

Python语言项目教学04

陈斌

北京大学地球与空间科学学院


2018.11.09

Python语言项目教学培训课程总体安排

日期	上午	下午
Day1	信息技术为基础的创新教育	Python语言概览和教学实践交流
Day2	Python 语言基本数据类型	输入输出、控制流和程序结构
Day3	扩展模块（时间/算术/持久化/数据库）	高级特性（面向对象/异常处理/迭代器/生成器等）
Day4	扩展模块（数值计算/网络/可视化）	Python科学编程实践项目
Day5	Python艺术编程教学	艺术编程实践项目
Day6	Python开源硬件基础	开源硬件实验
Day7	Python开源硬件编程教学	开源硬件编程实践项目
Day8	实习项目分组讨论、开发	编程开发、展示和总结

Python引用扩展模块：import

- import <模块> [as <别名>]
 - 将模块中的函数等名称导入当前程序
 - “命名空间” namespace
 - 引用方法：<模块>.<名称>
- dir(<名称>)函数
 - 列出名称的属性
- help(<名称>)函数
 - 显示参考手册
- from <模块> import <名称>
 - 导入模块的部分名称




```
>>> import time
>>> dir(time)
['_STRUCT_TM_ITEMS', '__doc__', '__loader__', '__name__', '__package__', '__spec__', 'altzone', 'asctime', 'clock', 'ctime', 'daylight', 'get_clock_info', 'gmtime', 'localtime', 'mktime', 'monotonic', 'perf_counter', 'process_time', 'sleep', 'strftime', 'strptime', 'struct_time', 'time', 'timezone', 'tzname', 'tzset']
>>> time.tzname
('CST', 'CST')
>>> help(time.time)
Help on built-in function time in module time:

time(...)
    time() -> floating point number

    Return the current time in seconds since the Epoch.
    Fractions of a second may be present if the system
    clock provides them.

>>> print(time.time())
1490280256.450634
```



时间相关：calendar模块

- 跟日历相关的若干函数和类，可以生成文本形式的日历
- `calendar.calendar(<年>)`
- `calendar.month(<年>,<月>)`
 - 返回多行字符串
- `calendar.isleap(<年>)`
 - 判别闰年
- `calendar.prmonth(<年>,<月>)`
- `calendar.prcal(<年>)`

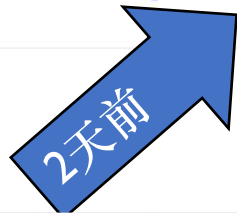
```
>>> import calendar
>>> calendar.month(2017,3)
'   March 2017\nMo Tu We Th Fr Sa Su\n   1  2  3  4  5\n  6  7  8  9 10 11 12\n 13 14 15 16 17 18 19\n 20 21 22 23 24 25 26\n 27 28 29 30 31\n'
>>> print (calendar.month(2017,3))
   March 2017
Mo Tu We Th Fr Sa Su
   1  2  3  4  5
  6  7  8  9 10 11 12
 13 14 15 16 17 18 19
 20 21 22 23 24 25 26
 27 28 29 30 31

>>> calendar.isleap(2017)
False
>>> |
```

时间相关：datetime模块

- 有4个主要的类
 - date处理年月日
 - time处理时分秒、毫秒
 - datetime处理日期加时间
 - timedelta处理时段（时间间隔）
- 常用函数／方法
 - datetime.date.today()
 - datetime.datetime.now()
 - datetime.datetime.isoformat()
- 两个时间相减就是timedelta

```
>>> import datetime
>>> datetime.date.today()
datetime.date(2018, 11, 8)
>>> d=datetime.datetime.now()
>>> d
datetime.datetime(2018, 11, 8, 23, 41, 36, 300425)
>>> d.isoformat()
'2018-11-08T23:41:36.300425'
>>> d-datetime.timedelta(days=2)
datetime.datetime(2018, 11, 6, 23, 41, 36, 300425)
>>>
```



```
>>> d2=datetime.datetime.now()
>>> d2-d
datetime.timedelta(0, 91, 230385)
>>> d2
datetime.datetime(2018, 11, 8, 23, 43, 7, 530810)
>>> dt=d2-d
>>> type(dt)
<class 'datetime.timedelta'>
```


时间相关：time模块

- `time.time()`浮点数表示的现在时间
 - 从1970-1-1 0:0:0开始的秒数
- `time.struct_time`结构化时间类
 - `time.localtime(<纪元时间>)->结构`
 - `time.gmtime(<纪元时间>)->结构`
 - `time.mktime(<结构化时间>)->纪元时间`
- `time.strftime(<格式>)`表示格式化输出（结构化）时间
- `time.strptime(<字符串>,<格式>)`按照格式识别字符串，返回时间

```
>>> import time
>>> n = time.time()
>>> n
1490285666.071055
>>> time.localtime(n)
time.struct_time(tm_year=2017, tm_mon=3, tm_mday=24, tm_hour=0, tm_min=14, tm_sec=26, tm_wday=4, tm_yday=83, tm_isdst=0)
>>> time.gmtime(n)
time.struct_time(tm_year=2017, tm_mon=3, tm_mday=23, tm_hour=16, tm_min=14, tm_sec=26, tm_wday=3, tm_yday=82, tm_isdst=0)
>>> p=time.localtime(n)
>>> time.mktime(p)
1490285666.0
>>> p=time.gmtime(n)
>>> time.mktime(p)
1490256866.0
>>> time.strftime("%Y%m%d %H%M%S", p)
'20170323 161426'
>>> time.strptime("20170324", "%Y%m%d")
time.struct_time(tm_year=2017, tm_mon=3, tm_mday=24, tm_hour=0, tm_min=0, tm_sec=0, tm_wday=4, tm_yday=83, tm_isdst=-1)
```

基本模块简介：算术

- math: 常用的算术函数、三角函数、幂指数等等
- cmath: 支持复数的math函数
- decimal: 十进制定点数
 - 十进制小数
 - 不再有浮点数的误差
- fractions: 有理数，比例
 - 进行分数运算

```
>>> import decimal as d
>>> 0.1+0.2
0.30000000000000004
>>> d.Decimal("0.1")+d.Decimal("0.2")
Decimal('0.3')
>>> import fractions as f
>>> f.Fraction("2/3")
Fraction(2, 3)
>>> f.Fraction("2/3")+f.Fraction("3/4")
Fraction(17, 12)
>>>
```

```
>>> from decimal import Decimal as D
>>> D("3.14159265351234567323211234")+D("7.292314563434234235343")
Decimal('10.43390721694657990857511234')
>>> 3.14159265351234567323211234+7.292314563434234235343
10.43390721694658
>>> |
```

基本模块简介：算术

- random: 随机数
 - random.randint(a,b)
 - random.randrange(start, stop, step)
 - random.choice(seq)
 - random.sample(seq, n)
- statistics: 一些统计函数
 - 平均值: mean
 - 中位数: median
 - 标准偏差: stdev/pstdev

```
>>> import random
>>> random.choice(["apple", "pie", "tea"])
'tea'
>>> random.randrange(10)
4
>>> random.randint(1,10)
5
```

```
>>> random.sample(["apple", "pie", "tea", "milk"], 2)
['tea', 'milk']
```

```
>>> import statistics
>>> statistics.mean([12,34,56])
34.0
>>> statistics.median([12,34,56,100])
45.0
>>> statistics.median([12,34,56,100,101])
56
```


持久化: shelve

- 将任何数据对象，保存到文件中
去
- 类似字典形式访问，可读可写
 - `import shelve`
 - `f = shelve.open(<文件名>)`
 - `f[key] = value`
 - `value = f[key]`
 - `del f[key]`
 - `f.close()`

无作用

```
import shelve

d = shelve.open(filename)  # open -- fi
                             # library

d[key] = data               # store data
                             # using an e
data = d[key]               # retrieve a
                             # if no such
del d[key]                  # delete dat
                             # if no such

flag = key in d              # true if th
klist = list(d.keys())      # a list of

# as d was opened WITHOUT writeback=True
d['xx'] = [0, 1, 2]          # this works
d['xx'].append(3)            # *this does

# having opened d without writeback=True
temp = d['xx']                # extracts t
temp.append(5)                # mutates th
d['xx'] = temp                # stores the

# or, d=shelve.open(filename,writeback=
# d['xx'].append(5) and have it work as
# consume more memory and make the d.cl

d.close()                    # close it
```

文本文件读写： 内置文件对象

- 内置的文本文件处理函数

- `f = open(<文件名>, <模式>)`
- `f.readline()`: 返回一行
- `f.readlines()`: 返回所有行, 列表
- `f.writelines(<字符串列表>)`: 写入文本行
- `f.close()`
- `with`语句可以自动调用`close`

```
>>> f=open("my.txt", "w")
>>> f.writelines(["apple\n", "pie\n"])
>>> f.close()
>>> f=open("my.txt", "r")
>>> f.readlines()
['apple\n', 'pie\n']
>>> f.close()
```

数据库操纵模块：sqlite3

- 无服务器端的微型关系数据库
 - 广泛应用在小型系统/移动app
- SQLite数据库访问接口
 - 连接：connect(dbname)
 - 执行SQL：execute(SQL, 参数)
 - executemany(SQL, 参数迭代)
 - executescript(多条SQL)
 - 游标对象cursor迭代获得数据记录
 - 关闭：close()

数据库操纵模块：sqlite3

```
>>> import sqlite3
>>> conn = sqlite3.connect("test.db")
>>> conn.execute("create table person (id text, name text, age integer)")
<sqlite3.Cursor object at 0x1163088f0>
>>> conn.execute("insert into person values (?, ?, ?)", ("A01", "John", 23))
<sqlite3.Cursor object at 0x1163089d0>
>>> plst=[("B01", "Tom", 18), ("B02", "Mike", 20), ("C01", "Alice", 22)]
>>> conn.executemany("insert into person values (?, ?, ?)", plst)
<sqlite3.Cursor object at 0x1163088f0>
>>> conn.commit()
```

插入多条记录

数据库操纵模块：sqlite3

```
>>> cur = conn.execute("select id, name, age from person where name=?", ("Tom",))
>>> for r in cur:
    print (r)

('B01', 'Tom', 18)
>>> cur = conn.execute("delete from person where id=?", ("A01",))
>>> cur = conn.execute("select * from person")
>>> for r in cur:
    print (r)

('B01', 'Tom', 18)
('B02', 'Mike', 20)
('C01', 'Alice', 22)
>>> conn.close()
>>> |
```

数据库操纵模块：sqlite3

- 查看数据库中有哪些表

```
>>> conn = sqlite3.connect("test.db")
>>> cur = conn.execute("select * from sqlite_master")
>>> for r in cur:
    print (r)
```

```
('table', 'person', 'person', 2, 'CREATE TABLE person
```

- 查看表的结构

```
>>> cur = conn.execute("pragma table_info(person)")
>>> for r in cur:
    print (r)
```

```
(0, 'id', 'text', 0, None, 0)
(1, 'name', 'text', 0, None, 0)
(2, 'age', 'integer', 0, None, 0)
>>>
```

数据库操纵模块：sqlite3

- 数据类型的对应关系
- SQLite支持类型
 - NULL, TEXT, INTEGER
 - REAL, BLOB
- Python对应的类型
 - None, str, int
 - float, bytes
- 很多SQLite数据库图形化工具
 - SQLiteStudio

- 执行SQL语句脚本
 - `executescript()` (多条SQL)

```
import sqlite3

con = sqlite3.connect(":memory:")
cur = con.cursor()
cur.executescript("""
    create table person(
        firstname,
        lastname,
        age
    );

    create table book(
        title,
        author,
        published
    );

    insert into book(title, author, published)
    values (
        'Dirk Gently''s Holistic Detective Agency',
        'Douglas Adams',
        1987
    );
""")
```

SQLiteStudio



SQLiteStudio v3.1.1

自由，开源，跨平台的 SQLite 数据库管理工具。

<http://sqlitestudio.pl>

MacOS X 应用版。

作者和活跃维护人：

SalSoft (<http://salsoft.com.pl>)

SQLiteStudio (3.1.1)

数据库

过滤名

- mytest (错误)
- test (SQLite 3)
 - Tables (1)
 - person
 - 字段 (3)
 - id
 - name
 - age
 - Indexes
 - 触发器
 - Views

person (test)

结构 数据 约束 Indexes 触发器 DDL

Grid view Form view

1

	id	name	age
1	A01	John	23
2	B01	Tom	18
3	B02	Mike	20
4	C01	Alice	22

Filter data

上机练习

- 给算法计时，看看阶乘累加 ($n=1\sim100$) 各需要多长时间？
- 将一篇文章写入一个文本文件。
 - 读出文本文件，统计单词数输出。
 - 读出文本文件，随机输出其中的10个单词。

图形用户界面：easygui

- 可以显示各种对话框、文本框、选择框与用户交互

- `easygui.egdemo()` 演示
- `easygui.msgbox`
- `easygui.fileopenbox`
- `easygui.choicebox`
- `easygui.textbox`
- `easygui.passwordbox`

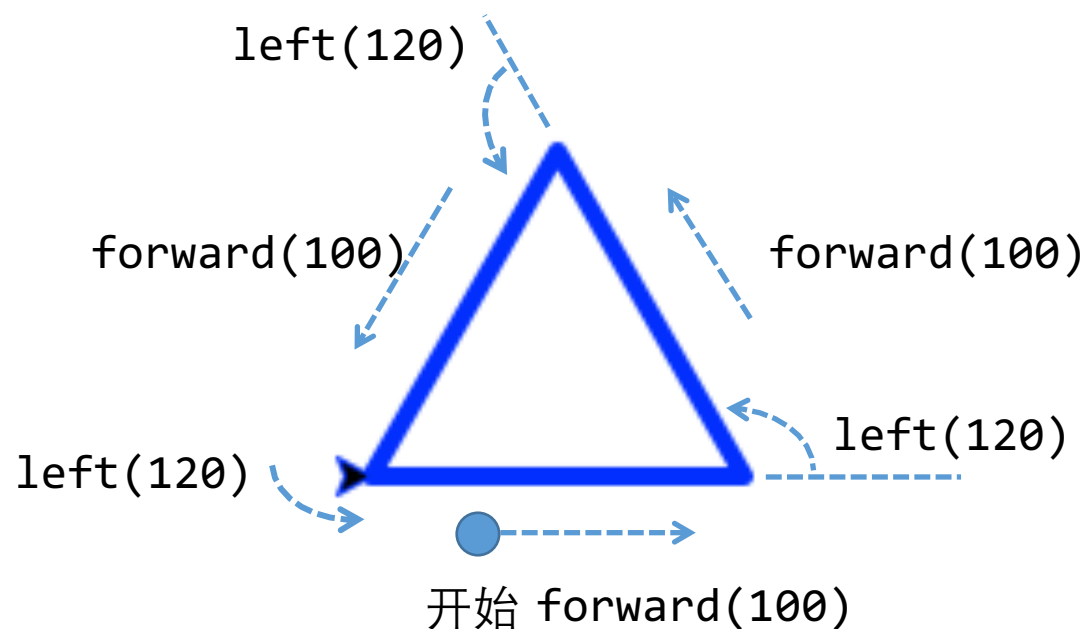
- 可以做出简单的图形界面程序

```
import easygui as g
import sys
while 1:
    g.msgbox('嗨, 欢迎进入第一个GUI制作的小游戏~')
    msg = '你希望学习到什么知识呢?'
    title = '互动小游戏'
    choices = ['琴棋书画', '四书五经', '程序编写', '逆向分析']
    choice = g.choicebox(msg, title, choices)
    # note that we convert the choice to string, in case the user
    # cancelled the choice, and we got None.
    g.msgbox('你的选择是: ' + str(choice), '结果')
    msg = '你希望重新开始小游戏么?'
    title = '请选择'
    if g.ccbox(msg, title):           # Show a Continue/Cancel dialog
        pass                          # user choose Continue
    else:                             # user choose Cancel
        sys.exit(0)
```

海龟做图：turtle

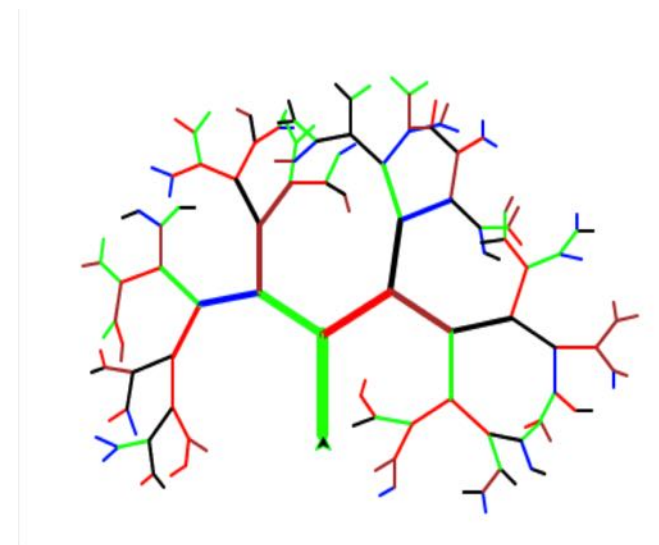
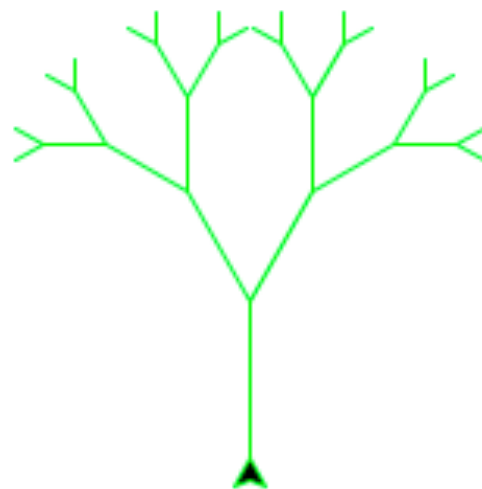
- 模拟海龟在沙滩上爬行所描绘的轨迹，从LOGO语言借鉴而来
- 海龟：显隐、外形、是否动画
- 画笔：抬起落下、颜色、粗细

```
import turtle # 导入turtle模块
p = turtle.Pen() # 创建一支画笔（海龟）
p.pencolor('blue') # 设置画笔颜色为蓝色
p.pensize(5) # 设置画笔的粗细为5
p.forward(100) # 最初画笔（海龟）朝向正右方，向前画长度为100的直线
p.left(120) # 画笔（海龟）向左转120度
p.forward(100) # 向前画长度为100的直线
p.left(120) # 画笔（海龟）向左转120度
p.forward(100) # 向前画长度为100的直线
p.left(120) # 画笔（海龟）向左转120度
```

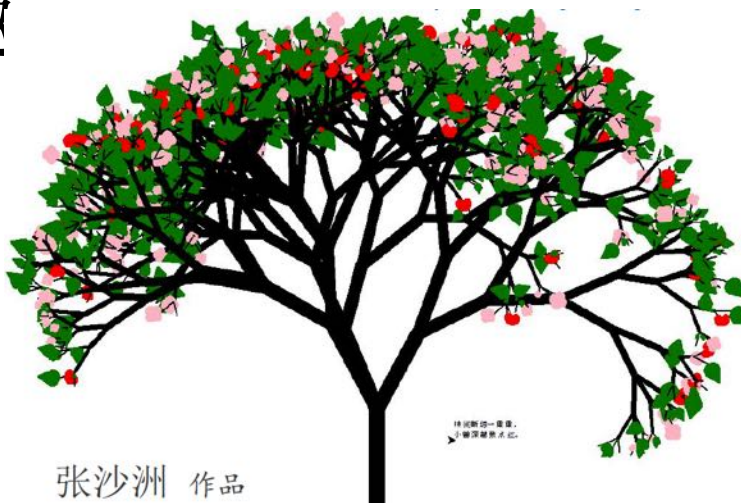


海龟做图：turtle

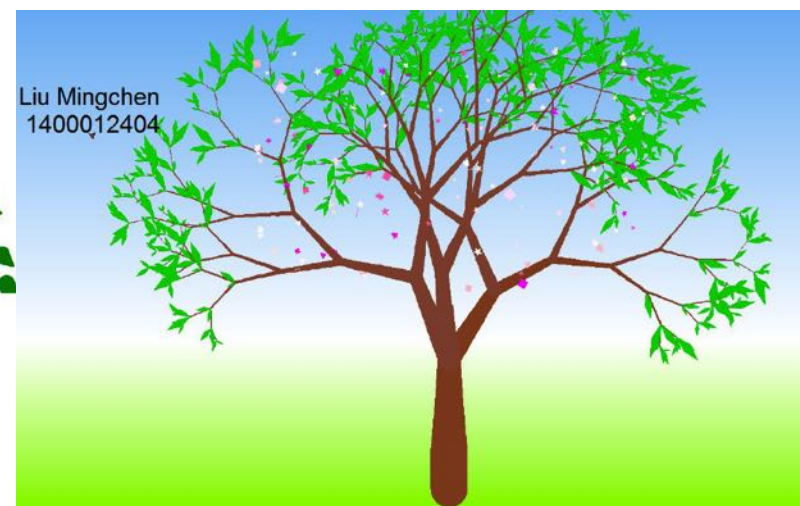
- 无需任何繁琐代码，立即开始绘图
- 海龟的外形可以设定，多个海龟，能够实现动画制作



李然 作品

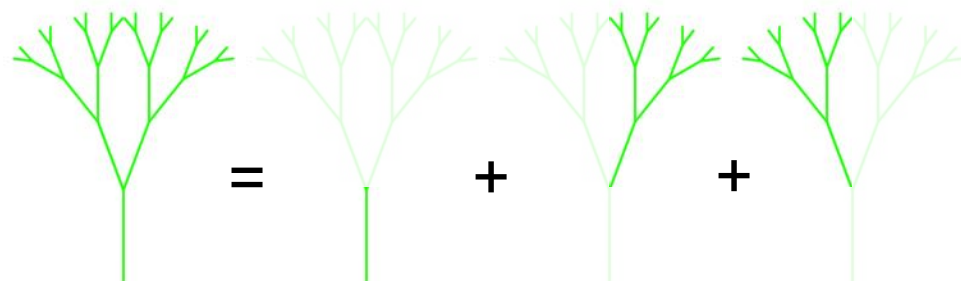


张沙洲 作品



程序: tree.py

```
1  import turtle
2
3
4  def tree(branch_len):
5      if branch_len > 5: # 树干太短不画, 即递归结束条件
6          t.forward(branch_len) # 画树干
7          t.right(20) # 右倾斜20度
8          tree(branch_len - 15) # 递归调用, 画右边的小树, 树干减15
9          t.left(40) # 向左回40度, 即左倾斜20度
10         tree(branch_len - 15) # 递归调用, 画左边的小树, 树干减15
11         t.right(20) # 向右回20度, 即回正
12         t.backward(branch_len) # 海龟退回原位置
```



二叉树

树干

倾斜的右小树

倾斜的左小树

```
15  t = turtle.Turtle()
16  t.left(90)
17  t.penup()
18  t.backward(100)
19  t.pendown()
20  t.pencolor('green')
21  t.pensize(2)
22  tree(75) # 画树干长度75的二叉树
23  t.hideturtle()
24  turtle.done()
```

上机练习

- 利用turtle模块，结合easygui，让用户选择一个图形进行绘制
 - 选择“正方形”，绘制一个边长100的绿色正方形
 - 选择“五角星”，绘制一个边长100的红色五角星。

