

Working with dates and times

datetime

Christopher Harrison | Content Developer
Susan Ibach | Technical Evangelist

We spend a lot of time thinking about deadlines and schedules

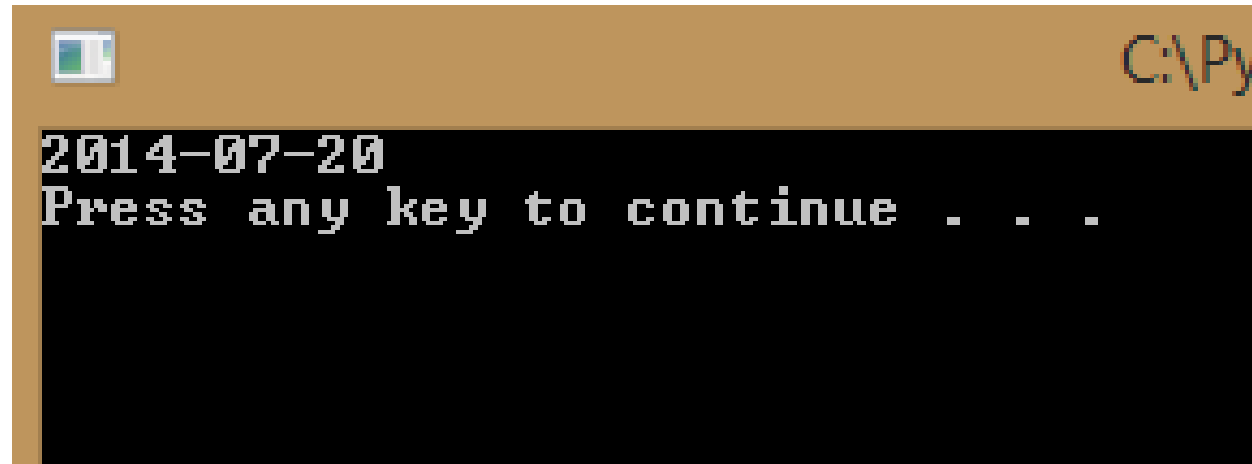
- How many days do I have until my birthday?
- When is my project due?
- I want to book an appointment in two weeks, what will the date be?

To solve these problems with computers we need to store and manipulate dates and times

If I want to know how many days until my birthday, first I need today's date

- The datetime class allows us to get the current date and time

```
#The import statement gives us access to  
#the functionality of the datetime class  
import datetime  
#today is a function that returns today's date  
print (datetime.date.today())
```



A screenshot of a Python terminal window. The window has a brown title bar with a small icon on the left and the text 'C:\Py' on the right. The terminal area has a black background with white text. The first line of output is '2014-07-20'. The second line is 'Press any key to continue . . .', followed by three dots.

```
2014-07-20  
Press any key to continue . . .
```

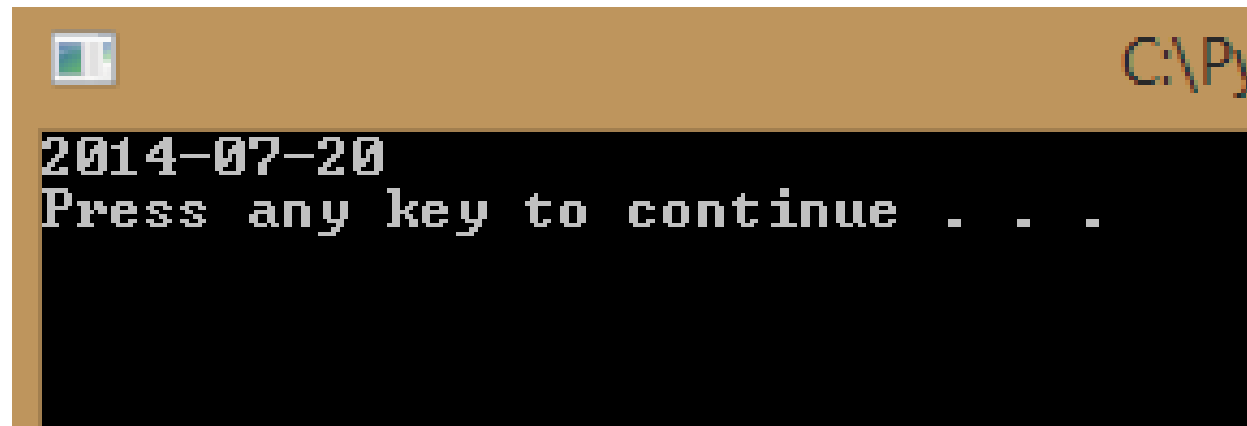
You can store dates in variables

```
import datetime
```

```
#store the value in a variable called currentDate
```

```
currentDate = datetime.date.today()
```

```
print (currentDate)
```



A screenshot of a Windows command prompt window. The title bar is brown and shows a small icon on the left and the path 'C:\P...' on the right. The command prompt itself has a black background with white text. It displays the date '2014-07-20' on the first line and 'Press any key to continue . . .' on the second line.

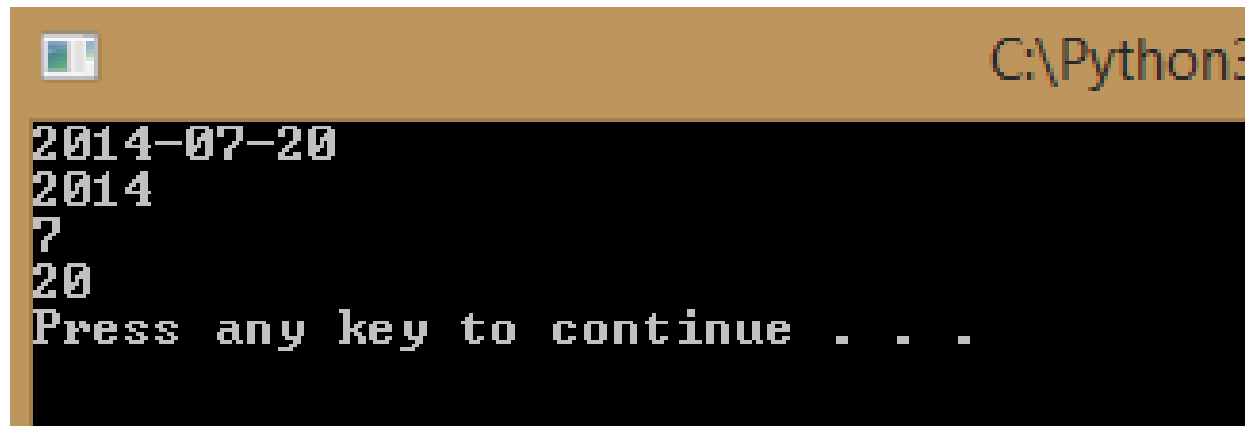
```
2014-07-20  
Press any key to continue . . .
```

DEMO

Displaying current date and time

You can access different parts of the date

```
import datetime
currentDate = datetime.date.today()
print (currentDate)
print (currentDate.year)
print (currentDate.month)
print (currentDate.day)
```



```
C:\Python34>python3
2014-07-20
2014
7
20
Press any key to continue . . .
```

DEMO

Using Date functions to access date parts

Date formats

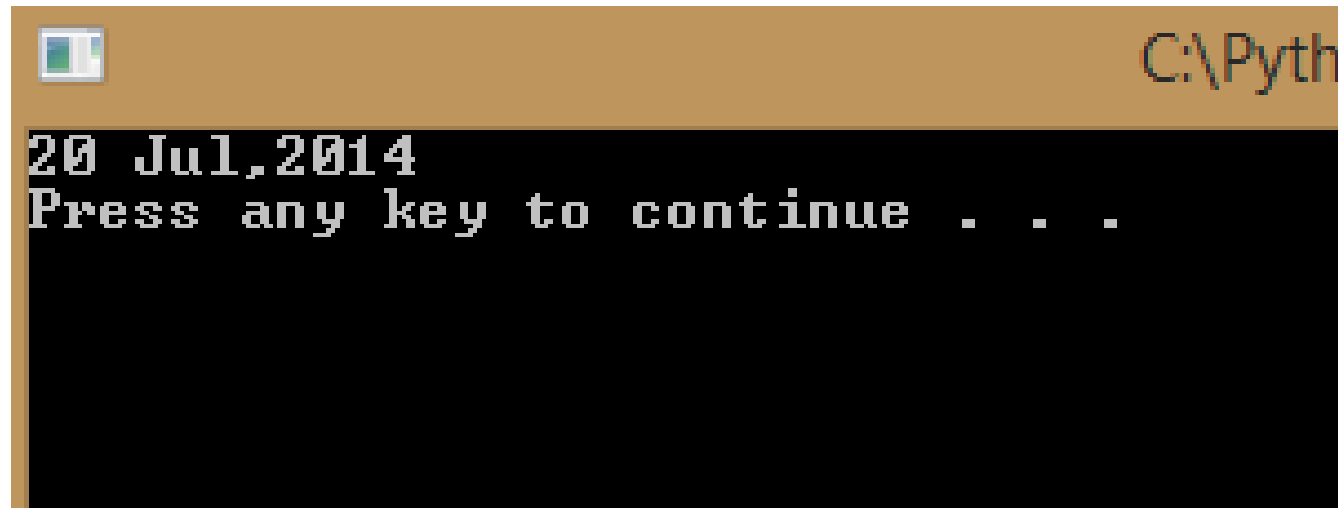
What date does 2/5/2014 represent?

But what if you want to display the date with a different format?

- Welcome to one of the things that drives programmers insane!
- Different countries and different users like different date formats, often the default isn't what you need
- There is always a way to handle it, but it will take a little time and extra code
- The default format is YYYY-MM-DD

In Python we use strftime to format dates

```
import datetime
currentDate = datetime.date.today()
#strftime allows you to specify the date format
print (currentDate.strftime('%d %b,%Y'))
```



A screenshot of a Python terminal window. The window has a brown title bar with a small icon on the left and the text 'C:\Pyth' on the right. The terminal area has a black background with white text. The first line of output is '20 Jul,2014'. The second line is 'Press any key to continue . . .'.

```
C:\Pyth
20 Jul,2014
Press any key to continue . . .
```

What the heck are %d %b and %Y?

- %d is the day of the month
- %b is the abbreviation for the month
- %Y is the 4 digit year

Here's a few more you may find useful

- %b is the month abbreviation
- %B is the full month name
- %y is two digit year
- %a is the day of the week abbreviated
- %A is the day of the week
- For a full list visit strftime.org

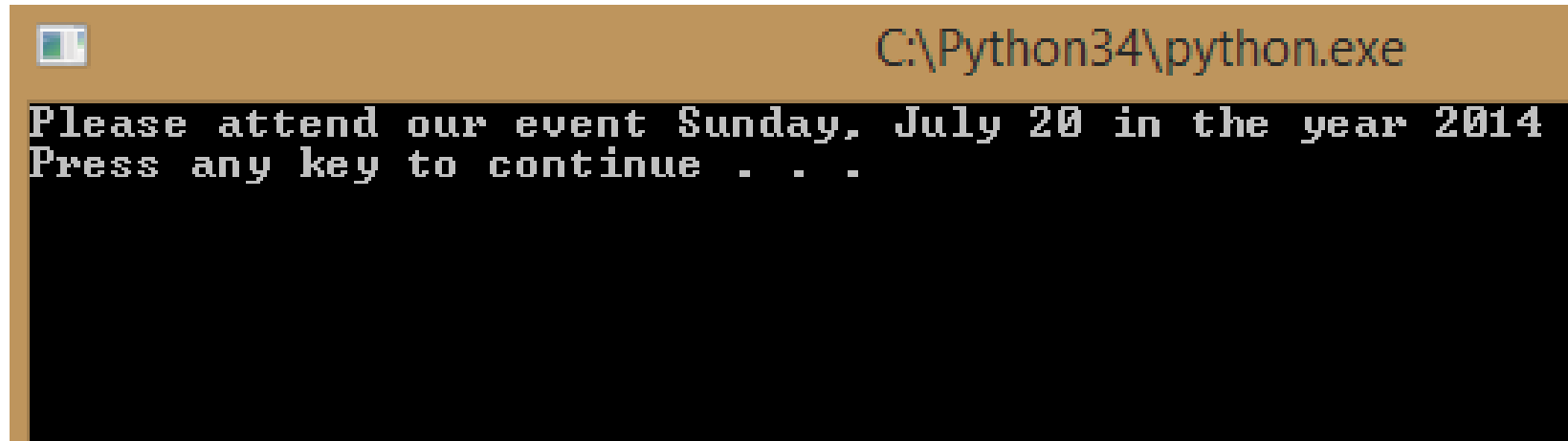
DEMO

Formatting dates

Could you print out a wedding invitation?

"Please attend our event Sunday, July 20 in the year 1997"

```
import datetime
currentDate = datetime.date.today()
#strftime allows you to specify the date format
print (currentDate.strftime
('Please attend our event %A, %B %d in the year %Y'))
```



```
C:\Python34\python.exe
Please attend our event Sunday, July 20 in the year 2014
Press any key to continue . . .
```


So... what if I don't want English?

- In programmer speak we call that localization
- Did I mention working with dates can be challenging?
- By default the program uses the language of the machine where it is running
- But... since you can't always rely on computer settings it is possible to force Python to use a particular language
- It just takes more time and more code. We won't go into that now, but if you need to do it check out the babel Python library <http://babel.pocoo.org/>

Let's get back to calculating days until your birthday, I need to ask your birthday.

```
birthday = input ("What is your birthday? ")  
print ("Your birthday is " + birthday)
```

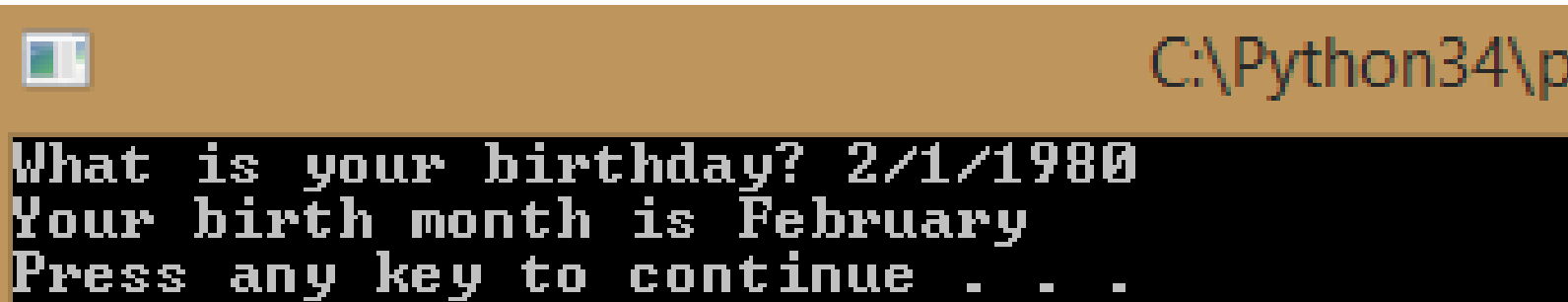
What datatype is birthday?

string

if we want to treat it like a date (for example use the datetime functions to print it in a particular format) we must convert it to a date

The strptime function allows you to convert a string to a date

```
import datetime
birthday = input ("What is your birthday? ")
birthdate =
datetime.datetime.strptime(birthday, "%m/%d/%Y").date()
#why did we list datetime twice?
#because we are calling the strptime function
#which is part of the datetime class
#which is in the datetime module
print ("Your birth month is " + birthdate.strftime('%B'))
```



```
C:\Python34\p
What is your birthday? 2/1/1980
Your birth month is February
Press any key to continue . . .
```

DEMO

Asking a user for a date value

But what if the user doesn't enter the date in the format I specify in strptime?

```
birthdate = datetime.datetime.strptime(birthday, "%m/%d/%Y")
```

- Your code will crash so...
- Tell the user the date format you want

```
birthday = input ("What is your birthday? (mm/dd/yyyy) ")
```

- Add error handling, which we will cover in a later module

Dates seem like a lot of hassle, is it worth it? Why not just store them as strings!

- You can create a countdown to say how many days until a big event or holiday

```
nextBirthday = \
    datetime.datetime.strptime('12/20/2014', '%m/%d/%Y').date()
currentDate = datetime.date.today()
#If you subtract two dates you get back the number of days
#between those dates
difference = nextBirthday - currentDate
print (difference.days)
```

Dates seem like a lot of hassle, is it worth it? Why not just store them as strings!

- You can tell someone when the milk in their fridge will expire

```
currentDate = datetime.date.today()
```

```
#timedelta allows you to specify the time
```

```
#to add or subtract from a date
```

```
print (currentDate + datetime.timedelta(days=15))
```

```
print (currentDate + datetime.timedelta(hours=15))
```

You will be amazed how often you need to work with dates!

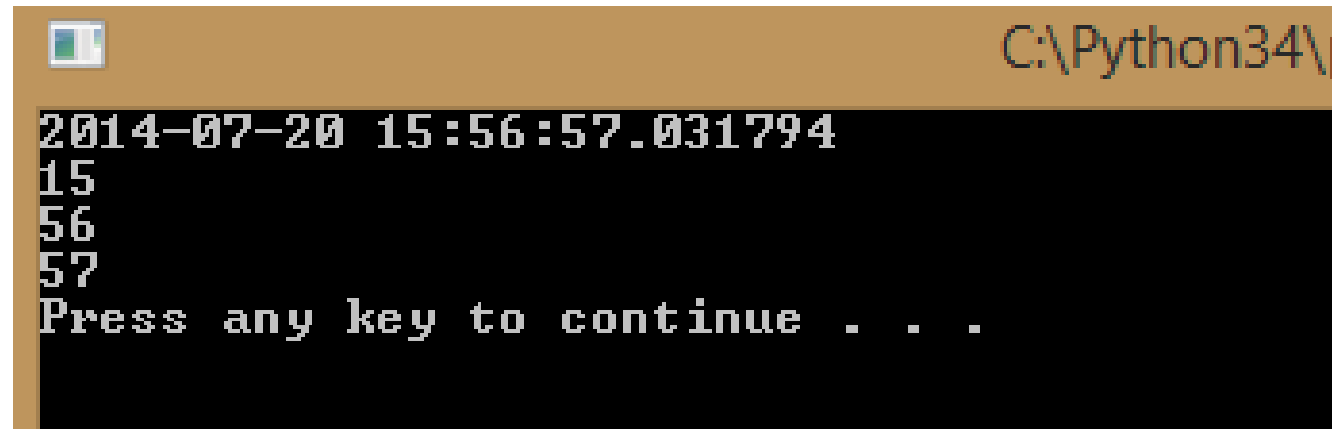
- If datetime doesn't have what you need, check out the [dateutil](#) library (for example you might want to know the number of years between two dates instead of number of days)

Working with time

What about times?

- It is called Datetime, so yes, it can store times.

```
import datetime
currentTime = datetime.datetime.now()
print (currentTime)
print (currentTime.hour)
print (currentTime.minute)
print (currentTime.second)
```



```
C:\Python34\
2014-07-20 15:56:57.031794
15
56
57
Press any key to continue . . .
```

Just like with dates you can use `strftime()` to format the way a time is displayed

```
import datetime
currentTime = datetime.datetime.now()
print (datetime.datetime.strftime(currentTime, '%H:%M'))
```

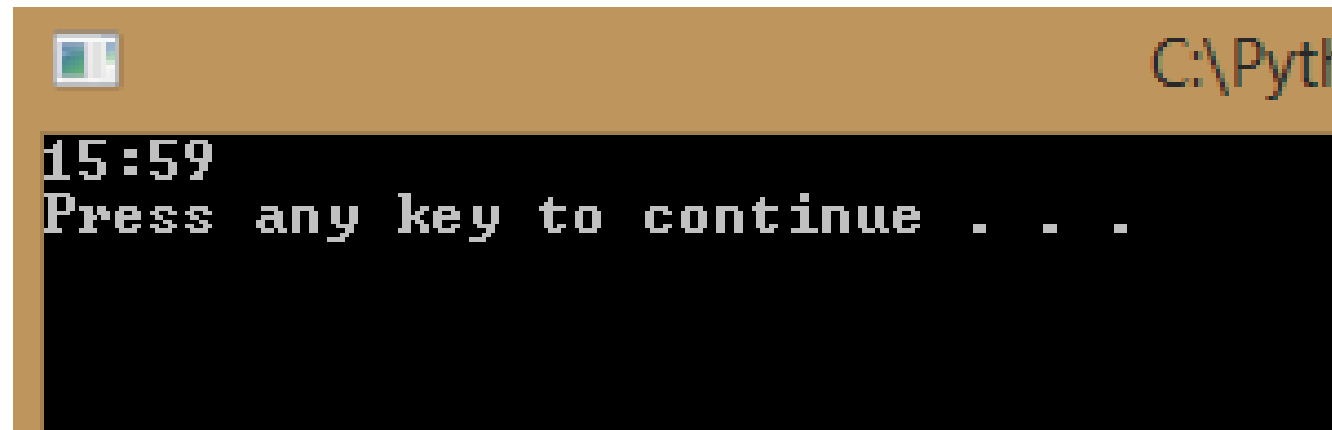
%H Hours (24 hr clock)

%I Hours (12 hr clock)

%p AM or PM

%m Minutes

%S Seconds

A screenshot of a Python terminal window. The title bar is brown and shows the file path 'C:\Pyth...'. The terminal has a black background with white text. It displays the output '15:59' and then prompts 'Press any key to continue . . .' with three dots.

```
C:\Pyth...
15:59
Press any key to continue . . .
```

DEMO

Working with times

Your challenge

- Ask a user to enter the deadline for their project
- Tell them how many days they have to complete the project
- For extra credit, give them the answer as a combination of weeks & days (Hint: you will need some of the math functions from the module on numeric values)

Congratulations!



- You can now format dates and times
- You can perform calculations with date values



Microsoft

©2013 Microsoft Corporation. All rights reserved. Microsoft, Windows, Office, Azure, System Center, Dynamics and other product names are or may be registered trademarks and/or trademarks in the U.S. and/or other countries. The information herein is for informational purposes only and represents the current view of Microsoft Corporation as of the date of this presentation. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information provided after the date of this presentation. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS PRESENTATION.