Classes and Objects in Python

Example: Dates

A date could be an object

With month, day, year as fields inside

Grouping relevant methods (functions)

In Python ...

class Date:
    """A date is made up of a month, a day, and a year."""
    def __init__(self, month, day, year):
        assert month >= 1 and month <= 12
        assert day >= 1 and day <= 31
        assert year >= 0
        self.month = month
        self.day = day
        self.year = year

date_1 = Date( 10, 14, 2011 )

Recall ...

global Current_Month
global Current_Day

def get_cur_mmdd() :
    global Current_Month  ## Yes, that's the one I mean
    global Current_Day    ## Not a local variable with the same name
    Current_Month = int(raw_input("Current month: "))
    Current_Day = int(raw_input("Current day: "))
    return

YUCK!

We did that because we didn't have a better way, yet.
Now we do.
Some methods ...

class Date:
...

def to_string_mmddyyyy(self):
    """Return date as string in mm/dd/yyyy format""
    return ( ("%2d" % self.month) +
    ("/%2d" % self.day) +
    ("/%4d" % self.year) )

def to_string_mmddyy(self):
    """Return date as string in mm/dd/yyyy format""
    yy = self.year % 100
    return ( ("%2d" % self.month) +
    ("/%2d" % self.day) +
    ("/%2d" % yy) )

A “factory” method for creating dates ...

class Date:
...

@classmethod
def parse_mmddyy(cls, mmddyy):
    """Create a new Date object from the mm/dd/yy formatted string.""
    fields = mmddyy.split( "/" )
    month = int( fields[0] )
    day = int( fields[1] )
    year = int( fields[2] )
    # If the year entered was less than 100, we'll interpret
    # it as a year between 1951 and 2050
    if year < 51:
        year = year + 2000
    elif year < 100:
        year = year + 1900
    return Date( month, day, year )

Fancier conversion ...

class Date:
...


def to_string_US(self):
    """Return date in US convention month day, year""
    return ( Date.Month_names[ self.month ] +
    ( "%2d" % self.day ) +
    ( ", %2d" % self.year ) )

date_1 = Date( 10, 14, 2011 )
print "US trad format: ", date_1.to_string_US()

Some things we could add ...

Add n days to date
Add (months, days) to date
Determine whether date1 > date2
Calculate days between date1 and date2 etc.

Let’s do date comparison ...