ASSIGNMENT REPORT 1: PROCESS AND THREAD IMPLEMENTATION

CENG2034, OPERATING SYSTEMS

Söz Tuana KURŞUN
soztuanakursun@mu.edu.tr

Monday 4th May, 2020

Abstract

In this study, we have seen how operating systems manage processes or memory and we learned that every process is actually pid.We learned that every folder or processor is a file.Linux commands and what can be done with these commands, linux is written in c language and we use os library to use it as native in python and we learned that we can manage with the data structures of python.On the other hand,we learned multithreading.It is the ability of a program to perform more than one job at the same time.We have learned that multithreading is more useful on a time basis if there is file reading operation.We learned that all threads started in the process share the same memory.Each new process runs independently from other processes,we learned that it is not shared between memory processes and that each proccess uses separate memory addresses.

1 Introduction

The purpose of this laboratory is to see how operating systems manage processes and memories and in the language of python is to see the differences in multiprocessing and multithread.In this lab, we learned to learn the differences and aim to create better quality software.

2 Assignments

In this experiment, although we learned some of the linux commands, we also used them.The following commands are what we use in the experiment.

2.1 Assignment print(”PID of this process: ”, pid)

We learned how a processor's pid address works.

2.2 Assignment print(”loadavg: ”, os.getloadavg())

With this command, we learned when our computer turned on and off, how long it turned on.

2.3 Assignment print(”CPU core count: ”, cpucount)

We learned how many CPUs there are.

2.4 Assignment print (”multithreading”)

Using multithreading we have seen whether it saves time.

3 Results

-As a result, we learned that every file, every folder can have a pid and all of them are different.

-We learned that if our operating system is linux, we can easily find loadavg.If not, we learned that we can do a few operations and learn.

-We learned that the values in loadavg are the output of the data in the 1st minute, 5th minute and 15th minute.We also learned how many cores our computer is.

-We learned multiprocess and multithreading in python.we learned the differences and convenience of these two.

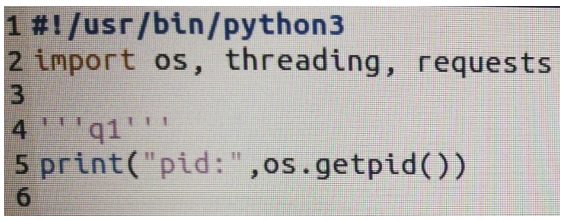

Figure 1: This is a pid of run.With this code, we can learn the pin of each processor.

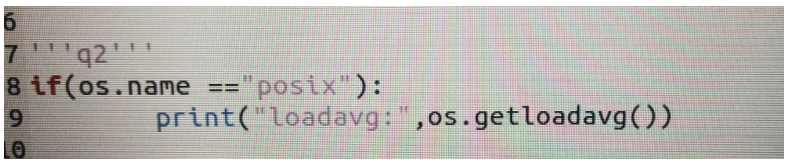

Figure 2: With this command, we learned when our computer turned on and off, how long it turned on.

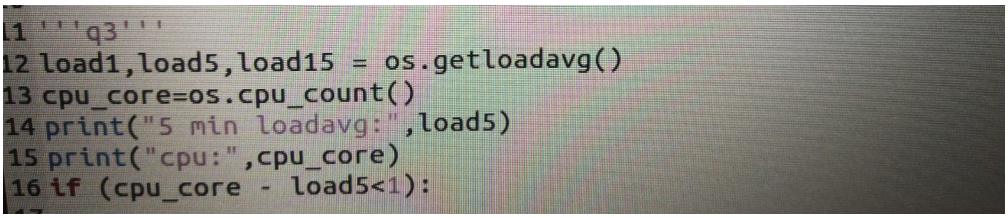

Figure 3: We learn how many cores our computer is.We have seen that we can write with nproc.If your processor is 4-core and your incoming transactions are more than 4, we learned that the transactions made should be in line and wait.

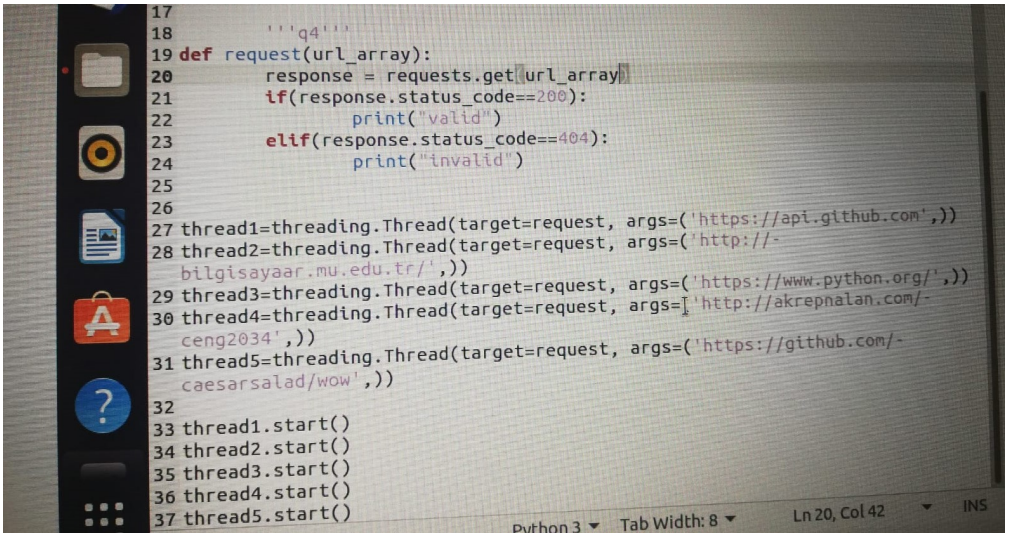

Figure 4: Case study with multithreading.

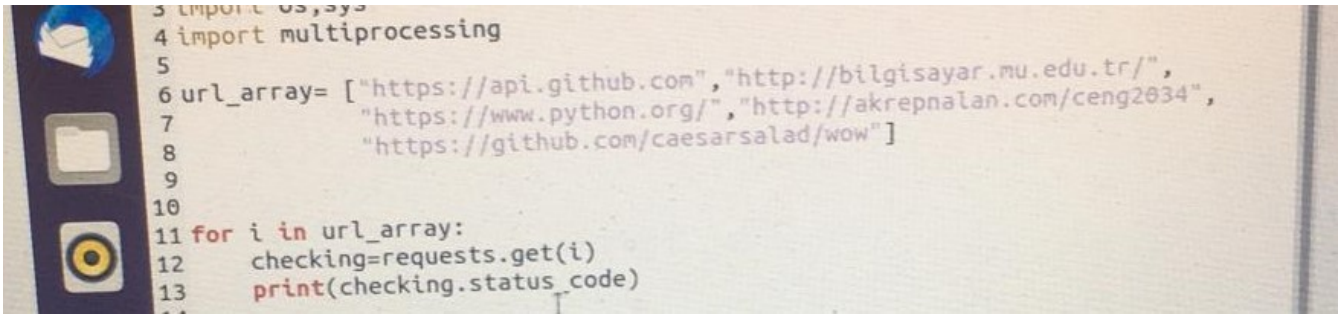

Figure 5: Case study with multiprocess.

4 Conclusion

As a result, at the end of this experiment, as I mentioned at all stages, we learned that all processes are files and they all have a special pid.Aactually, there are 3 important issues we learned in this lab.Firstly, contrary to the fact that os lesson is seen more theoretically,we have seen that we can make better software by knowing the operating system more closely with our lab.Another issue is CPU.If you have a cpu heavy task, and you want to make it faster use multiprocessing!For example, if you have 4 cores as in the examples we did, each multi-threaded core will have a capacity of approximately percent of 25 whereas with multi-processing you get percent of 100 in each core. This means that you will gain a speed of 4 in percent of 100 of 4 cores.Another topics we learned there can only be one thread running at any given time in a python process, multiprocessing is parallelism;multithreading is concurrency.Multiprocessing is for increasing speed and Multithreading is for hiding latency.Multiprocessing is best for computations. Multithreading is best for IO.

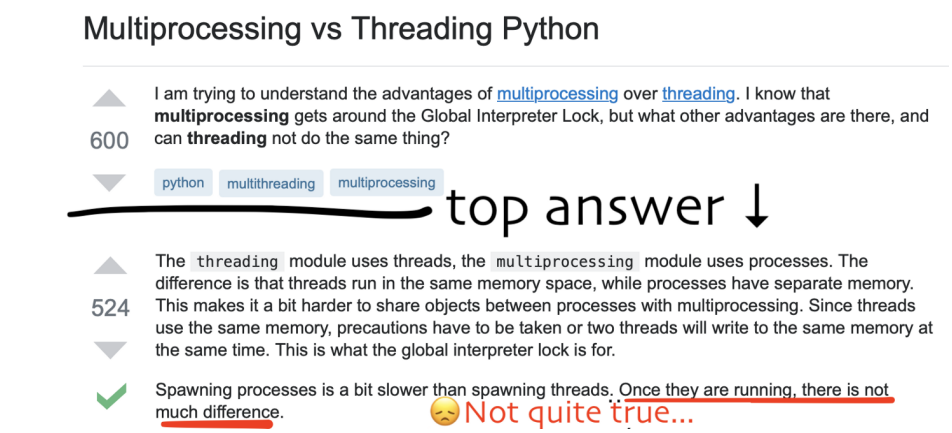

Figure 6: Other issues we learned multiprocessing and multithreading in python.In the picture above, we have seen a brief definition of the two terms.

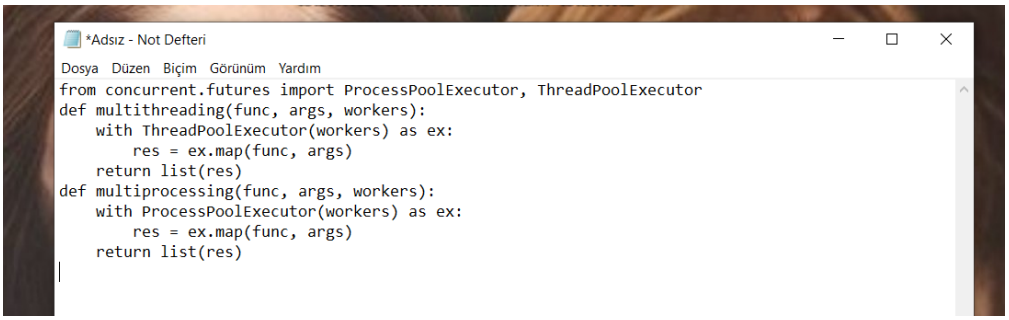

Figure 7: Now we will see the difference between the two with a code.

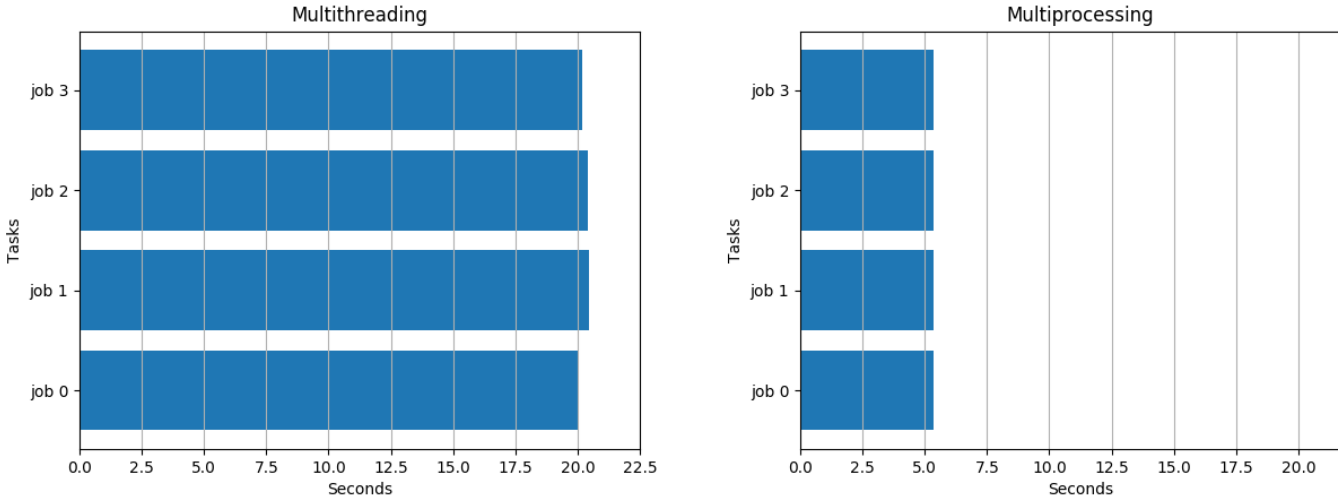

Figure 8: While Multithreading took 20 seconds, Multiprocessing took only 5 seconds. So now that we are convinced that they're not the same, we would like to know why.We have seen and understood the difference from the pictures above.