day006

3.格式化的字符串
%a 本地（locale）简化星期名称
%A 本地完整星期名称
%b 本地简化月份名称
%B 本地完整月份名称
%c 本地相应的日期和时间表示
%d 一个月中的第几天（01-31）
%H 一天中的第几个小时（24小时制，00-23）
%I 第几个小时（12小时制，01-12）
%j 一年中的第几天（001-366）
%m 月份（01-12）
%M 分钟数（00-59）
%p 本地am或者pm的相应符
%s 秒（01-61）
%U 一年中的星期数。（00-53星期天是一个星期的开始。）第一个星期天之前的所有天数都放在第0周
%w 一个星期中的第几天（0-6,0是星期天）
%W 和%U基本相同，不同的是%W以星期一为一个星期的开始
%x 本地相应日期
%X 本地相应时间
%y 去掉世纪的年份（00-99）
%Y 完整的年份
%Z 时区的名字（如果不存在空字符）
%% '%'字符

2.元组
一种Python的数据结构表示，这个元组有9个整型内容
year,month,day,hours,minutes,seconds,weekday,Julia day,flag(1或-1或0)

1.时间戳
以整型或浮点型表示时间的一个以秒为单位的时间间隔，这个时间间隔的基础值是从1970年1月1日零点开始算起

UTC(世界协调时间）：格林尼治时间，世界标准时间，在中国来说是UTC+8
DST（夏令时）：是一种节约能源而认为规定时间制度，在夏季调快1个小时

1-递归

- 递归调用：一个函数，调用了自身，称为递归函数
- 递归函数：一个会调用自身的函数称为递归函数
- 凡是循环可以，做到递归也可以

方式
- 1.写出临界条件
- 2.找这一次和上一次的关系
- 3.假设当前函数已经能用，调用自身计算上一次的结果，再求出本次的结果

2-栈

stack 先进后出表
- stack.append("A") #入栈
- stack.pop() #出栈

3-队列

queue 先进先出表
- import collections
- queue = collections.deque() #创建一个队列
- queue = append("A") #入队
- res = popleft() #出队

练习：递归，栈，广度遍历文件夹

7-time

1.时间戳
2.元组
3.格式化的字符串

示例

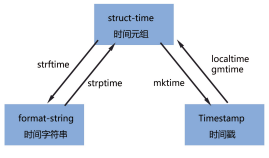
```
c=time.time()#返回当前时间的时间戳，不需要参数
t=time.gmtime(c)#将时间戳转为时间元组(格林尼治时间)
b=time.localtime(c)#将时间戳转为本地时间元组
p=time.ctime(c)#将时间戳转化为字符串
d=time.mktime(b)#将本地时间元组转为时间戳
s=time.asctime()#将时间元组转成字符串(现在的时间)
q=time.strftime("%Y-%m-%d %X",b)#将时间元组转换成给定格式的字符串，参数2为时间元组，如果没有参数2，默认转当前时间
w = time.strptime(q,"%Y-%m-%d %X")#将时间字符串转为时间元组
#可以做性能测试
y1 = time.perf_counter()
print(y1)
time.sleep(2)
y2 = time.perf_counter()
print(y2)
```

8-datetime

datetime比time高级，可以理解为datetime基于time进行了封装，提供了更为实用的函数
datetime模块的接口更直观，更容易调用。模块中的类：

datetime	同时有时间和日期
timedelta	主要用于计算时间的跨度
tzinfo	时区相关
time	只关注时间
date	只关注日期

```
import datetime
d1 = datetime.datetime.now()#获取当前时间
d2 = datetime.datetime(1999,10,1,10,28,15,12345)#获取指定时间
d3 = d1.strftime("%y-%m-%d %X")##将时间转为字符串
d4 = datetime.datetime.strptime(d3,"%y-%m-%d %X")#将格式化字符串转为datetime对象
d5 = datetime.datetime(1999,10,1,10,28,25,12345)
d6 = datetime.datetime.now()
d7 = d6-d5
print(d7.days)#间隔的天数
```

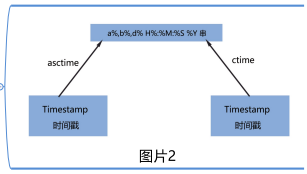


图片1

9-calendar

概要

```
import calendar
print(calendar.calendar(2017))#返回指定年的日历
print(calendar.month(2019,7))#返回指定某年某月日历
print(calendar.isleap(2000))#闰年返回True，否则返回False
print(calendar.monthrange(2019,7))#返回某年某月的weekday的第一天和天数
print(calendar.monthcalendar(2019,6))#返回某个月以每一周为元素的列表
```



图片2