What’s Multithreading?

The multithreading library is lightweight, shares memory, responsible for responsive UI and is used well for I/O bound applications. However, the module isn’t killable and is subject to the GIL  
Threading library in Python Multiple threads live in the same process in the same space, each thread will do a specific task, have its own code, own stack memory, instruction pointer, and share heap memory. If a thread has a memory leak it can damage the other threads and parent process.

import threading

def calc\_square(number):

print('Square': , number \* number)

def calc\_quad():

print('Quad': , number \* number \* number \* number)

if \_\_name\_\_ == "\_\_main\_\_":

number = 7

thread1 = threading.Thread(target=calc\_square, args=(number,))

thread2 = threading.Thread(target=calc\_quad, args=(number,))

# Will execute both in parallel

thread1.start()

thread2.start()

# Joins threads back to the parent process, which is this

# program

thread1.join()

thread2.join()

# This program reduces the time of execution by running tasks in parallel

What’s multiprocessing?

The multiprocessing library uses separate memory space, multiple CPU cores, bypasses GIL limitations in CPython, child processes are killable(ex. function calls in program) and is much easier to use. Some caveats of the module are a larger memory footprint and IPC’s a little more complicated with more overhead.  
Checkout Multiprocessing library in the Python docs

import multiprocessing

def calc\_square(number):

print('Square': , number \* number)

result = number \* number

print(result)

def calc\_quad():

print('Quad': , number \* number \* number \* number)

if \_\_name\_\_ == "\_\_main\_\_":

number = 7

result = None

p1 = multiprocessing.Process(target=calc\_square, args=(number,))

p2 = multiprocessing.Process(target=calc\_quad, args=(number,))

p1.start()

p2.start()

p1.join()

p2.join()

# Wont print because processes run using their own memory location

print(result)

https://realpython.com/python-gil/