nsd1909-py02-day01

time模块

时间表示方法：

- 时间戳：自1970-1-1 0:00:00到某一时间点之间的秒数
- UTC时间：世界协调时。以英国格林威治这个城市所在的经度为基准，向东向西每15度角划分一个时区，全球共24个时区。
- struct_time：九元组时间。年、月、日、时、分、秒、一周中的第几天、一年中的第几天、是否使用夏季节约时间

time模块的方法

https://docs.python.org/zh-cn/3/library/time.html

```
>>> import time
>>> time.time()      # 返回时间戳
1581730776.0589788
>>> time.ctime()     # 返回当前的时间字符串
'Sat Feb 15 09:40:52 2020'
>>> time.ctime(0)    # 时间戳为0秒时的时间字符串
'Thu Jan  1 08:00:00 1970'
>>> time.localtime()  # 反回当前时间的struct_time 9元组
time.struct_time(tm_year=2020, tm_mon=2, tm_mday=15, tm_hour=9, tm_min=41, tm_sec=55,
tm_wday=5, tm_yday=46, tm_isdst=0)
>>> t = time.localtime()
>>> t.tm_year
2020
>>> t.tm_yday
46
>>> t.tm_wday
5

>>> time.sleep(3)   # 睡眠3秒
>>> time.strftime('%Y-%m-%d %a %H:%M:%S')  # 返回指定格式的字符串
'2020-02-15 Sat 10:18:32'
# 将时间字符串转为9元组时间格式后，可以比较时间大小
>>> time.strptime('2020-2-14 12:00:00', '%Y-%m-%d %H:%M:%S')
time.struct_time(tm_year=2020, tm_mon=2, tm_mday=14, tm_hour=12, tm_min=0, tm_sec=0,
tm_wday=4, tm_yday=45, tm_isdst=-1)
>>> t1 = time.strptime('2020-2-14 12:00:00', '%Y-%m-%d %H:%M:%S')
>>> t2 = time.strptime('2020-1-14 12:00:00', '%Y-%m-%d %H:%M:%S')
>>> t1 > t2
True
```

datetime模块

```
>>> import datetime
>>> datetime.datetime.now()
datetime.datetime(2020, 2, 15, 10, 54, 16, 416819)
# 以上写法有些冗长，可以使用下面的方式替代
>>> from datetime import datetime
>>> t = datetime.now()  # 返回当前时间的年月日时分秒
>>> t.year, t.month, t.day, t.hour, t.minute, t.second
(2020, 2, 15, 10, 55, 41)
>>> t.strftime('%Y-%m-%d %H:%M:%S')  # 转成字符串形式
'2020-02-15 10:55:41'
# 将时间字符串，转换成 datetime对象
>>> t = datetime.strptime('2020-02-15 10:55:41', '%Y-%m-%d %H:%M:%S')
>>> t
datetime.datetime(2020, 2, 15, 10, 55, 41)

# 得到100天1小时之前的时间， 100天1小时之后的时间
>>> from datetime import datetime, timedelta
>>> days = timedelta(days=100, hours=1)
>>> t = datetime.now()
>>> t + days
datetime.datetime(2020, 5, 25, 12, 24, 15, 407633)
>>> t - days
datetime.datetime(2019, 11, 7, 10, 24, 15, 407633)
```

异常

- 异常就是不正常，当程序遇到各种各样错误时，程序将会崩溃，终止执行。
- 异常处理就是提前想到程序可能出现的错误，并给出解决方案。使得程序可以继续进行下去，不要再崩溃
- 异常处理语法格式：

```
try:
    有可能发生异常的代码
except 异常1:
    发生异常1时要执行的代码
except 异常2:
    发生异常2时要执行的代码
... ...
except 异常n:
    发生异常n时要执行的代码
else:
    不发生异常才执行的代码
finally:
    不管异常是否发生，一定会执行的语速句
```

触发异常

- 通过raise关键字，抛出异常
- 通过assert关键字，抛出AssertionError，即断言异常

os模块

- os模块是python访问文件系统主要采用的模块

```
>>> import os
>>> os.getcwd()    # pwd
'/var/ftp/nsd2019/nsd1909/py02/day01'
>>> os.listdir()   # ls
>>> os.listdir('/tmp')   # ls /tmp
>>> os.mkdir('/tmp/nsd1909')   # mkdir /tmp/nsd1909
>>> os.makedirs('/tmp/aaa/bbb/cccc')   # mkdir -p /tmp/aaa/bbb/cccc
>>> os.chdir('/tmp/nsd1909')   # cd /tmp/nsd1909
>>> os.getcwd()
'/tmp/nsd1909'

>>> os.mknod('hi.txt')   # touch hi.txt
>>> os.listdir()
['hi.txt']
>>> os.symlink('/etc/hosts', 'zhuji')  # ln -s /etc/hosts zhuji
>>> os.remove('hi.txt')  # rm -f hi.txt
>>> os.listdir()
['zhuji']
>>> os.mknod('hello.txt')
[root@localhost day01]# ll /tmp/nsd1909/hello.txt
-rw------- 1 root root 0 2月  15 13:45 /tmp/nsd1909/hello.txt
>>> os.chmod('hello.txt', 755)
[root@localhost day01]# ll /tmp/nsd1909/hello.txt
--wxrw--wt 1 root root 0 2月  15 13:45 /tmp/nsd1909/hello.txt
>>> os.chmod('hello.txt', 0o755)  # 注意，linux权限是8进制数
[root@localhost day01]# ll /tmp/nsd1909/hello.txt
-rwxr-xr-x 1 root root 0 2月  15 13:45 /tmp/nsd1909/hello.txt


>>> os.makedirs('aaa/bbb')
>>> os.mkdir('abc')
>>> import shutil
>>> shutil.copy('/etc/hosts', 'abc')
'abc/hosts'
>>> shutil.copy('/etc/passwd', 'abc')
'abc/passwd'
>>> os.mkdir('aaa/ccc')
>>> os.mknod('aaa/a.txt')
>>> os.mknod('aaa/b.txt')
>>> os.mknod('aaa/bbb/b1.txt')
>>> os.mknod('aaa/bbb/b2.txt')
>>> os.mknod('aaa/ccc/c1.txt')
>>> os.mknod('aaa/ccc/c2.txt')
[root@localhost nsd1909]# ls -R
.:
aaa  abc  hello.txt  zhuji

./aaa:
a.txt  bbb  b.txt  ccc

./aaa/bbb:
b1.txt  b2.txt

./aaa/ccc:
c1.txt  c2.txt
```

```
./abc:
hosts  passwd
```

os.walk

```
>>> mulu = list(os.walk('/tmp/nsd1909'))
>>> mulu
[('/tmp/nsd1909', ['aaa', 'abc'], ['zhuji', 'hello.txt']), ('/tmp/nsd1909/aaa', ['bbb',
'ccc'], ['a.txt', 'b.txt']), ('/tmp/nsd1909/aaa/bbb', [], ['b1.txt', 'b2.txt']),
('/tmp/nsd1909/aaa/ccc', [], ['c1.txt', 'c2.txt']), ('/tmp/nsd1909/abc', [], ['hosts',
'passwd'])]
>>> len(mulu)
5
>>> mulu[0]
('/tmp/nsd1909', ['aaa', 'abc'], ['zhuji', 'hello.txt'])
>>> mulu[1]
('/tmp/nsd1909/aaa', ['bbb', 'ccc'], ['a.txt', 'b.txt'])
>>> mulu[2]
('/tmp/nsd1909/aaa/bbb', [], ['b1.txt', 'b2.txt'])
>>> mulu[3]
('/tmp/nsd1909/aaa/ccc', [], ['c1.txt', 'c2.txt'])
>>> mulu[4]
('/tmp/nsd1909/abc', [], ['hosts', 'passwd'])
# 经分析，lumu列表由5个元组构成。每个元组拥有一样的结构。
# 每个元组有三项内容：(字符串，列表1，列表2)
# 字符串：目录路径
# 列表1：目录路径下的子目录
# 列表2：目录路径下的文件
```

os.path

```
>>> os.path.basename('/tmp/nsd2019/abc.txt')
'abc.txt'
>>> os.path.dirname('/tmp/nsd2019/abc.txt')
'/tmp/nsd2019'
>>> os.path.split('/tmp/nsd2019/abc.txt')   # 切割路径
('/tmp/nsd2019', 'abc.txt')
>>> os.path.join('/tmp/nsd2019', 'abc.txt')  # 路径拼接
'/tmp/nsd2019/abc.txt'
>>> os.path.isfile('/etc/hosts')  # 文件存在并且是文件吗？
True
>>> os.path.isdir('/etc/hosts')   # 存在并且是目录吗？
False
>>> os.path.islink('/etc/grub2.cfg')   # 存在并且是链接文件吗？
True
>>> os.path.ismount('/')    # 存在并且是挂载点吗？
True
>>> os.path.exists('/etc')  # 存在吗？
True
```

pickle模块

- 文件默认只能写入str或bytes类型的数据
- pickle可以将任意类型的数据写入文件，还可以无损地取出来。

```
>>> import pickle
>>> gouwu = ['方便面', '速冻水饺', '苹果']
>>> with open('/tmp/shop.data', 'wb') as fobj:
...     pickle.dump(gouwu, fobj)  # 将列表存入文件

>>> with open('/tmp/shop.data', 'rb') as fobj:
...   l = pickle.load(fobj)   # 从文件中取出数据
...
>>> l
['方便面', '速冻水饺', '苹果']
>>> type(l)
<class 'list'>
```

作业：记账程序

1. 发呆。思考程序的运行方式

```
(0) 收入
(1) 开销
(2) 查询
(3) 退出
请选择(0/1/2/3): 0
金额: 10000
备注: salary
(0) 收入
(1) 开销
(2) 查询
(3) 退出
请选择(0/1/2/3): 1
金额: 100
备注: buy clothes
(0) 收入
(1) 开销
(2) 查询
(3) 退出
请选择(0/1/2/3): 2
date         save          cost       balance       comment
2020-2-15    0             0          10000         init data
2020-2-15    10000         0          20000         salary
2020-2-15    0             100        19900         buy clothes
(0) 收入
(1) 开销
(2) 查询
(3) 退出
请选择(0/1/2/3): 3
Bye-bye
```

2. 数据存储

```
[
    ['2020-2-15', 0, 0, 10000, 'init data'],
    ['2020-2-15', 10000, 0, 20000, 'salary'],
    ['2020-2-15', 0, 100, 19900, 'buy clothes'],
]

>>> data = [['2020-2-15', 0, 0, 10000, 'init data']]
>>> data[-1]   # 取出大列表的最后一项，取出的又是一个小列表
['2020-2-15', 0, 0, 10000, 'init data']
>>> data[-1][-2]  # 继续取出小列表的倒数第二项
10000
>>> data[-1][-2] + 10000
20000
>>> line = ['2020-2-15', 10000, 0, 20000, 'salary']
>>> data.append(line)
>>> data
[['2020-2-15', 0, 0, 10000, 'init data'], ['2020-2-15', 10000, 0, 20000, 'salary']]
>>> data[-1][-2]
20000
```