Tick

In python, the time instants are counted since 12 AM, 1st January 1970. The function time() of the module time returns the total number of ticks spent since 12 AM, 1st January 1970. A tick can be seen as the smallest unit to measure the time.

Consider the following example.

Example

```
import time;

#prints the number of ticks spent since 12 AM, 1st January 1970

print(time.time())
```

The localtime() functions of the time module are used to get the current time tuple. Consider the following example.

Example

```
import time;

#returns a time tuple

print(time.localtime(time.time()))
```

The time is treated as the tuple of 9 numbers. Let's look at the members of the time tuple.

Index	Attribute	Values
0	Year	4 digit (for example 2018)
1	Month	1 to 12
2	Day	1 to 31
3	Hour	0 to 23
4	Minute	0 to 59
5	Second	0 to 60
6	Day of week	0 to 6
7	Day of year	1 to 366
8	Daylight savings	-1, 0, 1 , or -1

Python sleep time

The sleep() method of time module is used to stop the execution of the script for a given amount of time. The output will be delayed for the number of seconds given as float.

Consider the following example.

Example

```
import time
for i in range(0,5):
    print(i)
    #Each element will be printed after 1 second
    time.sleep(1)
```

The datetime Module

The datetime module enables us to create the custom date objects, perform various operations on dates like the comparison, etc.

To work with dates as date objects, we have to import datetime module into the python source code.

Consider the following example to get the datetime object representation for the current time.

Example

```
import datetime;

#returns the current datetime object

print(datetime.datetime.now())
```

Output:

```
2018-12-18 16:16:45.462778
```

Creating date objects

We can create the date objects by passing the desired date in the datetime constructor for which the date objects are to be created.

Consider the following example.

Example

```
import datetime;

#returns the datetime object for the specified date

print(datetime.datetime(2018,12,10))
```

Output:

```
2018-12-10 00:00:00
```

We can also specify the time along with the date to create the datetime object. Consider the following example.

Example

```
import datetime;

#returns the datetime object for the specified time

print(datetime.datetime(2018,12,10,14,15,10))
```

Output:

```
2018-12-10 14:15:10
```

Comparison of two dates

We can compare two dates by using the comparison operators like >, >=, <, and <=.

Consider the following example.

Example

```
from datetime import datetime as dt
#Compares the time. If the time is in between 8AM and 4PM, then it prints working hours
otherwise it prints fun hours
if dt(dt.now().year,dt.now().month,dt.now().day,8)<dt.now()<dt(dt.now().year,dt.now().mo
nth,dt.now().day,16):
    print("Working hours....")
else:
    print("fun hours")
```

Output:

```
fun hours
```

The calendar module

Python provides a calendar object that contains various methods to work with the calendars.

Consider the following example to print the Calendar of the last month of 2018.

Example

```
import calendar;
cal = calendar.month(2018,12)
#printing the calendar of December 2018
print(cal)
```

calendar of whole year

The prcal() method of calendar module is used to print the calendar of the whole year. The year of which the calendar is to be printed must be passed into this method.

Example

```
import calendar

#printing the calendar of the year 2019
calendar.prcal(2019)
```