Python’s built in time, datetime modules

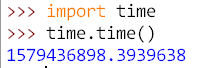
# Time Module

## General info

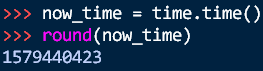
Python’s time module uses your computers system clock for the current time.

## Time.time() function

The unix epoch is a time reference commonly used in programming. The start point of this time is 12 AM January 1 1970. When you call time.time() it returns the number of seconds since that exact moment as a float. This number is called epoch timestamp.



We can use the round function to round this aswell.

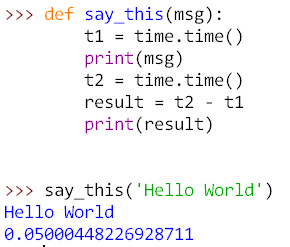


The return value is how many seconds have passed between the Unix epoch and the moment time.time() was called.

We can use this in a really useful way in our programming as measuring how long does it take to run our program. (As we used in the decorators information)

## Time.sleep()

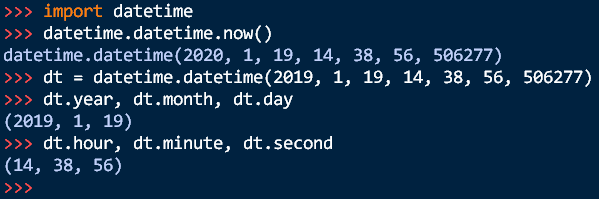
A simple function that makes the program wait a certain amount of seconds when its called.



# Datetime module

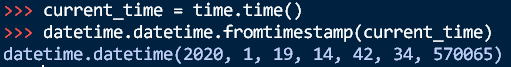
The time module is useful for getting a Unix epoch timestamp to work with. But if you want to display a date in a more convenient format or do calculation with dates you should use the datetime module (for eg. Figuring out what date was 205 days ago or what date is 123 days from now).

The datetime module has its own datetime data type, datetime values represent a specific moment in time.

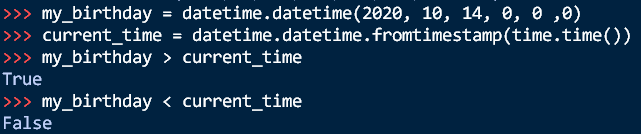


Calling datetime.datetime.now() returns a datetime object for the current date and time according to our computers clock. This tells us the following : Year, Month, Day, Hour, Minute, Second, Microseconds.

We can convert a unix epoch timestamp to a datetime object with the datetime.datetime.fromtimestamp() function.

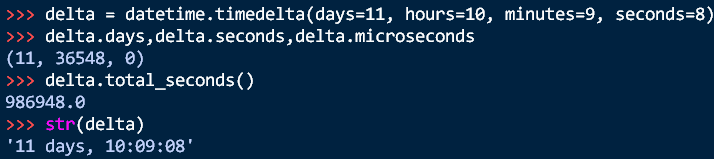


Another example:



## Timedelta Data Type

The datetime module also provides a timedelta data type, which represent a duration of time rather than a moment in time.

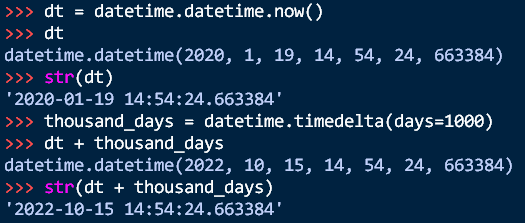


To create a timedelta object , we use datetime.timedelta(). This function only takes weeks, days, hours, minutes. Notice it doesn’t take months or years since a month or year can be variable depending on what month or year it is.

A time.delta() function has the total duration returned in days, seconds and microseconds.

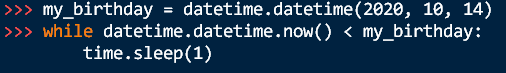
The total\_seconds() method will return the duration in number of seconds alone. Passing a timedelta object to str() will return a nicely and human readable form.

## Some arithmetic with datetime.



## Parsing until a specific date

Using time.sleep() method we can pause a program for a certain number of seconds. By using a while loop, we can pause a program until a specific date. For example:



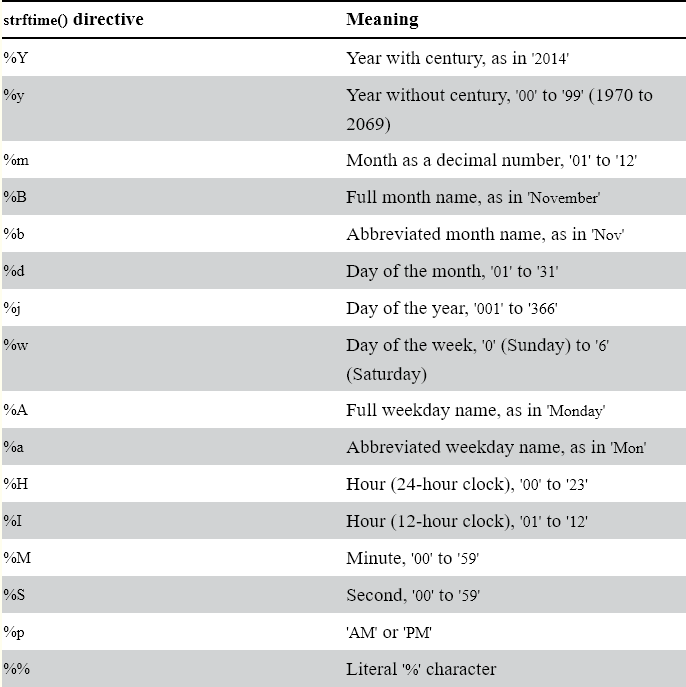
So what’s happening : our program checks every second if the current date is less than our birthday date.

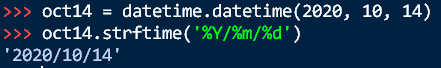
Time.sleep() pauses our computer so it doesn’t waste CPU resources checking the time over and over. This program will continue until my\_birthday is higher than current time(or whenever we stop it).

## Converting datetime Objects into strings

Epoch timestamps and datetime objects aren’t very friendly to the human eye. We use the strftime() method to display a datetime object as a string. (the f in the name stands for function).

### Strftime() directives





## Converting strings into datetime objects

If you have a string of date information such as ‘2020/10/14’ or ’October 14, 2020’ and need to convert it into a datetime object we can use the datetime.datetime.strptime() – the p stands for parse. This is the opposite of strftime() . A strptime() method uses the same directives as strftime() and knows how to parse and understand the string.



