3.14 计算当前月份的日期范围¶

问题¶

你的代码需要在当前月份中循环每一天，想找到一个计算这个日期范围的高效方法。

解决方案¶

在这样的日期上循环并需要事先构造一个包含所有日期的列表。 你可以先计算出开始日期和结束日期， 然后在你步进的时候使用 `datetime.timedelta` 对象递增这个日期变量即可。

下面是一个接受任意 `datetime` 对象并返回一个由当前月份开始日和下个月开始日组成的元组对象。

```
from datetime import datetime, date, timedelta
import calendar

def get_month_range(start_date=None):
    if start_date is None:
        start_date = date.today().replace(day=1)
    _, days_in_month = calendar.monthrange(start_date.year, start_date.month)
    end_date = start_date + timedelta(days=days_in_month)
    return (start_date, end_date)
```

有了这个就可以很容易的在返回的日期范围上面做循环操作了：

```
>>> a_day = timedelta(days=1)
>>> first_day, last_day = get_month_range()
>>> while first_day < last_day:
...     print(first_day)
...     first_day += a_day
...
2012-08-01
2012-08-02
2012-08-03
2012-08-04
2012-08-05
2012-08-06
2012-08-07
2012-08-08
2012-08-09
#... and so on...
```

讨论¶

上面的代码先计算出一个对应月份第一天的日期。 一个快速的方法就是使用 `date` 或 `datetime` 对象的 `replace()` 方法简单的将 `days` 属性设置成1即可。 `replace()` 方法一个好处就是它会创建和你开始传入对象类型相同的对象。 所以，如果输入参数是一个 `date` 实例，那么结果也是一个 `date` 实例。 同样的，如果输入是一个 `datetime` 实例，那么你得到的就是一个 `datetime` 实例。

然后，使用 `calendar.monthrange()` 函数来找出该月的总天数。 任何时候只要你想获得日历信息，那么 `calendar` 模块就非常有用了。 `monthrange()` 函数会返回包含星期和该月天数的元组。

一旦该月的天数已知了，那么结束日期就可以通过在开始日期上面加上这个天数获得。 有个需要注意的是结束日期并不包含在这个日期范围内(事实上它是下个月的开始日期)。 这个和Python的 `slice` 与 `range` 操作行为保持一致，同样也不包含结尾。

为了在日期范围上循环，要使用到标准的数学和比较操作。 比如，可以利用 `timedelta` 实例来递增日期，小于号<用来检查一个日期是否在结束日期之前。

理想情况下，如果能为日期迭代创建一个同内置的 `range()` 函数一样的函数就好了。 幸运的是，可以使用一个生成器来很容易的实现这个目标：

```
def date_range(start, stop, step):
    while start < stop:
        yield start
        start += step
```

下面是使用这个生成器的例子：

```
>>> for d in date_range(datetime(2012, 9, 1), datetime(2012,10,1),
                        timedelta(hours=6)):
...     print(d)
...
2012-09-01 00:00:00
2012-09-01 06:00:00
2012-09-01 12:00:00
2012-09-01 18:00:00
2012-09-02 00:00:00
2012-09-02 06:00:00
...
>>>
```

这种实现之所以这么简单，还得归功于Python中的日期和时间能够使用标准的数学和比较操作符来进行运算。