```
#TheSimpleDateClass.py

class SimpleDate(object):
    """
    Represents a
date.

    Attributes
       m : index of month [int]
       d : day  [int]
       y : year
[int]
    """
    # Class variable...
    nDays
=[0,31,28,31,30,31,30,31,31,30,31,30,31]

    def __init__(self,s):

""" Returns a SimpleDate representation of the date encoded in s.


  PreC: s is a date string of the form 'M/D/Y' where M, D and Y encode
        the month index,
the day and the year.
        """
        v = s.split('/')
        m =
int(v[0]); d = int(v[1]); y = int(v[2])
        self.m = m; self.d = d; self.y = y

    def
__str__(self):
        """ Pretty prints the date encoded in self. For example

      if D is a SimpleDate object that encodes 7/14/1776 then

print D
        displays July 4, 1776.
        """

TheMonths =['','January','February','March','April','May','June',

'July','August','September','October','November','December']
        return TheMonths[self.m] +
' ' + str(self.d) + ', ' + str(self.y)


    def __eq__(self,other):

""" Returns True if other encodes the same date as self
        and False
otherwise. Thus, if D1 and D2 are SimpleDate objects,
        then D1==D2 is True if and only
if the encode the same date.

        PreC: other is a SimpleDate object

"""
        return self.m==other.m and self.d==other.d and self.y==other.y



    def __sub__(self,other):
        """ Returns an int that is the number of
days that have
        elapsed from other to self. Thus, if D1 and D2 are SimpleDate objects

     and D2 encodes a later date, then the value of D2-D1 is an integer
        that is the
number of days from D1 to D2

        PreC: other are SimpleDate objects with the other
encoding a
```

```
        date that is before the date encoded in self.
        """

    k = 0
        Day = other
        while not (Day==self):
            k+=1
            Day =
Day.Tomorrow()
        return k

    def __add__(self,n):
        """
Returns a date that is n days after the date encoded in self. Thus
        if D is a SimpleDate
object a F = D + 5, then F is a SimpleDate object
        that encodes a date that is 5 days
later.

        PreC: n is a nonegative integer.
        """

Day = self
        for k in range(n):
            Day = Day.Tomorrow()
        return Day



    def Tomorrow(self):
        """ Returns a SimpleDate that encodes the
date of the day after
        the date encoded by self."""
        m = self.m

     d = self.d
        y = self.y
        # Assign the last day of the month to Last

Last = self.nDays[m]
        if self.isLeapYear() and m==2:
            Last+=1
        # Must
check if tomorrow is in the same month an year
        if d<Last:
            # Same Month

          d+=1
        else:
            # Different Month
            d=1
            if
m<12:
                # If it is not December...
                m+=1
            else:

           # It is December so tomorrow is New Years Day...
                m=1

y+=1
        return SimpleDate(str(m)+'/'+str(d)+'/'+str(y))


    def
isLeapYear(self):
        """ Returns True if y is a leap year. False otherwise


        PreC: y is a positive integer
        """
        y = self.y

return ((y%100>0) and y%4==0) or ((y%100==0) and (y%400==0))
```