# Python第二课——认识小海龟

1. 教学目标
2. 培养学生“分析问题——程序设计——编程执行”的计算思维能力。
3. 掌握海龟作图前进、后退、左转、右转等基本操作。
4. 学生能发现生活中的美好事物素材，提高收集信息的能力，加深对用编程解决问题的兴趣。
5. 教学重点

掌握海龟作图的导入、设置图形、填充颜色等基本操作。

1. 教学难点

海龟作图的基本操作的应用。

1. 教学方法

讲授法、任务驱动法、演示法。

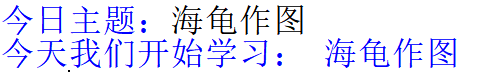
1. 教学过程
2. 导入

通过两个例子复习上一节课的内容并导入这一节课。

任务18：输出以下图案。



任务19：编写代码，输入“海龟作图”，出现以下结果。



1. 教学过程
2. 初探小海龟

海龟作图是python内置的模块，在使用前需要导入小海龟。怎么导入小海龟呢？一共有三种方法。

1. import turtle 通过模块名来使用其中的方法、属性。
2. import turtle as t 通过别名t来使用其中的方法、属性。
3. from turtle import \* 可以直接使用其中的方法、属性。

执行完后，发现并没有出现小海龟。这是因为我们还没召唤出小海龟。

怎么召唤出小海龟呢？需要用到shape（）函数。怎么写代码呢？根据你导入的方法不同而分别召唤。

1. import turtle 通过模块名来使用其中的方法、属性。

turtle.shape(“turtle”)

1. import turtle as t 通过别名t来使用其中的方法、属性。

t.shape(“turtle”)

1. from turtle import \* 可以直接使用其中的方法、属性。

shape(“turtle”)

任务20：用以上三种方法召唤出小海龟。

shape（）函数在括号里加“召唤图形”，可以召唤出不同图案的小海龟。当然，图案不是你想召唤什么就能出来什么图案，比如在“”中写pig，会报错，显然我们并不能召唤出猪。那我们就简单尝试一下可以召唤的几个图案，

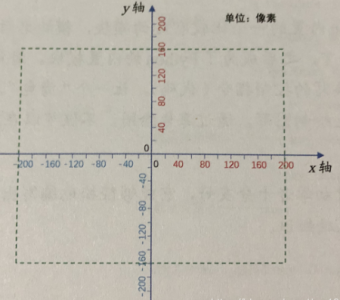
召唤图形名称：默认是箭头，turtle是海龟，circle是圆形，triangle是三角形，square是正方形。

任务21：选择一种自己喜欢的召唤方式，尝试召唤出不同形状的小海龟。

1.1了解海龟作图坐标系

海龟绘图采用的是平面坐标系，即画布(窗口)的中心为原点(0,0)，横向为 x 轴，纵向为 y 轴，x 轴控制水平位置，y 轴控制垂直位置。总的来说跟数学上的坐标系没有什么实际区别。

例如，一个 400×320 的画布，对应的坐标系如下图所示。



绿色的长方形框就是我们画布的大小，坐标为（0,0）的位置也就是原点是我们画布的中心。我们可以看到长方形是关于x轴和y轴对称的。所以在长方形框内x轴在-200~200移动，y轴在-160~160之间移动。

1.2海龟作图的三要素

海龟作图有三个关键要素，即方向、位置和画笔。在作图时，主要是控制这些关键要素来绘制出需要的图形。

1. 方向：在进行海龟绘图时，方向主要用于控制海龟的移动方向。

主要有以下三种方法：

left()/lt()

right()/rt()

setheading()/seth()

1. 位置:在进行海龟绘图时，位置主要用于控制海龟移动的距离。主要有以下 6 个方法进行设置：
2. forward(distance)
3. backward(distance)
4. goto(x,y)
5. setx(x)
6. sety(y)
7. home()
8. 画笔:海龟绘图中的画笔就相当于现实生活中绘图所用的画笔。在海龟绘图中，通过画笔可以控制线条的粗细、颜色和运动的速度。

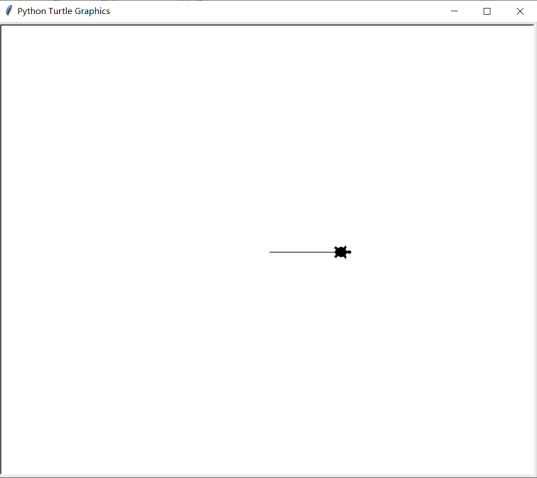
1.3基于原点绘制一只会动的海龟

教师演示小海龟走直线。

第一步：介绍forward（distance）/fd()，海龟前进 distance 指定的距离，方向为海龟的朝向。

第二步：召唤小海龟，小海龟向前走100步

任务22：小海龟向前走100步



代码：

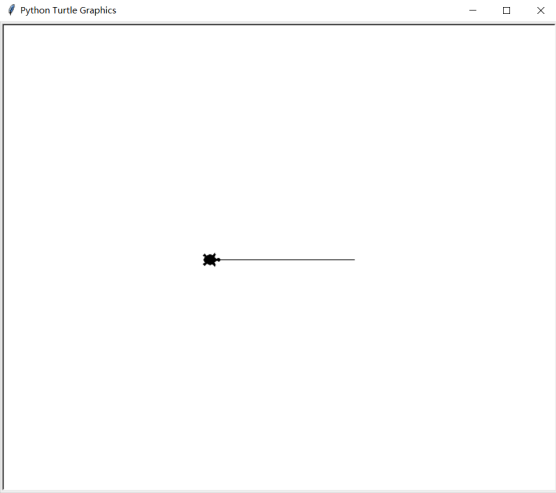
from turtle import \*

shape("turtle")

forward(100)

第三步:介绍backward（）/bk()/back()，海龟后退 distance 指定的距离，方向与海龟的朝向相反。不改变海龟的朝向。

任务23：接任务22，小海龟在向前走100步之后，后退200步。



代码：

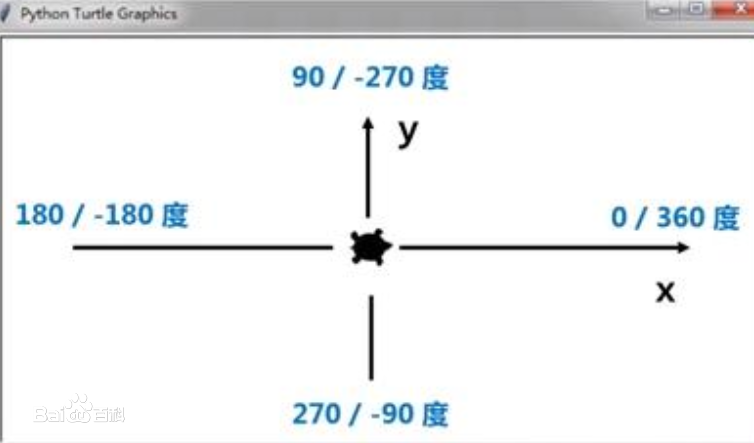
from turtle import \*

shape("turtle")

forward(100)

backward(200)

第四步：介绍左转右转函数，但在这之前，介绍角度旋转问题。

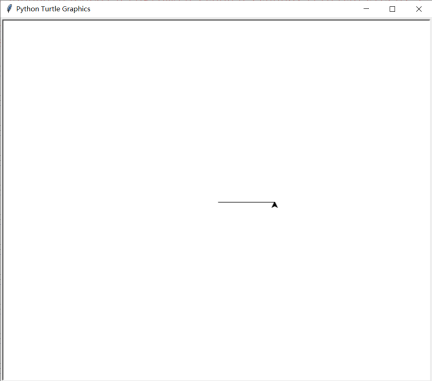


注意起始位置箭头指向的方向，之后介绍right（angle）/rt()和left(angle)/lt()。

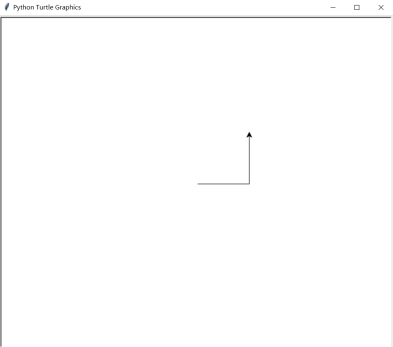
海龟右转 angle 个单位。单位默认为角度，单位可以改变，这个以后深入再细说。

任务24：输出一个直角。

教师指导完成：



箭头方向相较于起始方向向右旋转了90度。



代码：

from turtle import \*

shape()

forward(100)

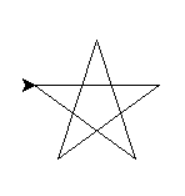
left(90)

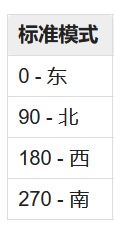
forward(100)

任务25：顺时针输出一个正方形

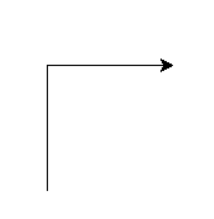
任务26：逆时针输出一个正方形

提升任务(27)：绘制五角星。提示：五角星内角36度，外角144度。



第五步：介绍setheading(to\_angle)函数。设置海龟的朝向为 to\_angle。以下是以角度表示的几个常用方向（标准模式下）：

任务28：输出以下图形



代码：

from turtle import \*

setheading(90)

forward(100)

right(90)

forward(100)

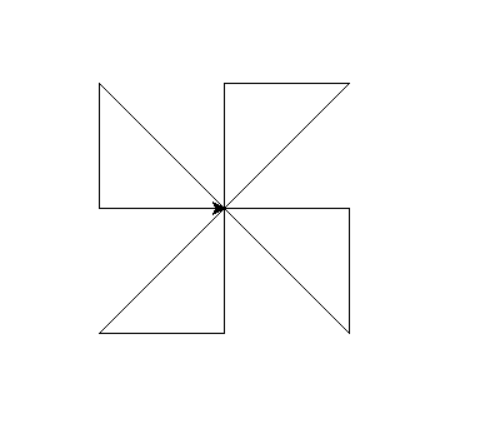
第六步：思考一下：如果我想让任务26的首尾相连成为三角形，我们应该怎么办？(29)

方法一：计算旋转角度，让小海龟爬过去

方法二：让小海龟回到原点（两点之间线段最短），该方法并不适用于所有图形。

home（）函数:海龟移至初始坐标 (0,0)，并设置朝向为初始方向。

任务30：输出以下图案



代码：

from turtle import \*

shape()

setheading(90)

forward(100)

right(90)

forward(100)

home()

forward(100)

right(90)

forward(100)

home()

setheading(-90)

forward(100)

right(90)

forward(100)

home()

setheading(180)

forward(100)

right(90)

forward(100)

home()

1.4根据坐标绘制图形

第一步：

介绍goto(x,y=None)/setpos(x,y=None)/setposition(x,y=None)。括号里的参数：x：一个数值或数值对/向量， y：一个数值或 None。

如果 y 为 None，x 应为一个表示坐标的数值对或 Vec2D 类对象 (例如 pos() 返回的对象).

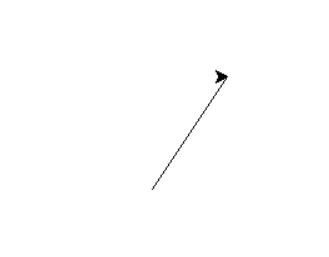
海龟移动到一个绝对坐标。如果画笔已落下将会画线。不改变海龟的朝向。

代码：

from turtle import \*

shape()

goto(60,90)



第二步：

介绍setx(x):

参数:x :一个数值 (整型或浮点型)

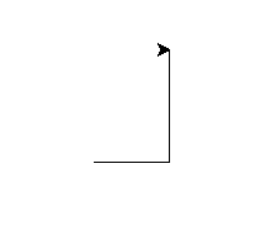
设置海龟的横坐标为 x，纵坐标保持不变。

sety(y):

参数: y:一个数值 (整型或浮点型)

设置海龟的纵坐标为 y，横坐标保持不变。

教师演示：



代码：

from turtle import \*

shape()

setx(60)

sety(90)

第三步：查看坐标位置

position()/pos():返回海龟当前的坐标 (x,y) (为 Vec2D 矢量类对象)。

xcor():返回海龟的 x 坐标。

ycor():返回海龟的 y 坐标。

heading():返回海龟当前的朝向。

distance(x,y=None):返回从海龟位置到由 (x,y)，适量或另一海龟对应位置的单位距离。

参数:x -- 一个数值或数值对/矢量，或一个海龟实例

y -- 一个数值——如果 x 是一个数值，否则为 None.

尝试：加print才能看见。

import turtle

turtle.home()

print(turtle.distance(30,40))

import turtle

print(turtle.distance((30,40)))

import turtle

joe = turtle.Turtle()

joe.forward(77)

print(turtle.distance(joe))

第四步：我想不再是一笔画的图形，该怎么办？——隐藏运动轨迹就好了。

pendown()|pd()|down():画笔落下 -- 移动时将画线。

penup()|pu()|up():画笔抬起 -- 移动时不画线。

我也想隐藏海龟怎么办？

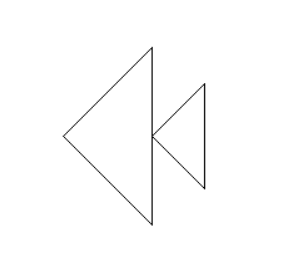
showturtle()|st():使海龟可见。

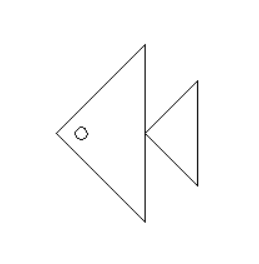
hideturtle()|ht():使海龟不可见。当你绘制复杂图形时这是个好主意，因为隐藏海龟可显著加快绘制速度。

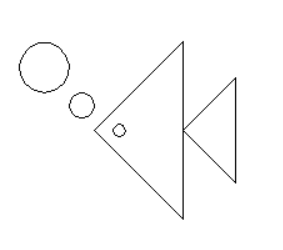
isvisible():如果海龟显示返回 True，如果海龟隐藏返回 False。

实例：画一条会吐泡泡的鱼。

1. 先画鱼身：



1. 画鱼眼睛：circle（radius，extent,steps）:绘制一个 radius 指定半径的圆。圆心在海龟左边 radius 个单位；extent 为一个夹角，用来决定绘制圆的一部分。circle(5):画一个半径为5的圆。
2. 
3. 画泡泡：



代码：

from turtle import \*

shape()

lt(45)

fd(100)

rt(135)

fd(141.4)

rt(135)

fd(100)

rt(135)

penup()

goto(70.7,0)

pendown()

lt(45)

fd(60)

rt(135)

fd(84.8)

rt(135)

fd(60)

rt(135)

penup()

goto(20,-5)

pendown()

circle(5)

penup()

goto(-10,10)

pendown()

circle(10)

penup()

goto(-40,30)

pendown()

circle(20)

ht()

1.5画笔设置

第一步：pencolor(“color”):设置画笔颜色.参数为英文的颜色名，也可以是颜色代码（#FFFFFF）。

pencolor():返回画笔颜色。

第二步：pensize():设置画笔的粗细，参数的数值越大，画笔越粗。

第三步：fillcolor():返回或设置填充颜色,参数为英文的颜色名，也可以是颜色代码（#FFFFFF）。

第四步：填充绘制图形颜色。begin\_fill（）：在绘制要填充的形状之前调用。end\_fill()：填充上次调用 begin\_fill() 之后绘制的形状。

实践：尝试让画笔粗细为5像素、颜色为红色。鱼填充颜色为蓝色，泡泡为粉色。

第五步：添加字体

write( arg, move=False, align=’left’, font=( fontname、fontsize、fonttype))

arg——必选参数，用于指定要输出的文字内容，该内容会输出到当前海龟光标的所在位置。

Move——可选参数，用于指定是否移动画笔到文本的右下角，默认为False (不移动)，如果move为true，则画笔将移动到文本的右下角。

align——可选参数，用于指定文字的对齐方式，其参数值为left（居左）、center（居中）或者right（居右）中的任意一个，默认为 left（居左）。

font——可选参数，用于指定字体、字号和字形，通过一个三元组(字体fontname、字号fontsize、字形fonttype）指定。其中字体宋体SimSun，黑体SimHei,幼圆YouYuan,隶书LiSu等。

字形fonttype可设置的值为normal（表示正常）、bold（粗体）、italic（斜体）、underline（下划线）等。

实践：用write（）函数写出在“在八中，坚持你就赢了！”要求：move为默认值，文字左对齐，字体为黑体，字体加粗，字号大小为58，颜色为红色。

代码：

from turtle import \*

penup()

goto(-400,0)

pencolor("red")

write("在八中，坚持你就赢了！",move=False, align='left',font=("SimHei",58,"bold"))

提交任务：实现以下图案



代码：

from turtle import \*

pensize(20)

pencolor("red")

circle(130)

penup()

goto(-55,-10)

pendown()

pencolor("black")

write("P",font=("KaiTi",200,"bold"))

penup()

goto(-80,220)

pendown()

pencolor("red")

rt(45)

fd(240)

penup()

goto(-130,-90)

pendown()

pencolor("black")

write("禁止泊车",font=("SimHei",50,"bold"))

ht()

实现任务二：



代码:

from turtle import \*

pensize(20)

pencolor("red")

circle(130)

penup()

goto(-120,50)

pendown()

pencolor("black")

write("100",font=("YouYuan",120,"bold"))

penup()

goto(-110,-90)

pendown()

pencolor("black")

write("限速100",font=("SimHei",50,"bold"))

ht()

1.5画布设置

第一步：bgcolor(args): 设置或返回 TurtleScreen 的背景颜色。

参数：args可以是英文的颜色名，也可以是颜色代码（#FFFFFF）。

第二步：setup(width=200,height=200,startx=0,starty=0)：设置画布大小。

参数：

width -- 如为一个整型数值，表示大小为多少像素，如为一个浮点数值，则表示屏幕的占比；默认为屏幕的 50%

height -- 如为一个整型数值，表示高度为多少像素，如为一个浮点数值，则表示屏幕的占比；默认为屏幕的 75%

startx -- 如为正值，表示初始位置距离屏幕左边缘多少像素，负值表示距离右边缘，None 表示窗口水平居中

starty -- 如为正值，表示初始位置距离屏幕上边缘多少像素，负值表示距离下边缘，None 表示窗口垂直居中。

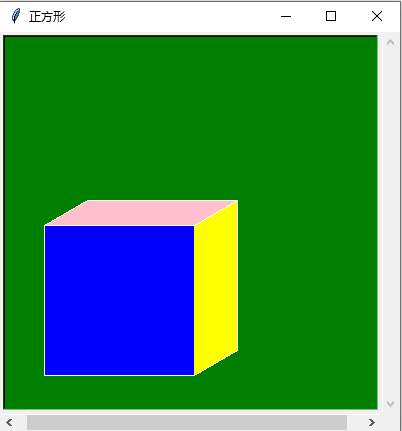
第三步：speed()：设置画笔的速度。

参数：0到10，数越大速度越快。

第四步：title(titlestring): 设置海龟窗口标题为 titlestring 指定的文本。

参数: titlestring -- 一个字符串，显示为海龟绘图窗口的标题栏文本。

动手实践：出现以下界面。



代码：

from turtle import \*

title("正方形")

setup(400,400)

bgcolor("green")

pencolor("white")

fillcolor("yellow")

begin\_fill()

lt(30)

fd(50)

rt(120)

fd(150)

rt(60)

fd(50)

end\_fill()

fillcolor("blue")

begin\_fill()

rt(30)

fd(150)

rt(90)

fd(150)

rt(90)

fd(150)

end\_fill()

fillcolor("pink")

begin\_fill()

lt(30)

fd(50)

lt(150)

fd(150)

lt(30)

fd(50)

end\_fill()

ht()