**day15**

time、datetime、calendar模块的引入讲解(重视)

Unix时间戳(timestamp)：返回得到一个数值类型的数据(python表现浮点值)

知道：记录了时间原点(1970年00点00分00秒)至今的秒值

UTC时间：世界协调时间

GMT时间：格林尼治/格林威治时间

CCT时间：北京时间(当前时间)，属于东八区 【注意】比UTC时间 + 8小时

**1.time模块：**

**演示time模块的使用：常用的函数**

**import** time

**① time()： 返回时间戳对象，数据类型浮点值**ts = time.time()  
print(ts,type(ts)) *#随着时间的变化，输出的也是不断变化的*

*#1547637538.7860892 <class 'float'>*

**② localtime([ts])： 将时间戳数据转换为一个本地元祖对象返回 数据类型：time.struct\_time类型  
【注意】：返回time.struct\_time类型的对象，内部封装维护着9个元素**b = time.localtime()  
print(b,type(b)) *#得到time.struct\_time(tm\_year=2019,tm\_mon=1,*

*tm\_mday=16,tm\_hour=19,tm\_min=8,tm\_sec=10,tm\_wday=2,tm\_yday=16,tm\_isdst=0)<class'time.struct\_time'>*

**③ gmtime(ts)：将时间戳数据转换为utc时间元祖对象返回 数据类型：time.struct\_time类型**utc = time.gmtime()  
print(utc,type(utc)) *#time.struct\_time(tm\_year=2019, tm\_mon=1, tm\_mday=16, tm\_hour=11, tm\_min=24, tm\_sec=9, tm\_wday=2, tm\_yday=16, tm\_isdst=0) <class 'time.struct\_time'>*   
  
time.sleep(2)  
  
ts = time.time()  
print(time.gmtime(ts)) *#time.struct\_time(tm\_year=2019, tm\_mon=1, tm\_mday=16, tm\_hour=11, tm\_min=24, tm\_sec=11, tm\_wday=2, tm\_yday=16, tm\_isdst=0)*

**④ mktime(tp)： 将本地元祖对象转换为时间戳对象(逆过程)：**

**【注意：】 mktime(tp)函数被调用执行，必须要传入一个实际参数，参数类型为time.struct\_time类型对象，返回值不会保留小数点后7位，只有一个(默认为0)**

tp = (2019,1,5,16,35,45,5,5,0)  
  
ts = time.mktime(tp)

print(ts,type(ts)) *#得到 1546677345.0 <class 'float'>*

**⑤ ctime(ts)： 将时间戳对象转换为字符串数据**

ts = time.time()str1 = time.ctime(ts)  
print(str1,type(str1))  *#Wed Jan 16 19:43:35 2019 <class 'str'>*

**⑥ asctime(tp)： 将本地元祖对象转化为字符串数据**

tp = (2019,1,5,16,35,45,5,5,0)  
str2 = time.asctime(tp)  
print(str2,type(str2))  *#Sat Jan 5 16:35:45 2019 <class 'str'>*

**将本地时间对象对象转换为字符串数据(最终以自定义样式显示输出)  
  
字符符号：  
%Y： 四位的年 相当于：yyyy  
%y： 两位的年 相当于：yy  
%m： 月份  
%d： 天数  
%H： 小时  
%M： 分钟  
%S： 秒钟  
  
补充扩展：  
%h： 月份(英文简写)  
%D: 格式：月/日/年(两位)  
%X： 格式：小时:分钟:秒钟**

**⑦strftime(format,tp)：将本地元祖对象以format格式化为str类型的数据返回**

**import** time

tp = (2019,1,5,16,35,45,5,5,0)  
str3 = time.strftime(**'%Y-%m-%d %H:%M:%S'**,tp)  
str4 = time.strftime(**'%Y/%m/%d %H:%M:%S'**,tp)  
str5 = time.strftime(**'%y/%m/%d %H:%M:%S'**,tp)  
str6 = time.strftime(**'%y/%h/%d %H:%M:%S'**,tp)  
str7 = time.strftime(**'%D %X'**,tp)

print(str3,type(str3)) *# 2019-01-05 16:35:45 <class 'str'>*  
print(str4,type(str4)) *# 2019/01/05 16:35:45 <class 'str'>*  
print(str5,type(str5)) *# 19/01/05 16:35:45 <class 'str'>*  
print(str6,type(str6)) *# 19/Jan/05 16:35:45 <class 'str'>*  
print(str7,type(str7)) *# 01/05/19 16:35:45 <class 'str'>*

**以下代码存在问题的，报错：UnicodeEncodeError 原因 --> 格式内容存在中文(非法字符)***# str3 = time.strftime('%Y"年"%h"月"%d"日" %H:%M:%S',tp)  
# print(str3,type(str3))*

**⑧ strptime(string,format)：将字符串数据以format格式化为本地元祖对象返回**

str3 = 2019-01-05 16:35:45tp1 = time.strptime(str3,**'%Y-%m-%d %H:%M:%S'**)  
  
print(tp1,type(tp1)) *#得到time.struct\_time(tm\_year=2019, tm\_mon=1, tm\_mday=5, tm\_hour=16, tm\_min=35, tm\_sec=45, tm\_wday=5, tm\_yday=5, tm\_isdst=-1) <class 'time.struct\_time'>*

**以下代码报错了：ValueError  
原因：解析字符串数据时的(format)必须和字符串格式化时的format保持一致,否则报错...***# tp1 = time.strptime(str3,'%Y/%m/%d %H:%M:%S')  
# print(tp1,type(tp1))*

**#clock()：打点 返回一个科学计数法得到的浮点数据(非常小)**c1 = time.clock()  
print(c1)  
time.sleep(2)  
c2 = time.clock()  
print(c2)  
time.sleep(2)  
c3 = time.clock()  
print(c3)

*#得到 8.1042498280886e-07*

*1.999508882460418*

*3.999531979572428*

**使用time模块的clock()或者time()函数实现性能测试**

**import** time*#打一个起始点*start = time.time() *#或者 start = time.clock()*  
*#模拟执行一段逻辑功能*i = 0  
sum = 0  
**while** i < 100000000:  
 sum += i  
 i += 1  
  
*#打一个结束点*end = time.time() *#或者 end = time.clock()*  
print(**'一共花费：%s时间'** %(end-start))

**2.datetime模块：**

**演示datetime模块的使用：  
理解datetime模块是time模块的补充/扩展  
datetime模块中有一些常用类：  
1).datetime类：记录了日期和时间数据  
2).date类： 记录了日期数据  
3).time类： 记录了时间数据**

**import** datetime

**#datetime类：****① now() 和② today()：获取当前的日期时间对象(返回值是一个datetime类型的数据)  
③ utcnow()：获取当前的日期时间对象(utc时间，数据类型为datetime)**

dt = datetime.datetime.now()  
print(dt,type(dt))

dt1 = datetime.datetime.today()  
print(dt1,type(dt1))  
  
dt2 = datetime.datetime.utcnow()  
print(dt2,type(dt2))

*① 2019-01-16 19:52:44.402614 <class 'datetime.datetime'>*

*② 2019-01-16 19:52:44.402615 <class 'datetime.datetime'>*

*③ 2019-01-16 11:52:44.402614 <class 'datetime.datetime'>*

**④获取指定的日期和时间对象(类型：datetime)（就是获取某一个自己想要的时间点）  
#构造函数：datetime(year,month,day,[hour,minute,seconds,num])**

dt3 = datetime.datetime(2018,12,12,11,11,11,345678)  
print(dt3,type(dt3))

*#得到2018-12-12 11:11:11.345678 <class 'datetime.datetime'>*

**⑤ strftime(fmt)：将datetime类型的数据转换为指定格式的字符串数据**

dt3 = datetime.datetime(2018,12,12,11,11,11,345678)str1 = dt3.strftime(**'%Y-%m-%d %X'**)  
print(str1,type(str1))  *#2018-12-12 11:11:11 <class 'str'>*

**从日期时间对象中分离出日期对象和时间对象  
date()：得到date类型对象  
time()：得到time类型对象**

print(dt3.date(),type(dt3.date()))   
print(dt3.time(),type(dt3.time()))

*#得到 2018-12-12 <class 'datetime.date'>*

*11:11:11.345678 <class 'datetime.time'>*

**⑥ timestamp() 和 timetuple()从日期时间对象中得到对应的 时间戳对象，（日期时间 -> 时间元祖）**ts = dt3.timestamp()  
print(ts,type(ts)) *#得到1544584271.345678 <class 'float'>*

print(dt3.timetuple(),type(dt3.timetuple()))

*#得到time.struct\_time(tm\_year=2018, tm\_mon=12,*

*tm\_mday=12, tm\_hour=11, tm\_min=11, tm\_sec=11,*

*tm\_wday=2, tm\_yday=346, tm\_isdst=-1) <class 'time.struct\_time'>*

**⑦操作属性：尝试获取日期时间对象中的年、月、日、时、分、秒、小数点后的数据  
【注意】：此操作不是针对函数层面，而是属性层面的操作**

print(dt3.year,dt3.month,dt3.day)  
print(dt3.hour,dt3.minute,dt3.second,dt3.microsecond)

*#得到 2018 12 12*

*11 11 11 345678*

**⑧ fromtimestamp(ts)：将时间戳对象转换为日期对象**

**import** datetime  
ts=1544584271.345678d = datetime.date.fromtimestamp(ts)  
print(d,type(d))  *#2018-12-12 <class 'datetime.date'>*

**⑨ timedelta()：得到一个timedelta类型的对象，可以和datetime对象做运算(+和-)；**delta = datetime.timedelta(days=1,hours=1,minutes=1,seconds=1)

print(delta,type(delta)) *#1day,1:01:01<class'datetime.timedelta'>*  
  
*#计算时间差*obj = dt3 - delta  
print(obj,type(obj)) print(dt3 + delta)

*#2018-12-11 10:10:10.345678 <class 'datetime.datetime'>*

*#2018-12-13 12:12:12.345678*

**3.calendar模块：**

**演示calendar(日历)：模块的使用**

**import** calendar  
 **① 构造：calendar(year,[w=2,l=1,c=6])：返回year年的完成的日历信息对象 数据类型str**c = calendar.calendar(2018,w=2,l=1,c=6)  
print(c)  
print(type(c))

**关于闰年的一些函数：  
② isleap(year)：判断year是否是闰年；如果返回True,是闰年；返回False，不是闰年  
③ leapdays(y1,y2)：返回[y1,y2)之间的闰年个数**print(calendar.isleap(2004))  *#True*print(calendar.leapdays(2000,2020))  *#5*

**#和month有关的一些函数：  
④ month(year,month)：返回year年month月的日历信息对象 数据类型：str**print(calendar.month(2019,1))  
print(type(calendar.month(2019,1)))

**⑤monthrange(year,month)：返回一个元祖对象，一共有两个元素，第一个元素记录了1号是当前周的第几天，第二个元素记录了是本月一共有多少天**print(calendar.monthrange(2019,1)) *#(1, 31)*  
print(calendar.monthrange(2019,2)) *#(4, 28*)  
**⑥timegm(tp)：将时间元祖对象转换为时间戳对象 数据类型：int**ts = calendar.timegm((2019,1,16,16,13,43))  
print(ts,type(ts)) *#1547655223 <class 'int'>*