Deree
School of Graduate and Professional Education

# Big Data Architectures
# Winter 2023

## Assignment #1

**Due: Sunday Feb 19, 2023, 23:59.**

Exercise 1 (25 points)

Write a parallel Python program in file num_primes.py that uses the ProcessPoolExecutor class to find the number of prime numbers that are < max_num and reports on its timing requirements. Summarize your results, i.e., the size of the problem (max_num = 1000, 10000, 100000, 1000000, 10000000, 100000000) vs. the number of processes (4, 8, 16). A sample output is shown below.

```
# Performance with linear ranges
#                  16        8        4  (nprimes)
#  100000000   456.92   467.27   582.38    5761455
#   10000000    16.66    18.12    21.28     664579
#    1000000     0.72     0.76     0.89      78498
#     100000     0.08     0.09     0.08       9592
#      10000     0.06     0.04     0.03       1229
#       1000     0.06     0.04     0.03        168
```

Exercise 2 (25 points)

Assume we have 4 files f1.txt, f2.txt, f3.txt, f4.txt, each one containing 1048576 ($2^{20}$) numbers. We want to find the maximum and the minimum number contained in all files. Write a parallel Python program in file max_min_num.py that uses the ThreadPoolExecutor class to find the minimum and maximum of the numbers contained in the 4 files. Four threads should be used, each one assigned to a different file. You also need to write code in file gen_nums.py to generate random numbers for each file.

Exercise 3 (25 points)

Write a Python program in file ping_pong.py that spawns two threads: one called ping and the other called pong. Both threads iterate a number N of times; the first one prints “ping” on the output, the second prints “pong” on the output. The two threads must synchronize their actions, so that the output produced is N lines, each one reading “ping pong”.

# Big Data Architectures
# Winter 2023

Exercise 4 (25 points)

Consider the operation of a network printer. The printer maintains a queue of print jobs to be executed, which contains a finite number of slots. Multiple processes may produce print jobs, which they add to the next available slot of the printer's queue. The printer picks the next job from the queue and executes it, i.e., prints whatever the job asks for. Assume that there are N processes and that each process iterates M times. In each iteration, a process produces a print job and then sleeps for a random amount of time. If the print queue is full, a process must wait for a slot to become available. Moreover, if the print queue is empty the printer must wait for a print job to become available. Write a Python program in file process_printer.py, in which the processes and the printer are represented as threads, and print their actions on the output, i.e., production of a print job, and execution of a print job.

Submission

Put all .py files into a rar or zip compressed file with name <Lastname>_<Firstname>.rar or .zip (e.g., Nikolaou_Maria.zip) and submit it to Blackboard.

Penalties

Violation of any naming conventions will result into 15 points reduced from your grade.