Date.py

```python
#!/usr/bin/python
# Date.py

class Error(Exception):
    """The base class of erros"""
    # __code=0
    # constructor
    def __init__(self,  n):
        self.__code = n

    def getCode(self):
        return self.__code

class ErrorYear(Error):
    """Raised when the input value for year is negative"""
    # __num=0
    # constructor
    def __init__(self, n):
        Error.__init__(self, 1)
        self.__num = n
    def getNum(self):
        return self.__num

class ErrorDay(Error):
    """Raised when the input value for day is not between 1 to 31"""
    # __num=0
    # constructor
    def __init__(self, n):
        Error.__init__(self, 1)
        self.__num = n
    def getNum(self):
        return self.__num

class ErrorMonth(Error):
    """Raised when the input value for month is not between 1 to 12"""
    # __num=0
    # constructor
    def __init__(self, n):
```

```python
        Error.__init__(self, 1)
        self.__num = n
    def getNum(self):
        return self.__num


class Date:
    def __trace(self, s):
        print(s)
    def __init__(self, y, d, m):
        if y < 0:
            raise ErrorYear (y)
        if d < 1 and d > 31 :
            raise ErrorDay (d)
        if m < 1 and m > 12 :
            raise ErrorMonth (m)
        self.__year = y
        self.__day = d
        self.__month = m
    def __del__(self):
        pass
    def getYear(self):
        return self.__year
    def setYear(self, y):
        self.__year = y
    def getDay(self):
        return self.__day
    def setDay(self, d):
        self.__day = d
    def getMonth(self):
        return self.__month
    def setMonth(self, m):
        self.__month = m
    def isCentennial(self):
        return self.__year % 100 == 0
    def reset(self):
        self.__year=0
        self.__day=1
        self.__month=1
```

Date_test.py

```python
#!/usr/bin/python
# Date_test.py

import Date

if __name__ == '__main__':
  try:
    t1 = Date.Date(20111, 2, 3)
  except Date.ErrorYear as obj:
    print("Error: The Year", obj.getNum(), "is negative.")
  except Date.ErrorDay as obj:
    print("Error: The Day ", obj. getNum(), "is not between 1 to 31.")
  except Date.ErrorMonth as obj:
    print("Error: The Month ", obj. getNum(), "is not between 1 to 12.")
  except:
    print("Error: Unknwon exception.")
  else:
    print("No exception.")
  finally:
    print("End")
```
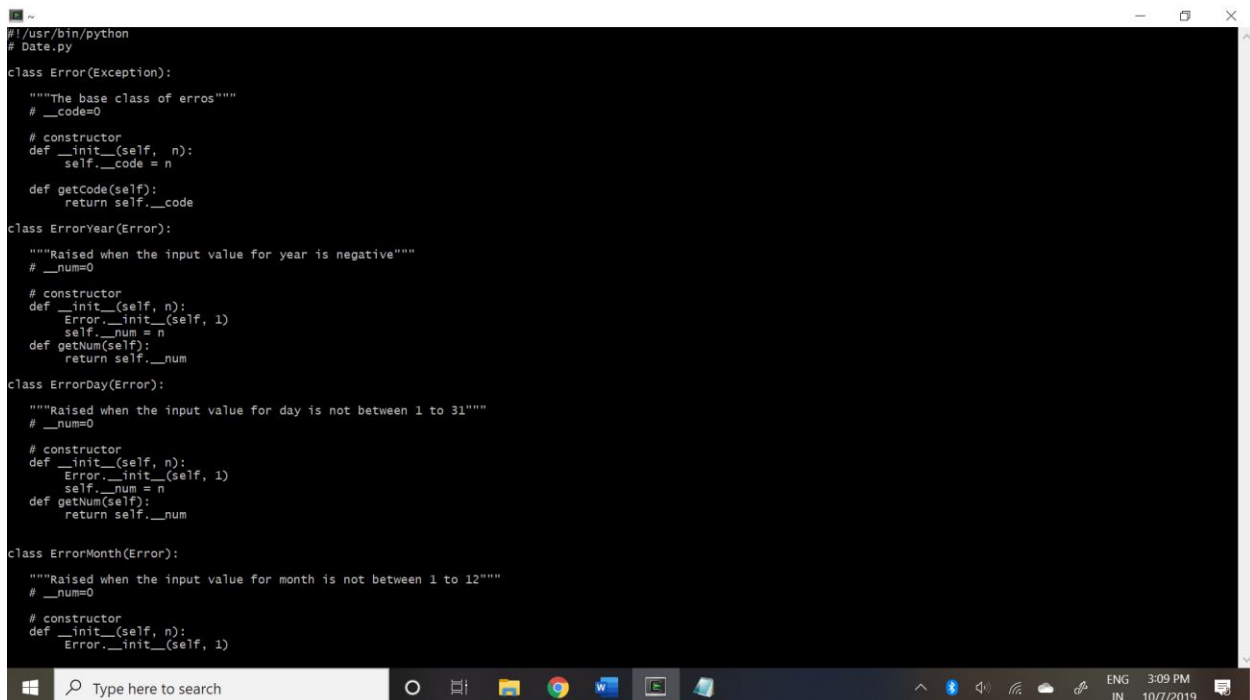


```
#!/usr/bin/python
# Date.py

class Error(Exception):

    """The base class of erros"""
    # __code=0

    # constructor
    def __init__(self, n):
        self.__code = n

    def getCode(self):
        return self.__code

class ErrorYear(Error):

    """Raised when the input value for year is negative"""
    # __num=0

    # constructor
    def __init__(self, n):
        Error.__init__(self, 1)
        self.__num = n
    def getNum(self):
        return self.__num

class ErrorDay(Error):

    """Raised when the input value for day is not between 1 to 31"""
    # __num=0

    # constructor
    def __init__(self, n):
        Error.__init__(self, 1)
        self.__num = n
    def getNum(self):
        return self.__num

class ErrorMonth(Error):

    """Raised when the input value for month is not between 1 to 12"""
    # __num=0

    # constructor
    def __init__(self, n):
        Error.__init__(self, 1)
```
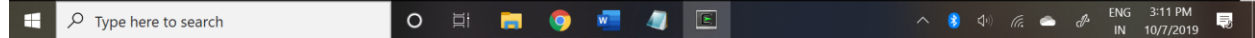
```python
            Error.__init__(self, 1)
            self.__num = n
    def getNum(self):
        return self.__num


class Date:


    def __trace(self, s):
        print(s)
    def __init__(self, y, d, m):
        if y < 0:
            raise ErrorYear (y)


        if d < 1 and d > 31 :
            raise ErrorDay (d)
        if m < 1 and m > 12 :
            raise ErrorMonth (m)
        self.__year = y
        self.__day = d
        self.__month = m
    def __del__(self):
        pass

    def getYear(self):
      return self.__year
    def setYear(self, y):
      self.__year = y

    def getDay(self):
      return self.__day
    def setDay(self, d):
      self.__day = d

    def getMonth(self):
      return self.__month
    def setMonth(self, m):
      self.__month = m

    def isCentennial(self):
      return self.__year % 100 == 0

    def reset(self):
      self.__year=0
      self.__day=1
      self.__month=1
```

```python
#!/usr/bin/python

# Date_test.py
import Date

if __name__ == '__main__':
    try:
        t1 = Date.Date(20111, 2, 3)

    except Date.ErrorYear as obj:
        print("Error: The Year", obj.getNum(), "is negative.")
    except Date.ErrorDay as obj:
        print("Error: The Day ", obj. getNum(), "is not between 1 to 31.")
    except Date.ErrorMonth as obj:
        print("Error: The Month ", obj. getNum(), "is not between 1 to 12.")
    except:
        print("Error: Unknwon exception.")
    else:
        print("No exception.")
    finally:
        print("End")
```

"Date_test.py" 38 lines, 576 characters

```
Mittal Patel@LAPTOP-NDH89GF7 ~
$ vi Date.py

Mittal Patel@LAPTOP-NDH89GF7 ~
$ vi Date_test.py

Mittal Patel@LAPTOP-NDH89GF7 ~
$ python Date_test.py
No exception.
End

Mittal Patel@LAPTOP-NDH89GF7 ~
$ |
```