What is Multitasking

It’s an ability of operating system to perform multiple tasks simultaneously. For example you are working on jupyter notebook, downloading a movie and listening to a song. All these tasks are performed by the same OS an in sync.

There are two types of multitasking in an OS:

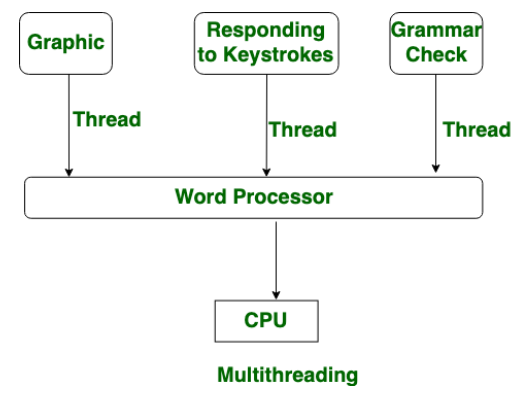
* Process-based
* Thread-based

Let's begin with multithreading...

Let’s take an example of FIFA game on your pc, the game as whole is a single process, but it consist of several threads responsible for playing the music, taking user input, running opponent synchronously etc. So what's a thread? Its an ***independent*** flow of execution where each thread of a process is responsible to perform a particular task.

Let's take an another example -

I have grabbed this image from internet and it clearly explains the concept of threading, lets say you're working on Microsoft Word, then Ms-word is the process and responding to each keystrokes, showing various options, grammar check, word count are 4 different threads.



Source - <https://media.geeksforgeeks.org/wp-content/uploads/20190522151811/Untitled-Diagram-351.png>

Remember, process has one "main thread" that is always running. The main thread creates the child thread objects and it is also initiated by the main thread.

So what is multiprocessing? It’s an ability of a system to support more than one processor at a time to execute more than one operation at any given instance

where applications are broken down into small pieces and then executed independently of each other to increase the efficiency and reduces the overall run time.

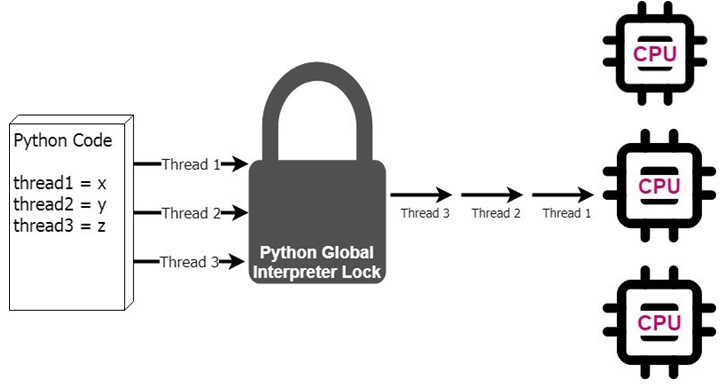
Consider a situation, say we have assigned several processes to a single processor system then it will have to interrupt each task in between and switch over to the next, to keep all of the processes running. This will not only increase the amount of time to complete the task but also will reduce the overall efficiency of the system.

If we look into a multiprocessing system, it can have one processor with multiple cores which have the an ability to run different tasks independently at the same time. This will not only increases the efficiency of the system but in long run it will also reduces the runtime of the system by a significant amount.

To understand this concept we will take an example of a chef, if he is working in a kitchen alone, then only he has to do several tasks like baking, stirring, kneading dough, etc. But when he is being assisted by his assistants, the tasks are divided and the chef doesn’t need to switch between his tasks.

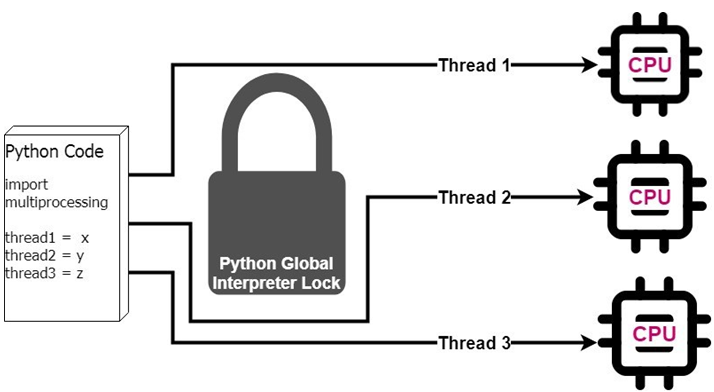
Let's understand how multithreading works in python.

Although we say python supports multithreading but its different from whats happening behind the scenes.Most programming languages don’t take advantage of multiple cores. Languages like Java and C automatically send tasks to multiple CPUs simultaneously. But in python each process executes on an single core, so multiple threads of same process will execute on the same core and thus will share resources and the memory space. To prevent one thread from altering the result of another thread since they share same resources Python has a concept of GIL. the global interpreter lock (GIL), allows only one thread to use the shared memory space and resources at a time and does intelligent context switching between threads so no thread can alter another threads working. The GIL was implemented to handle a memory management issue, but as a result, Python is limited to using a single processor.



Source - [https://miro.medium.com/max/875/1\*7\_RyuHshae\_-yuY-Tj59kg.jpeg](https://miro.medium.com/max/875/1*7_RyuHshae_-yuY-Tj59kg.jpeg)

In case of multi-processing, each process runs on different core depending upon the number of cores on the machine. Hence there’s no need of a GIL while doing multi-processing since all processes are independent. Thus multi-processing is actually providing us the actual parallelism in python.



Source - [https://miro.medium.com/max/875/1\*iEE5\_Am-9x7iCVnn6F1MGQ.jpeg](https://miro.medium.com/max/875/1*iEE5_Am-9x7iCVnn6F1MGQ.jpeg)

Remember, Multiprocessing is parallelism. Multithreading is concurrency.

Concurrency is a approach that is used for decreasing the response time of the system by using the single processing unit. Concurrency is that the illusion of parallelism, however in actual the chunks of a task aren’t parallelly processed. Whereas in Parallelism means that an application splits its tasks up into smaller subtasks which can be processed in parallel, for instance on multiple CPUs at the exact same time.

You should also know some difference between multithreading and multitasking, threads run in the same memory space, while processes have separate memory . This makes it a bit harder to share objects between processes with multiprocessing. Since threads use the same memory, precautions have to be taken or two threads will write to the same memory at the same time. This is what the global interpreter lock is for.

So when to use multiprocessing or mutithreading?

I'll give you a rule of thumb to decide when to use which multitasking, if the task is CPU heavy then go for multitasking otherwise go for multithreading.