序语法元素分析



python
语言程序设计

CC BY-NC-SA 嵩天

第2章 Python基本图形绘制



方法论

- Python语言及海龟绘图体系

实践能力

- 初步学会使用Python绘制简单图形

python
语言程序设计

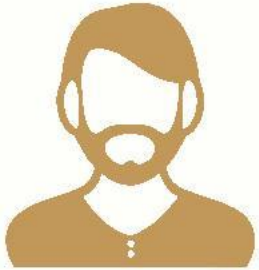
CC BY-NC-SA 4.0 嵩天

Python语言程序设计

深入理解Python语言

CC BY-NC-SA 嵩天

深入理解Python语言



- 计算机技术的演进
- 编程语言的多样性
- Python语言的特点
- "超级语言"的诞生

python
语言程序设计

CC BY-NC-SA 4.0 嵩天

计算机技术的演进

CC BY-NC-SA 4.0 嵩天

计算机技术的演进过程


2008-2016
复杂信息系统时代 (8年)
1981-2008
网络和视窗时代 (27年)
2016-
人工智能时代
1946-1981
计算机系统结构时代 (35年)

计算机技术的演进过程

2017-	人工智能时代	人类的问题
	新计算时代	
2008-2016	复杂信息系统时代	数据问题
1981-2008	网络和视窗时代	交互问题
1946-1981	计算机系统结构时代	计算能力问题

CC BY-NC-SA 嵩天

编程语言多样性

CC BY-NC-SA 4.0 嵩天

编程语言有哪些?

Basic, C, C++, C#, CSS, Fortran, Go, HTML, Java,

JavaScript, Lisp, Lua, Matlab, Object C, Pascal, Perl, PHP,

PostScript, Python, Ruby, Scala, SQL, Swift, VBA,

VB.NET, Verilog, VHDL, Visual Basic

CC BY-NC-SA 4.0 高天

不同编程语言和适用对象

编程语言	学习内容	语言本质	解决问题	适用对象
C	指针、内存、数据类型	理解计算机系统结构	性能	计算机类专业
Java	对象、跨平台、运行时	理解主客体关系	跨平台	软件类专业
C++	对象、多态、继承	理解主客体关系	大规模程序	计算机类专业
VB	对象、按钮、文本框	理解交互逻辑	桌面应用	不确定
Python	编程逻辑、第三方库	理解问题求解	各类问题	所有专业

各编程语言所处历史时期和使命不同，Python是计算时代演进的选择！

2018年以后的计算环境...

计算机性能不再是解决一般问题的瓶颈

移动互联网广泛普及

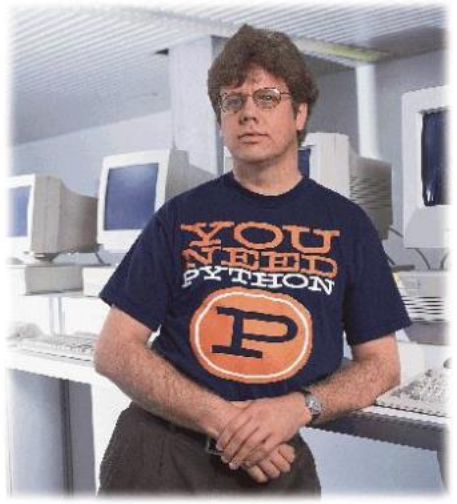
大数据、云计算、物联网、信息安全、人工智能等需求爆发

解决日益增长的计算需求，用什么语言?

CC BY-NC-SA 4.0 青天

Python语言的特点

CC BY-NC-SA 4.0 嵩天



- Python语言是通用语言
- Python语言是脚本语言
- Python语言是开源语言
- Python语言是跨平台语言
- Python语言是多模型语言

Guido van Rossum

Python语言创立者

2002年，Python 2.x

2008年，Python 3.x

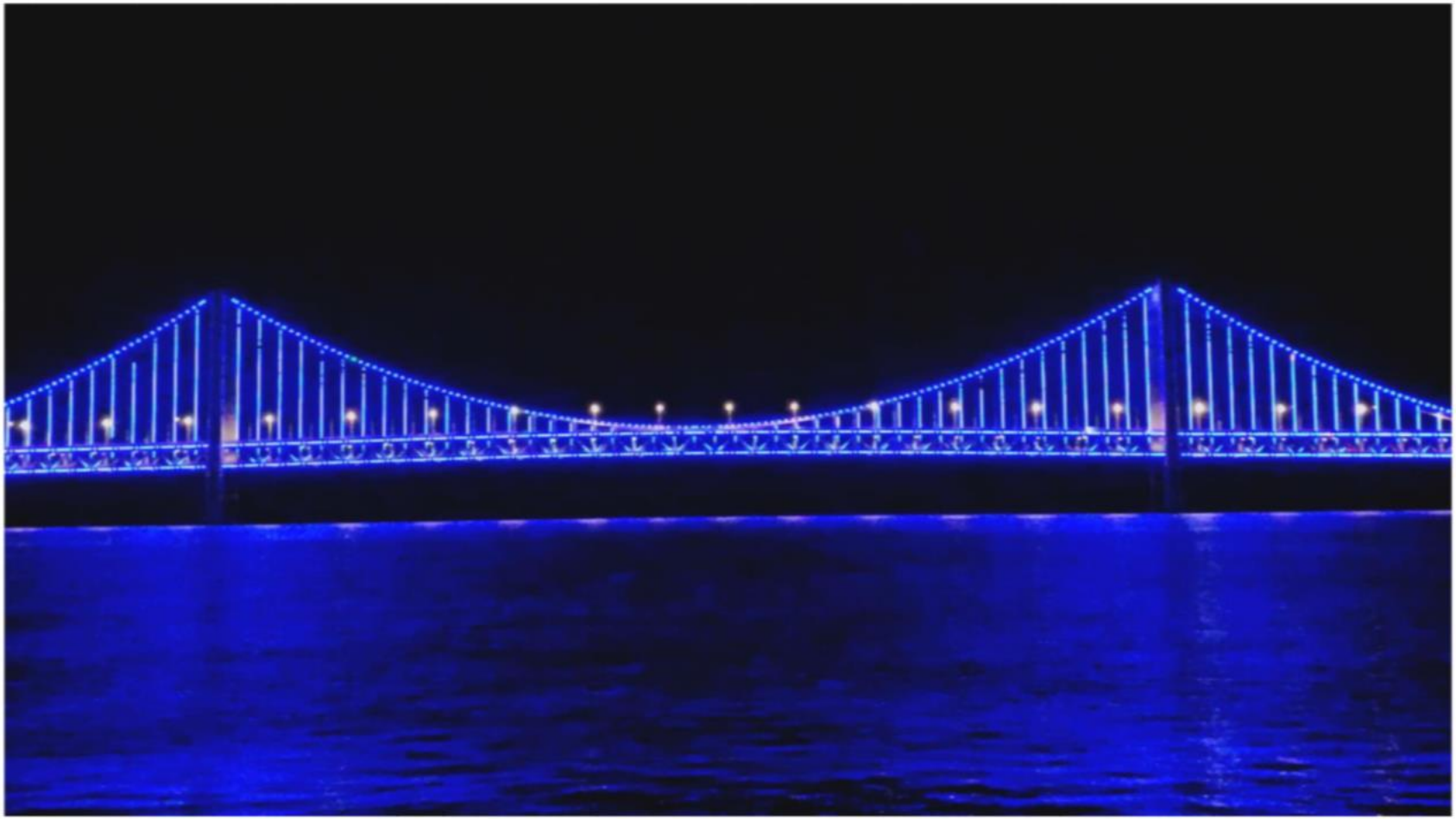
CC BY NC-SA 4

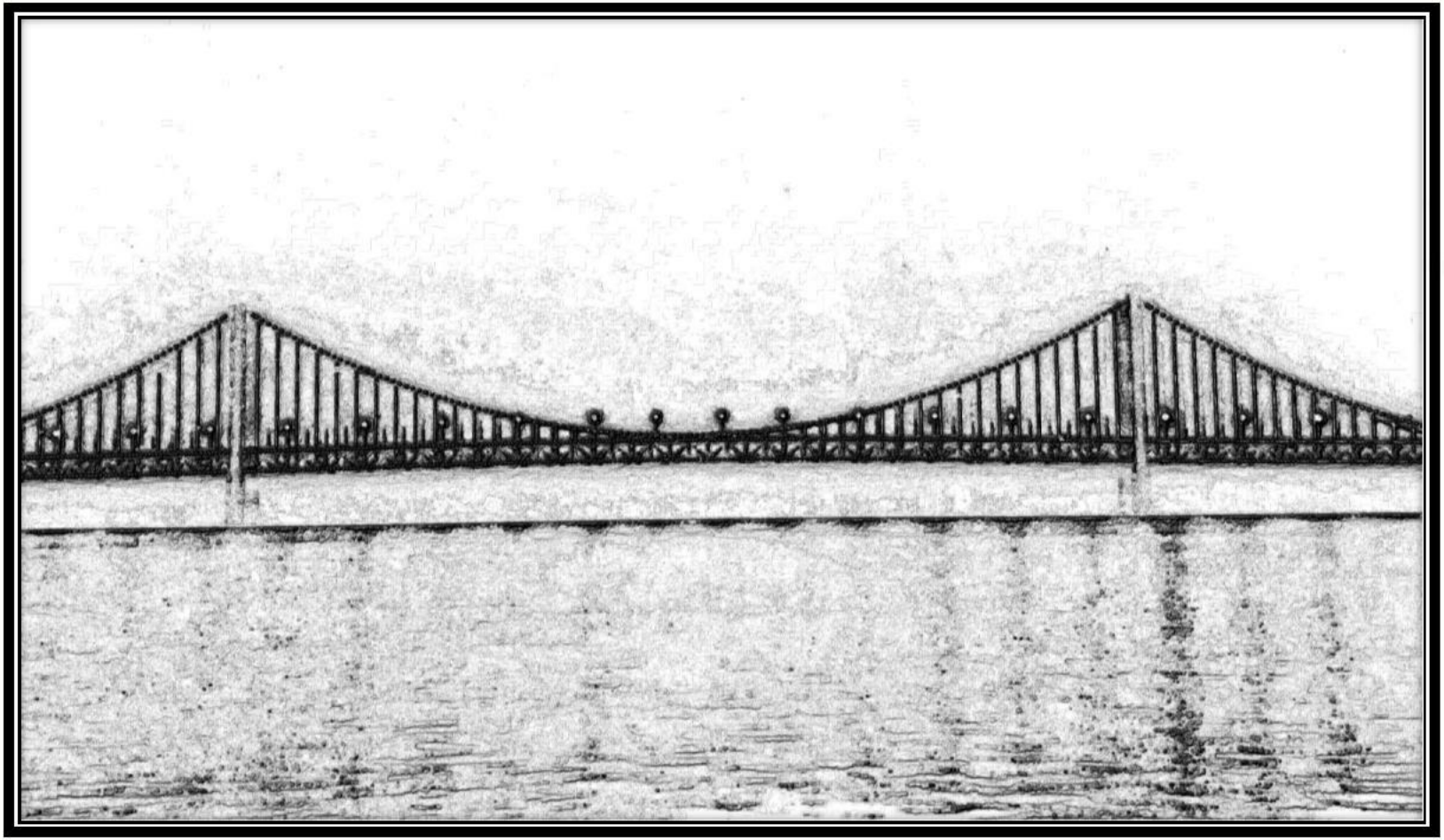
Python特点与优势

- C代码量的10%
- 强制可读性
- 较少的底层语法元素
- 多种编程方式
- 支持中文字符

- >13万第三方库
- 快速增长的计算生态
- 避免重复造轮子
- 开放共享
- 跨操作系统平台

CC BY-NC-SA 4.0 嘉木





Python 21行代码

如何看待Python语言？

人生苦短，我学Python

- **C/C++：Python归Python，C归C**
- **Java：针对特定开发和岗位需求**
- **HTML/CSS/JS：不可替代的前端技术，全栈能力**
- **其他语言：R/Go/Matlab等，特定领域**

CC BY-NC-SA 4.0 嵩天

如何看待Python语言？

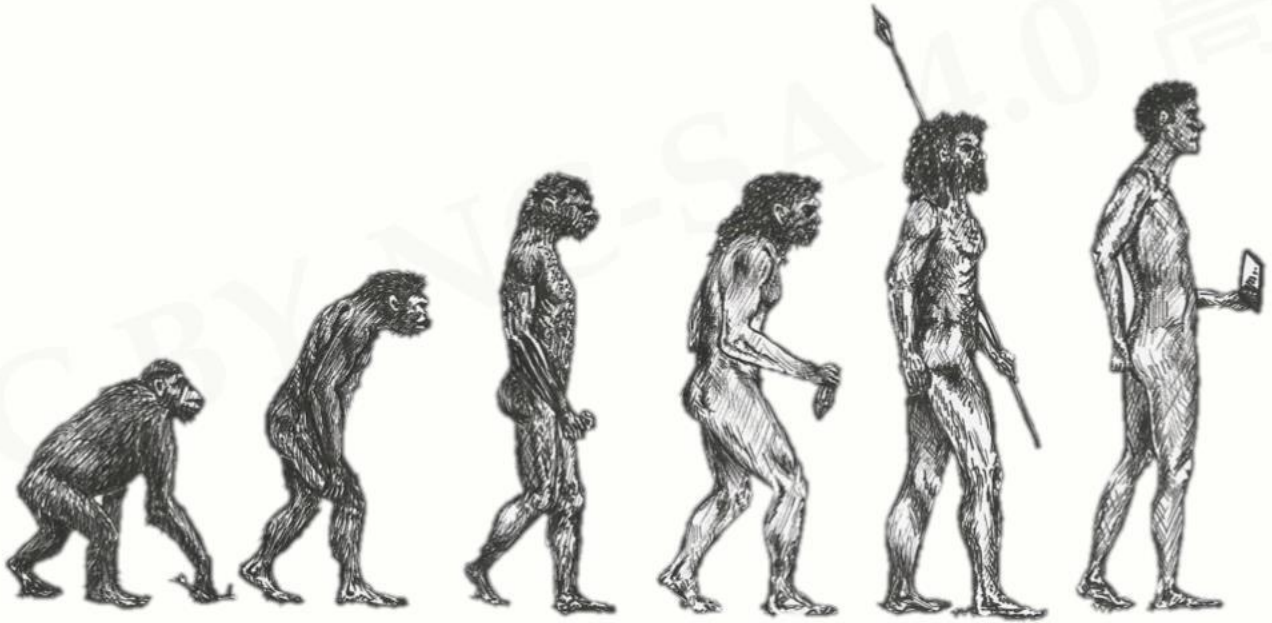
Python是最高产的程序设计语言及……

- 掌握**抽象并求解**计算问题综合能力的语言
- 了解产业界解决**复杂计算问题**方法的语言
- 享受利用编程**将创新变为实现**乐趣的语言

CC BY-NC-SA 4.0 嵩天

如何看待Python语言？

工具决定思维：关注工具变革的力量！



CC BY-NC-SA 4.0 高天

"超级语言"的诞生

CC BY-NC-SA 4.0 嵩天

编程语言的种类

机器语言

- 一种二进制语言，直接使用二进制代码表达指令
- 计算机硬件(CPU)可以直接执行，与具体CPU型号有关
- 完成 2+3 功能的机器语言

```
11010010 00111011
```

编程语言的种类

汇编语言

- 一种将二进制代码直接对应助记符的编程语言
- 汇编语言与CPU型号有关，程序不通用，需要汇编器转换
- 完成 2+3 功能的汇编语言

```
add 2,3,result
```

CC BY-NC-SA 4.0 嵩天

编程语言的种类

高级语言

- 更接近自然语言，同时更容易描述计算问题
- 高级语言代码与具体CPU型号无关，编译后运行
- 完成 2+3 功能的高级语言

```
result = 2 + 3
```

CC BY NC-SA 4.0 高天

编程语言种类的发展

超级语言 - 粘性整合已有程序，具备庞大计算生态

高级语言 - 接近自然语言，编译器，与CPU型号无关

汇编语言 - 有助记符，汇编器，与CPU型号有关

机器语言 - 代码直接执行，与CPU型号有关

CC BY-NC-SA 4.0 声明

编程语言的种类

超级语言

- 具有庞大计算生态，可以很容易利用已有代码功能
- 编程思维不再是刀耕火种，而是集成开发
- 完成 2+3 功能的超级语言

```
result = sum(2,3)
```

CC BY-NC-SA 4.0 高天

Python: 唯一的"超级语言"！

CC BY-NC-SA 4.0 嵩天

Python前进的步伐不可阻挡

CC BY-NC-SA 4.0 嵩天

深入理解Python语言

- 计算机系统结构时代到人工智能时代的演进路线
- 五种编程语言和历史使命
- Python语言的通用性、简洁性和生态性
- Python是以计算生态为标志的"超级语言"

CC BY-NC-SA 4.0 嵩天

Python语言程序设计

实例2: Python蟒蛇绘制

"Python蟒蛇绘制"问题分析

CC BY-NC-SA 4.0 嵩天

Python蟒蛇绘制

设计蟒蛇的基本形状

CC BY-NC-SA 4.0 嵩天

Python蟒蛇绘制

用程序绘制一条蟒蛇

- **问题1: 计算机绘图是什么原理?**

  一段程序为何能够产生窗体? 为何能在窗体上绘制图形?

- **问题2: Python蟒蛇绘制从哪里开始呢?**

  如何绘制一条线? 如何绘制一个弧形? 如何绘制一个蟒蛇?

CC BY-NC-SA 4.0 嵩天

Python蟒蛇绘制

用程序绘制一条蟒蛇

实例1: 温度转换 → **Python蟒蛇绘制**

能否借鉴?

CC BY-NC-SA 4.0 嵩天

"Python蟒蛇绘制"实例编写

CC BY-NC-SA 4.0 嵩天

```
#PythonDraw.py
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

使用IDLE的文件方式

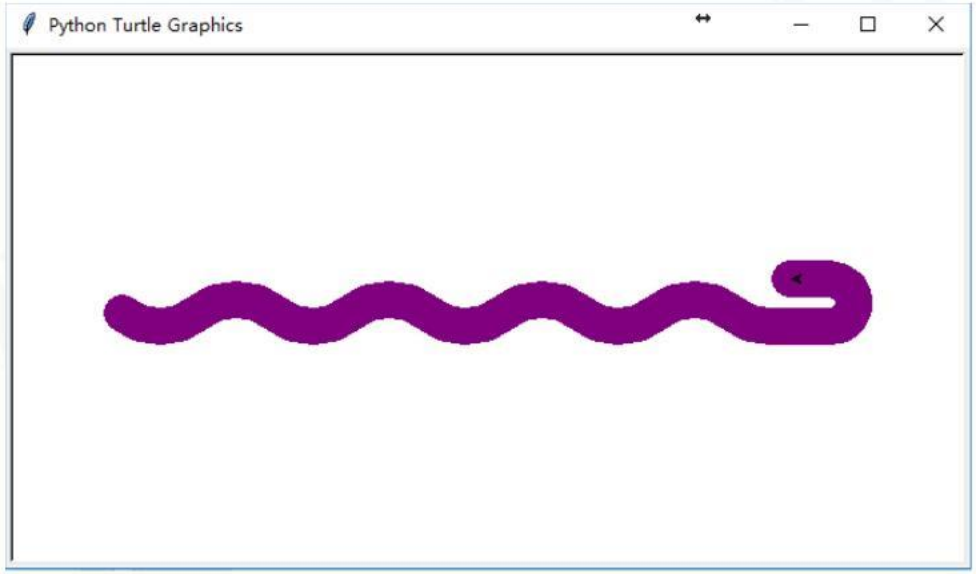
编写代码

并保存为

PythonDraw.py 文件

运行效果

IDLE打开文件，按F5运行



```
#PythonDraw.py
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

程序关键

import 保留字

引入了一个绘图库

名字叫：turtle

没错，就是 海龟

"Python蟒蛇绘制"举一反三

CC BY-NC-SA 4.0 嵩天

```
#PythonDraw.py
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

CC BY 4.0 嵩天

python

举一反三

Python语法元素理解

- Python蟒蛇绘制共17行代码，但很多行类似
- 清楚理解这17行代码能够掌握Python基本绘图方法
- 参考框架结构、逐行分析、逐词理解

CC BY-NC-SA 4.0 嵩天

举一反三

程序参数的改变

- Python蟒蛇的颜色：黑色、白色、七彩色…
- Python蟒蛇的长度：1节、3节、10节…
- Python蟒蛇的方向：向左走、斜着走…

CC BY-NC-SA 4.0 嵩天

举一反三

计算问题的扩展

- Python蟒蛇绘制问题是各类图像绘制问题的代表
- 圆形绘制、五角星绘制、国旗绘制、机器猫绘制...
- 掌握绘制一条线的方法，就可以绘制整个世界

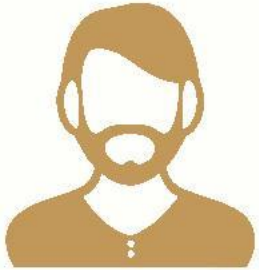
CC BY NC SA 4.0 嵩天

Python语言程序设计

模块1: turtle库的使用

CC BY-NC-SA 嵩天

模块1: turtle库的使用



- turtle库基本介绍
- turtle绘图窗体布局
- turtle空间坐标体系
- turtle角度坐标体系
- RGB色彩体系

CC BY-NC-SA 4.0 嵩天

turtle库基本介绍

CC BY-NC-SA 4.0 嵩天

turtle库概述

turtle(海龟)库是turtle绘图体系的Python实现

- **turtle绘图体系：1969年诞生，主要用于程序设计入门**
- **Python语言的标准库之一**
- **入门级的图形绘制函数库**

CC BY-NC-SA 4.0 嵩天

标准库

Python计算生态 = 标准库 + 第三方库

- **标准库：随解释器直接安装到操作系统中的功能模块**
- **第三方库：需要经过安装才能使用的功能模块**
- **库Library、包Package、模块Module，统称模块**

CC BY-NC-SA 4.0 嵩天

turtle的原理

turtle(海龟)是一种真实的存在

- **有一只海龟，其实在窗体正中心，在画布上游走**
- **走过的轨迹形成了绘制的图形**
- **海龟由程序控制，可以变换颜色、改变宽度等**

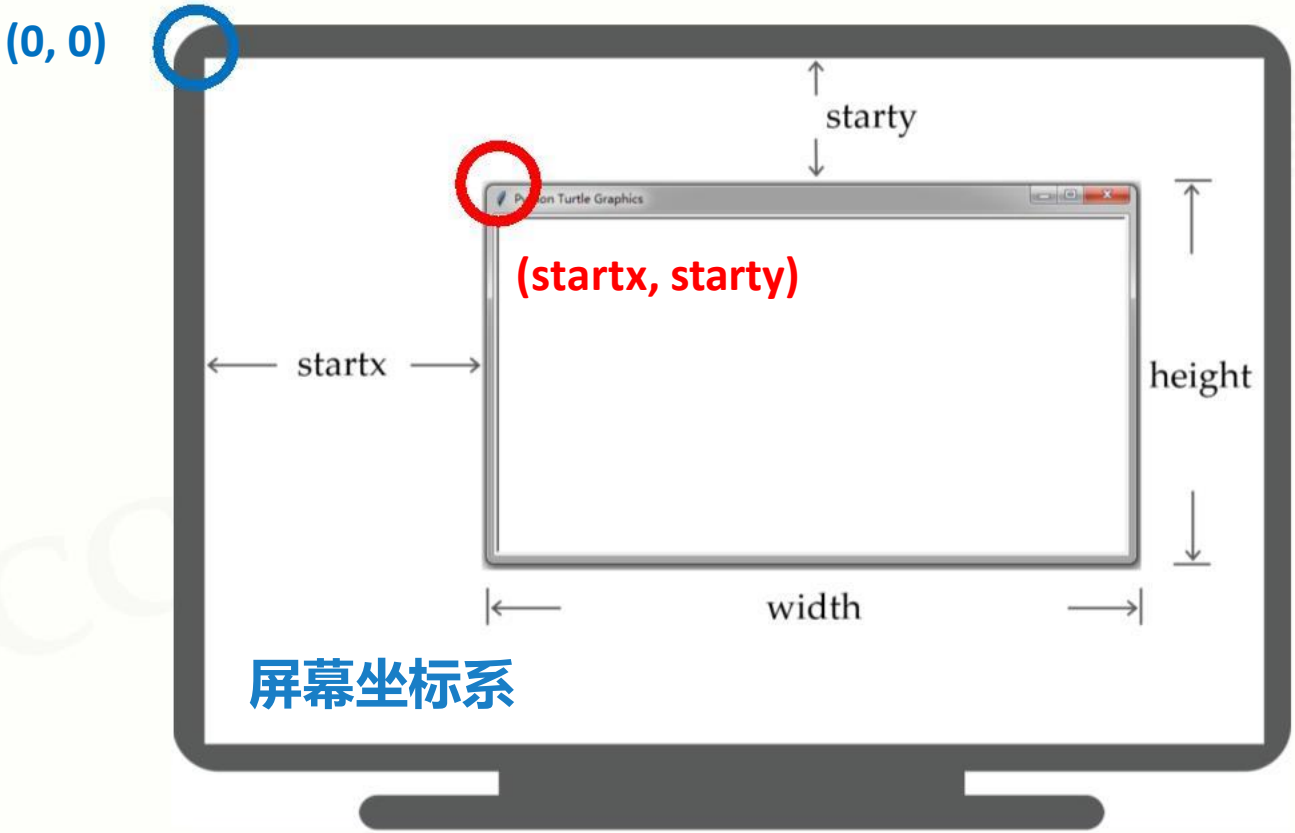
CC BY-NC-SA 4.0 嵩天

turtle绘图窗体布局

CC BY-NC-SA 4.0 嵩天

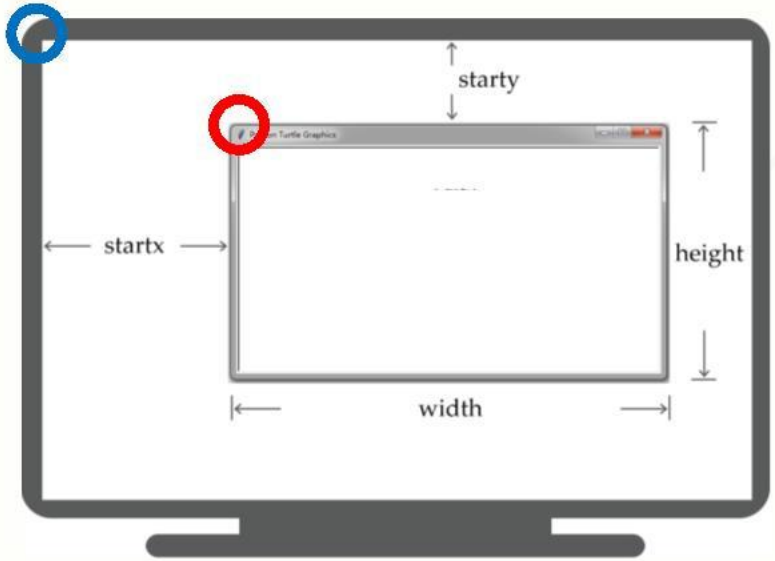
turtle的绘图窗体

turtle的绘图窗体



turtle的绘图窗体

```
turtle.setup(width, height, startx, starty)
```



- setup()设置窗体大小及位置
- 4个参数中后两个可选
- setup()不是必须的

turtle的绘图窗体

```
turtle.setup(800,800,0,0)
```

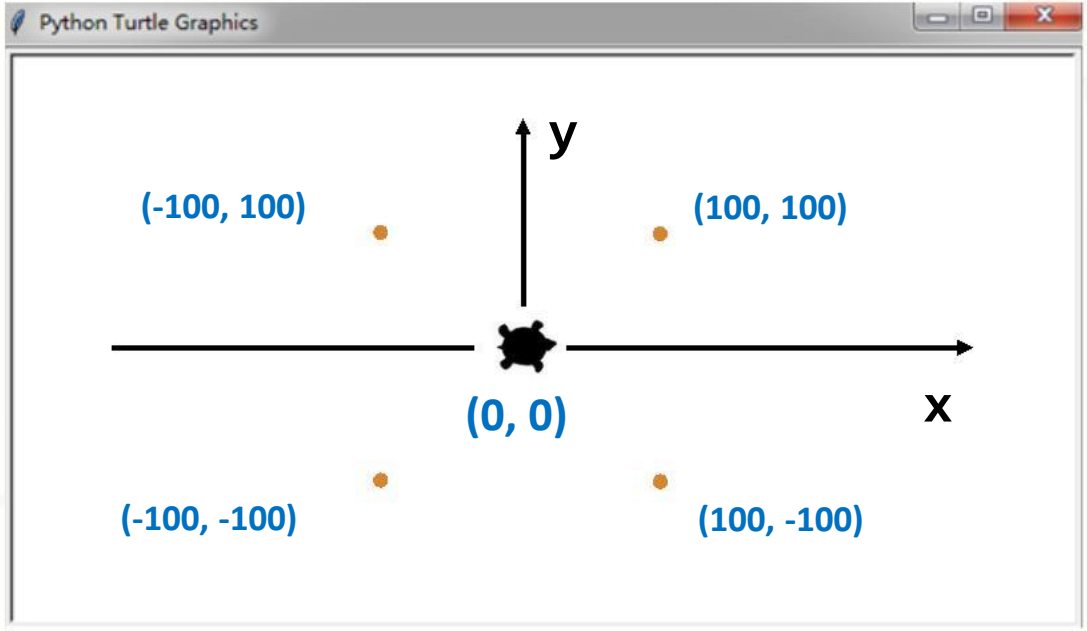
```
turtle.setup(800,800)
```

turtle空间坐标体系

CC BY-NC-SA 4.0 嵩天

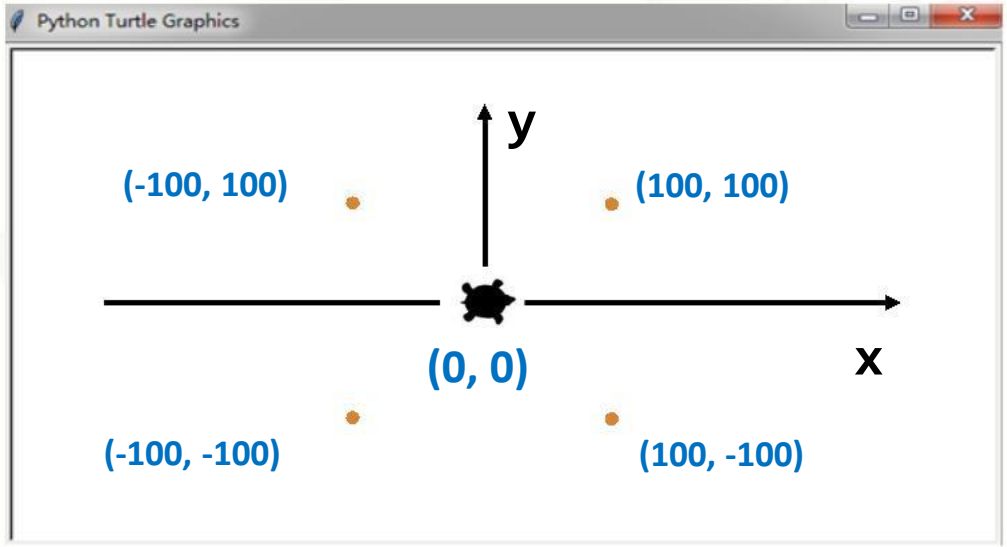
turtle空间坐标体系

绝对坐标



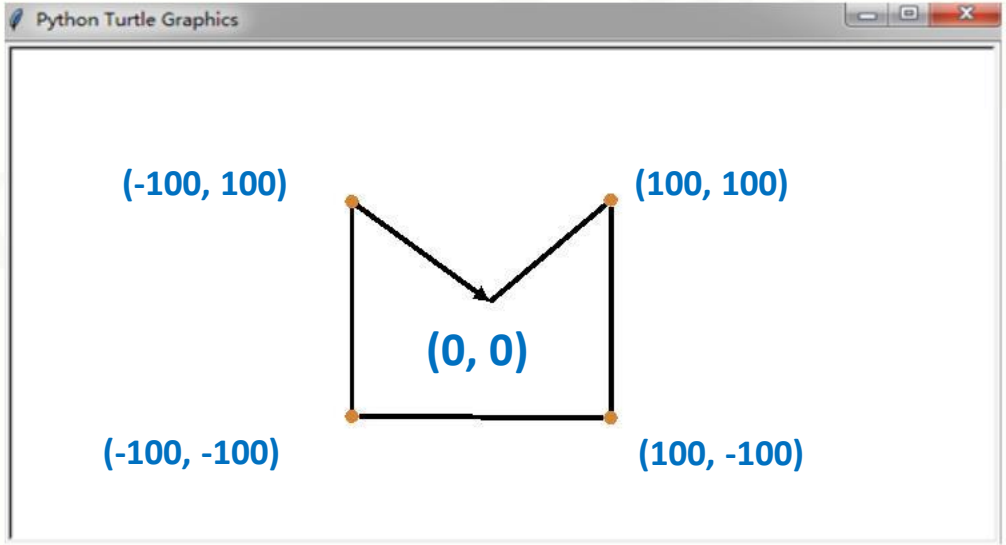
turtle空间坐标体系

```
turtle.goto(x, y)
```



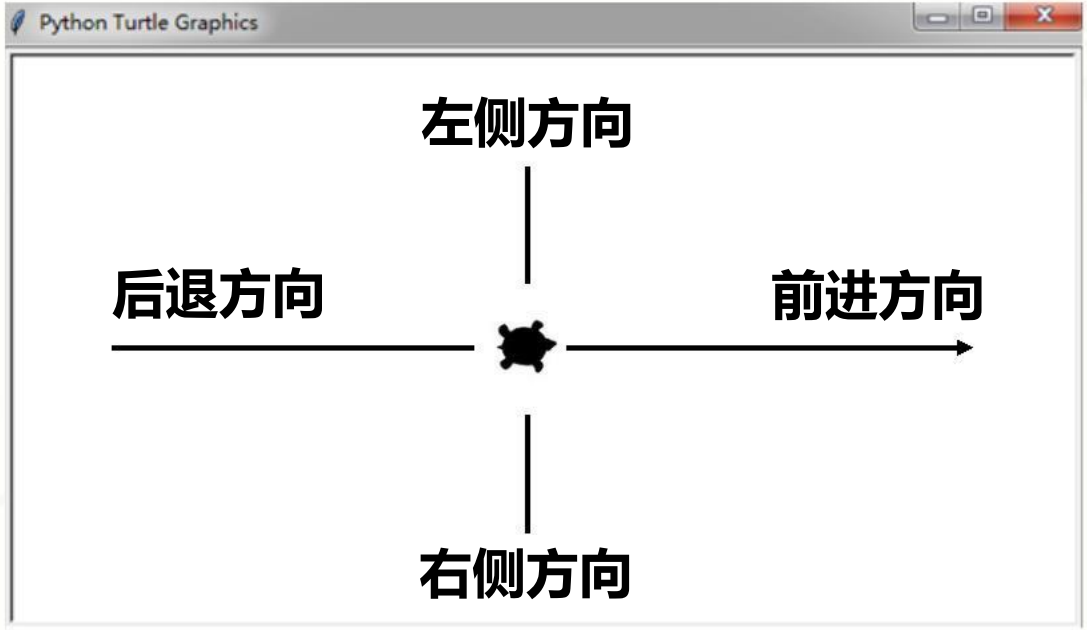
turtle空间坐标体系

```
import turtle
turtle.goto( 100, 100)
turtle.goto( 100,-100)
turtle.goto(-100,-100)
turtle.goto(-100, 100)
turtle.goto(0,0)
```

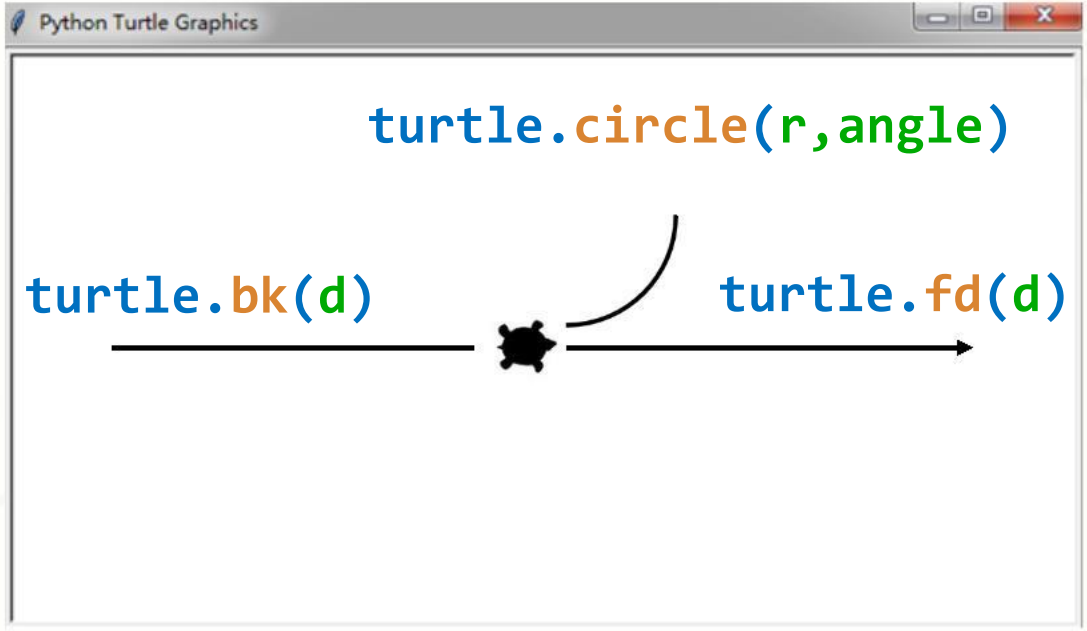


turtle空间坐标体系

海龟坐标



turtle空间坐标体系

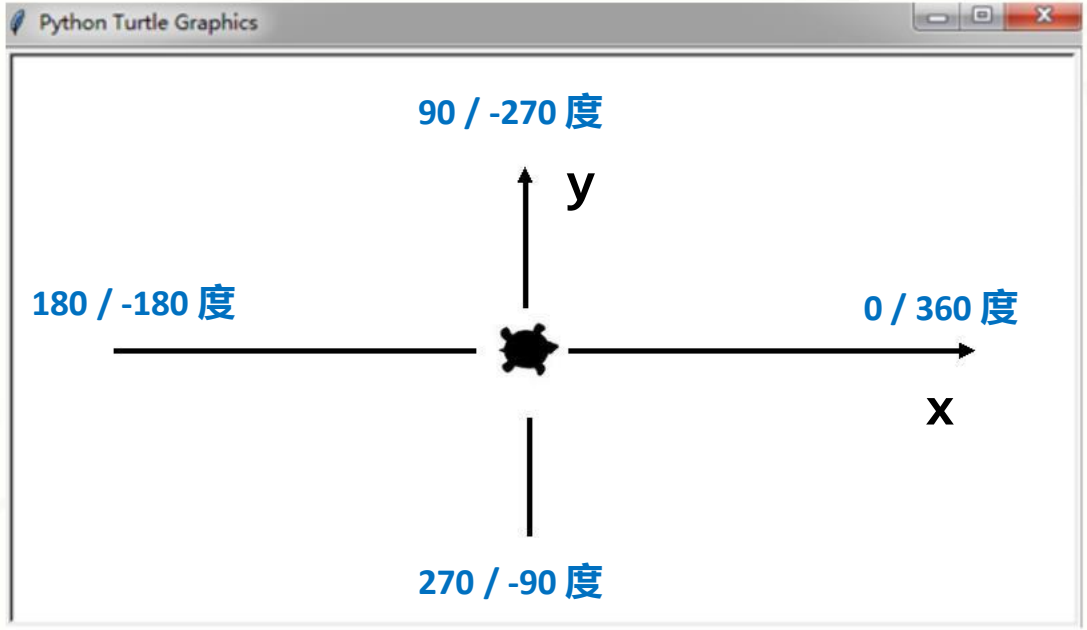


turtle角度坐标体系

CC BY-NC-SA 4.0 嵩天

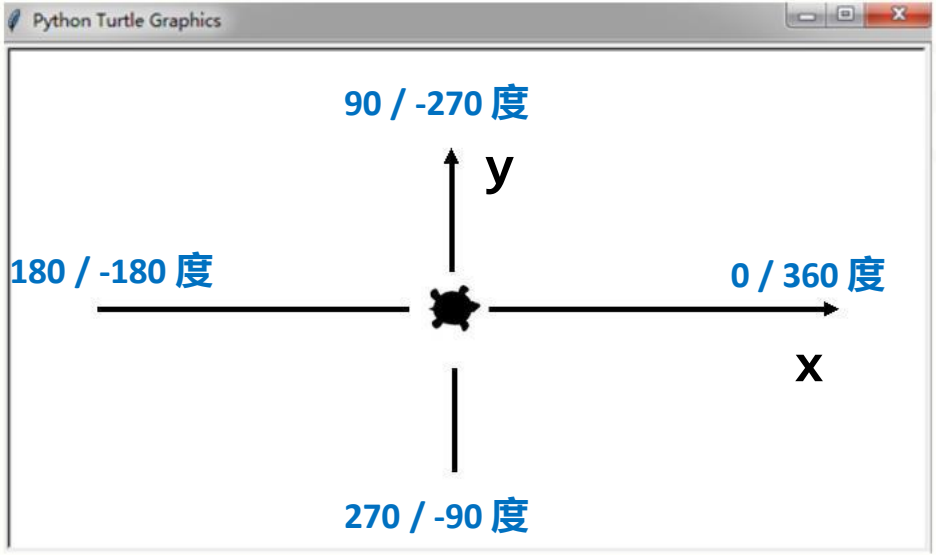
turtle角度坐标体系

绝对角度



turtle角度坐标体系

```
turtle.seth(angle)
```



- seth()改变海龟行进方向
- angle为绝对度数
- seth()只改变方向但不行进

CC BY-SA 4.0 嵩天

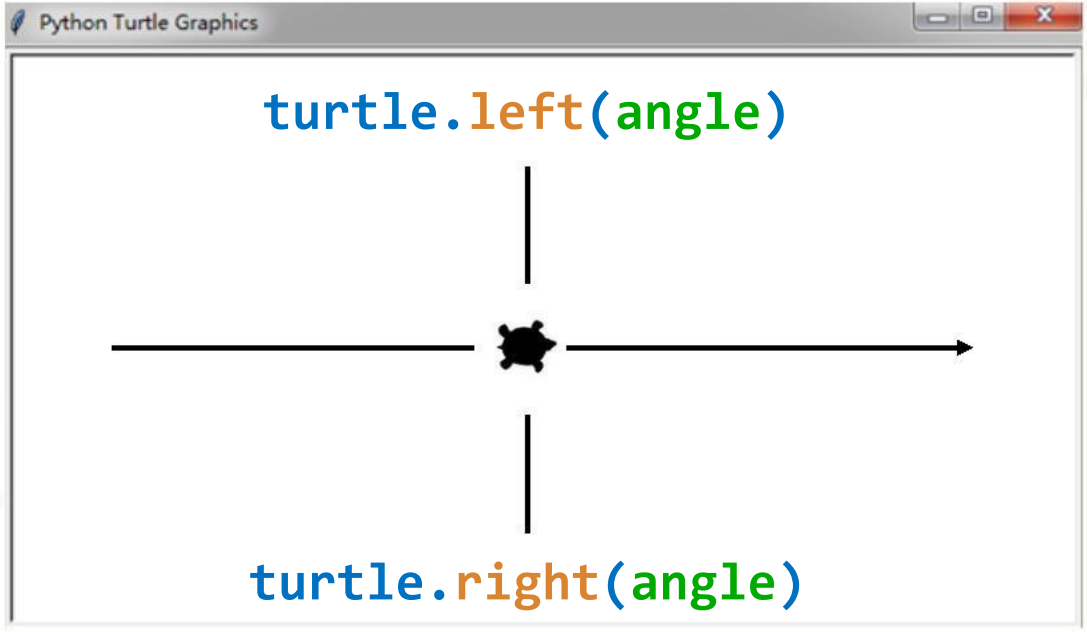
turtle角度坐标体系

```
turtle.seth(45)
```


```
turtle.seth(-135)
```

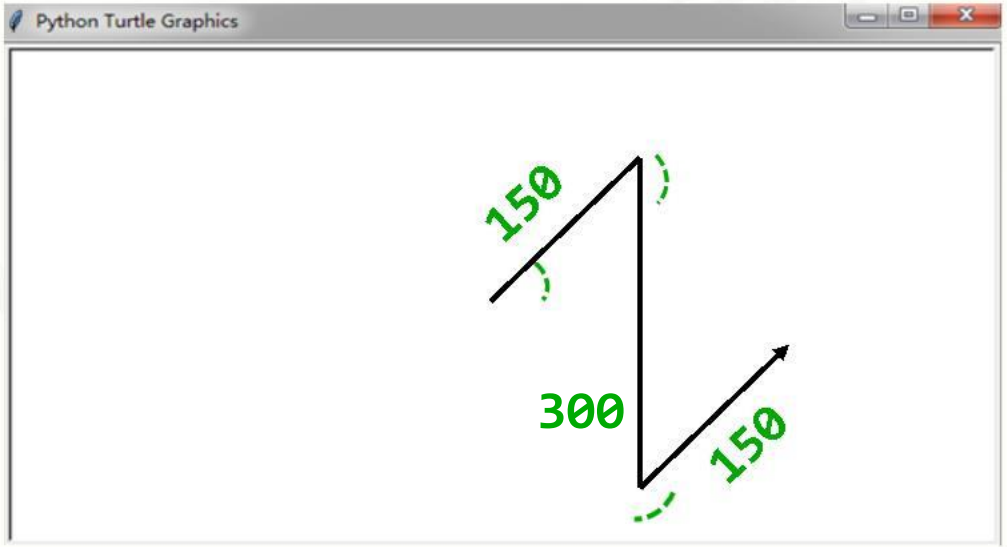
Turtle角度坐标体系

海龟角度



Turtle角度坐标体系

```
import turtle
turtle.left(45)
turtle.fd(150)
turtle.right(135)
turtle.fd(300)
turtle.left(135)
turtle.fd(150)
```

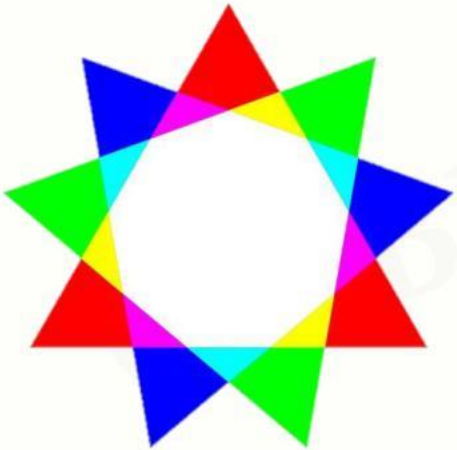


RGB色彩体系

CC BY-NC-SA 4.0 高天

RGB色彩模式

由三种颜色构成的万物色



- RGB指红蓝绿三个通道的颜色组合
- 覆盖视力所能感知的所有颜色
- RGB每色取值范围0-255整数或0-1小数

BY-NC-SA 4.0 嵩天

常用RGB色彩

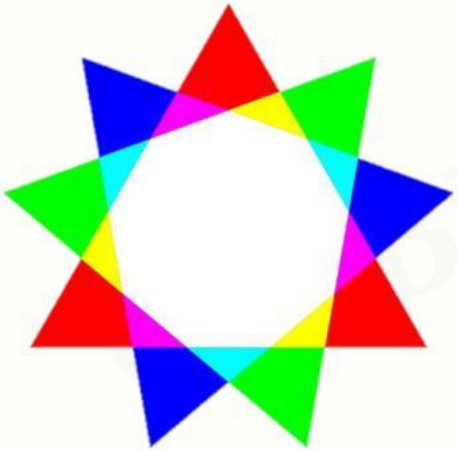
英文名称	RGB整数值	RGB小数值	中文名称
white	255, 255, 255	1, 1, 1	白色
yellow	255, 255, 0	1, 1, 0	黄色
magenta	255, 0, 255	1, 0, 1	洋红
cyan	0, 255, 255	0, 1, 1	青色
blue	0, 0, 255	0, 0, 1	蓝色
black	0, 0, 0	0, 0, 0	黑色

常用RGB色彩

英文名称	RGB整数值	RGB小数值	中文名称
seashell	255, 245, 238	1, 0.96, 0.93	海贝色
gold	255, 215, 0	1, 0.84, 0	金色
pink	255, 192, 203	1, 0.75, 0.80	粉红色
brown	165, 42, 42	0.65, 0.16, 0.16	棕色
purple	160, 32, 240	0.63, 0.13, 0.94	紫色
tomato	255, 99, 71	1, 0.39, 0.28	番茄色

turtle的RGB色彩模式

默认采用小数值 可切换为整数值



```
turtle.colormode(mode)
```

- **1.0：RGB小数值模式**
- **255：RGB整数值模式**

BY-NC-SA 4.0 嵩天

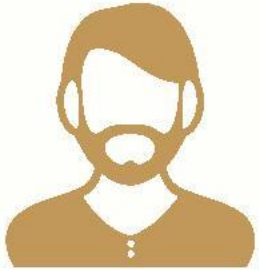
模块1: turtle库的使用

- turtle库的海龟绘图法
- turtle.setup()调整绘图窗体在电脑屏幕中的布局
- 画布上以中心为原点的空间坐标系: 绝对坐标&海龟坐标
- 画布上以空间x轴为0度的角度坐标系: 绝对角度&海龟角度
- RGB色彩体系，整数值&小数值，色彩模式切换

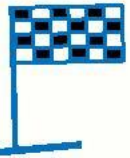
Python语言程序设计

turtle程序语法元素分析

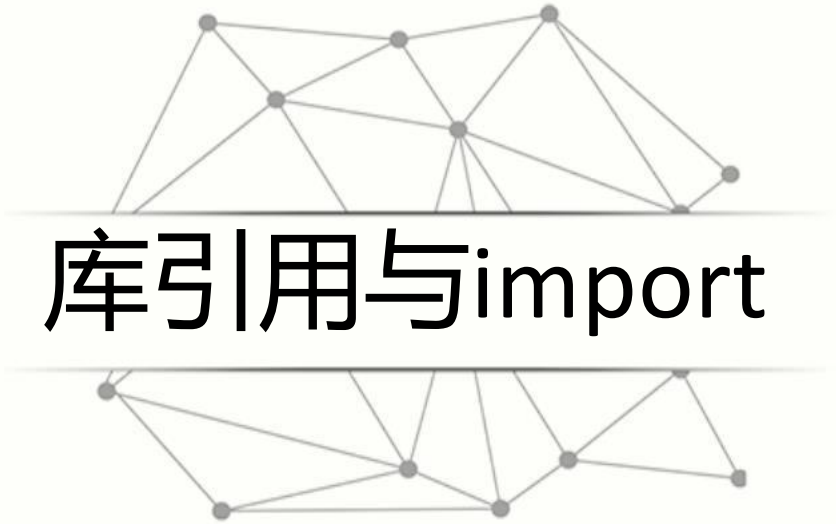
turtle程序语法元素分析



- 库引用与import
- turtle画笔控制函数
- turtle运动控制函数
- turtle方向控制函数
- 基本循环语句
- "Python蟒蛇绘制"代码分析

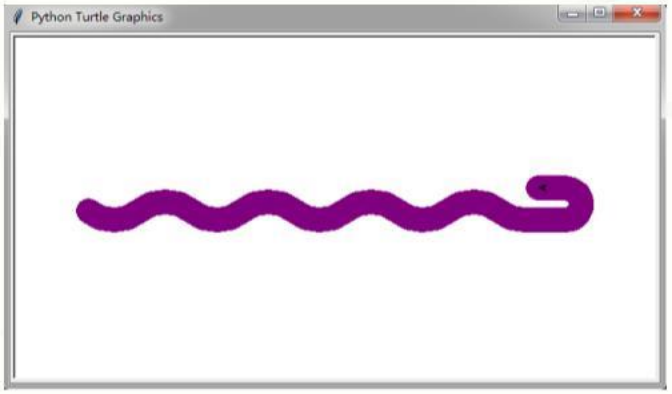


python
语言程序设计



库引用与import

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```



<a>.<b>()的编码风格

python

库引用

扩充Python程序功能的方式

- 使用import保留字完成，采用<a>.<b>()编码风格

import <库名>

<库名>.<函数名>(<函数参数>)

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

引入turtle库

使用turtle库函数

完成功能

可是可是, 好多turtle，很繁琐嘛...

import更多用法

使用from和import保留字共同完成

from <库名> import <函数名>

from <库名> import *

<函数名>(<函数参数>)

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

```
from turtle import *
setup(650, 350, 200, 200)
penup()
fd(-250)
pendown()
pensize(25)
pencolor("purple")
seth(-40)
for i in range(4):
    circle(40, 80)
    circle(-40, 80)
circle(40, 80/2)
fd(40)
circle(16, 180)
fd(40 * 2/3)
done()
```

import更多用法

两种方法比较

import <库名>

<库名>.<函数名>(<函数参数>)

from <库名> import <函数名>

from <库名> import *

<函数名>(<函数参数>)

第一种方法不会出现函数重名问题，第二种方法则会出现

import更多用法

使用import和as保留字共同完成

import <库名> as <库别名>

<库别名>.<函数名>(<函数参数>)

给调用的外部库关联一个更短、更适合自己的名字

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

```
import turtle as t
t.setup(650, 350, 200, 200)
t.penup()
t.fd(-250)
t.pendown()
t.pensize(25)
t.pencolor("purple")
t.seth(-40)
for i in range(4):
    t.circle(40, 80)
    t.circle(-40, 80)
t.circle(40, 80/2)
t.fd(40)
t.circle(16, 180)
t.fd(40 * 2/3)
t.done()
```

turtle画笔控制函数

cc

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

penup(), pendown()

pensize(), pencolor()

CC BY 4.0 高大

python

画笔控制函数

画笔操作后一直有效，一般成对出现

- **turtle.penup()　　别名　　turtle.pu()**

抬起画笔，海龟在飞行

- **turtle.pendown()　　别名　　turtle.pd()**

落下画笔，海龟在爬行

画笔控制函数

画笔设置后一直有效，直至下次重新设置

- turtle.pensize(width)　别名　turtle.width(width)

  画笔宽度，海龟的腰围

- turtle.pencolor(color) color为颜色字符串或r,g,b值

  画笔颜色，海龟在涂装

画笔控制函数

pencolor(color)的color参与可以有三种形式

- **颜色字符串　：** turtle.pencolor("purple")
- **RGB的小数值：** turtle.pencolor(0.63, 0.13, 0.94)
- **RGB的元组值：** turtle.pencolor((0.63,0.13,0.94))

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

penup()

pendown()

pensize(width)

pencolor(colorstring)

pencolor(r,g,b)

pencolor((r,g,b))

turtle运动控制函数

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

fd()

circle()

运动控制函数

控制海龟行进：走直线 & 走曲线

- turtle.forward(d)　别名　turtle.fd(d)

向前行进，海龟走直线

- d: 行进距离，可以为负数

运动控制函数

控制海龟行进：走直线 & 走曲线

- turtle.circle(r, extent=None)

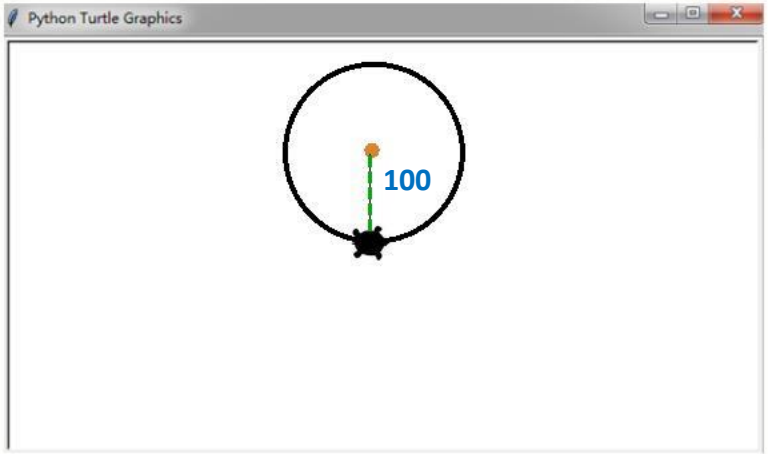
  根据半径r绘制extent角度的弧形

- r: 默认圆心在海龟左侧r距离的位置

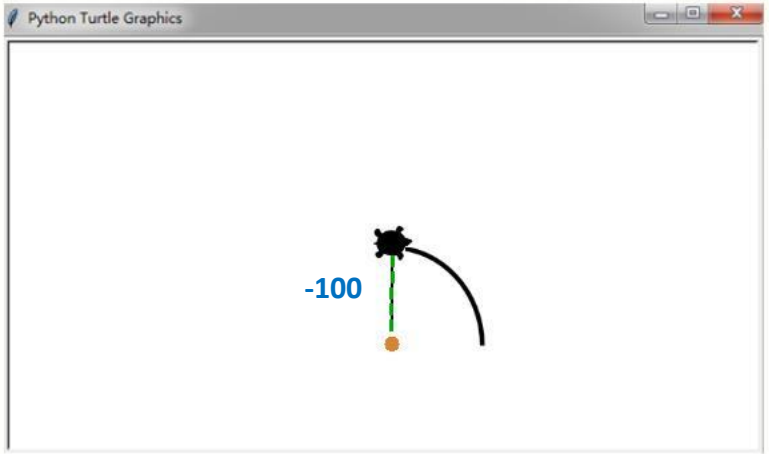
- extent: 绘制角度，默认是360度整圆

运动控制函数

```
turtle.circle(100)
```




```
turtle.circle(-100,90)
```



```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

fd(d)

circle(r,extent=None)

运动控制函数

画笔设置后一直有效，直至下次重新设置

- **turtle.forward(d)　别名　turtle.fd(d)**

  向前行进，海龟走直线

- **d: 行进距离，可以为负数**

turtle方向控制函数

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

seth()

方向控制函数

控制海龟面对方向: 绝对角度 & 海龟角度

- turtle.setheading(angle)　　别名　　turtle.seth(angle)

  改变行进方向，海龟走角度

- angle: 行进方向的绝对角度

方向控制函数

```
turtle.seth(45)
```


```
turtle.seth(-135)
```

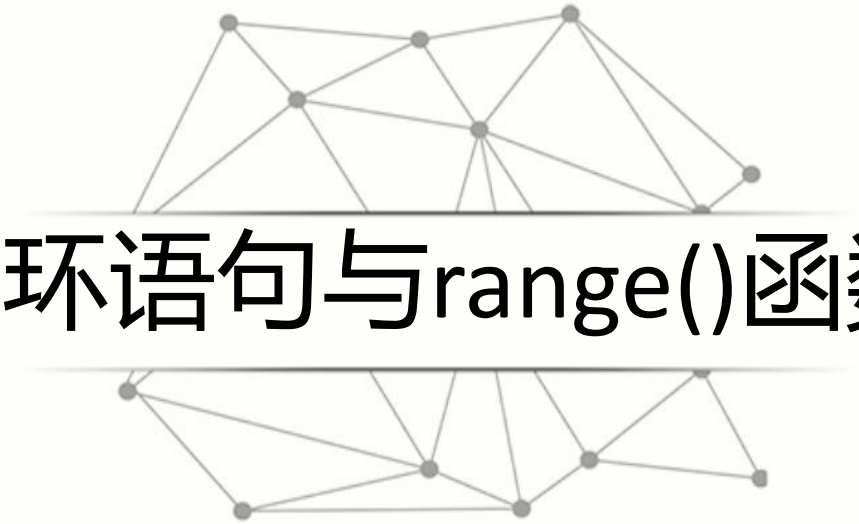
方向控制函数

控制海龟面对方向: 绝对角度 & 海龟角度

- turtle.left(angle)　　海龟向左转
- turtle.right(angle)　　海龟向右转
- angle: 在海龟当前行进方向上旋转的角度

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

seth(angle)



循环语句与range()函数

cc

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

for 和 *in* 保留字

range()

循环语句

按照一定次数循环执行一组语句

for <变量> in range(<次数>):

<被循环执行的语句>

- **<变量>表示每次循环的计数，0到<次数>-1**

循环语句

```
>>> for i in range(5):
        print(i)

0
1
2
3
4
```

```
>>> for i in range(5):
        print("Hello:",i)

Hello: 0
Hello: 1
Hello: 2
Hello: 3
Hello: 4
```

range()函数

产生循环计数序列

- range(N)

  产生 0 到 N-1的整数序列，共N个

- range(M，N)

  产生 M 到 N-1的整数序列，共N-M个

```
range(5)
    0, 1, 2, 3, 4
range(2, 5)
    2, 3, 4
```

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

for i in range(N):

range(N)

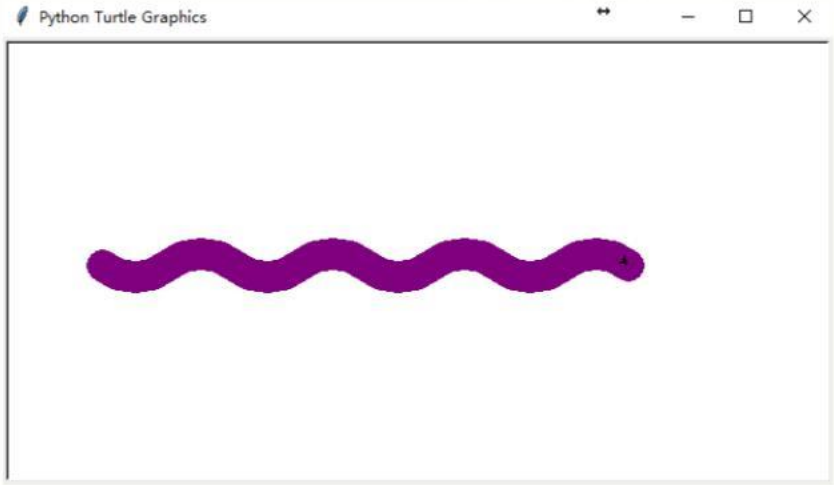
range(M, N)

"Python蟒蛇绘制"代码分析

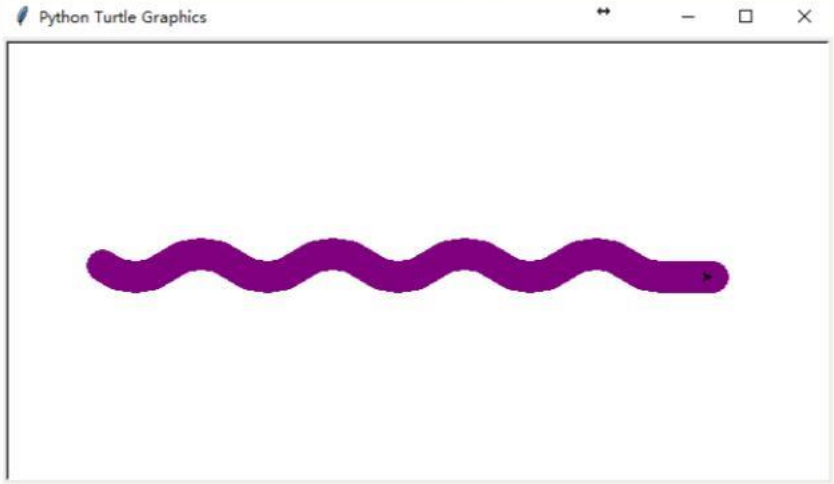
CC BY NC SA

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

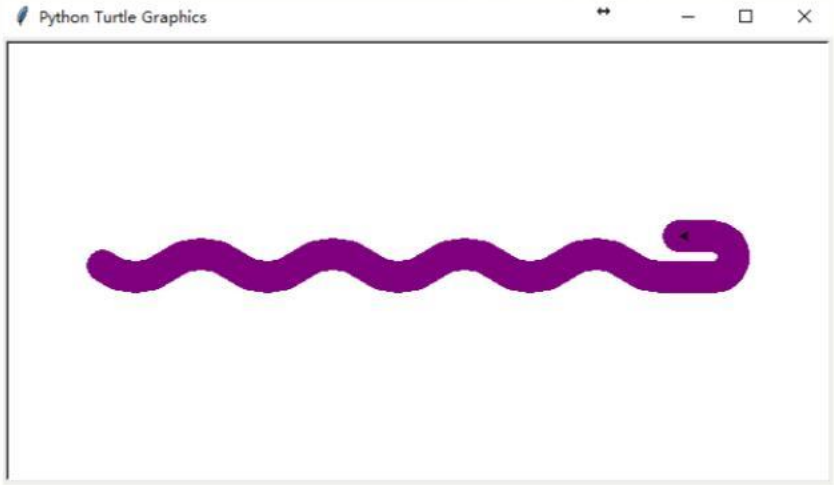
```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```



```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```

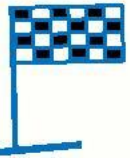


```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)
turtle.done()
```



turtle程序语法元素分析

- **库引用: import、 from…import、 import…as…**
- **penup()、 pendown()、 pensize()、 pencolor()**
- **fd()、 circle()、 seth()**
- **循环语句： for和in、 range()函数**



python
语言程序设计