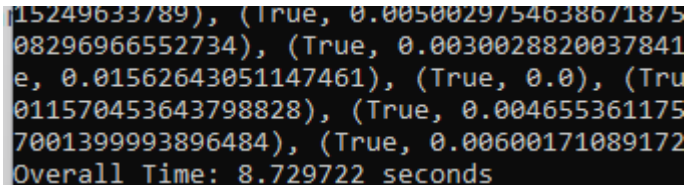
Task 1

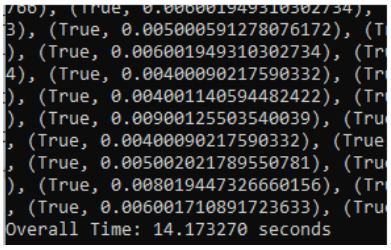
Python code in the accompanied py file.

Output

2 cores



1 core



Lessons learned

1. When the pool size is 2 the program can run 2 tasks parallel with one another which improves performance by utilising more of the CPU.

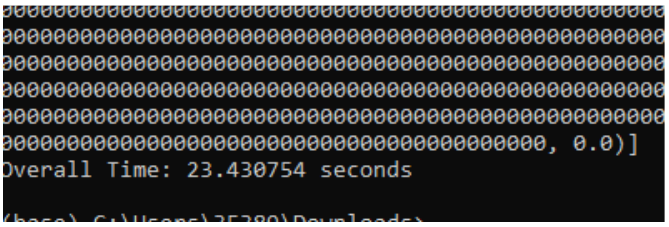
2. The prime checking function is highly efficient since it takes close to 0 seconds to check each prime number even to 6 decimal places.
3. Larger calculations benefit more from greater parallelism. The time difference between using pool size 1 and 2 will be greater for larger calculations.

4. When the calculations are small the pool size of one can actually take less time due to the associated overhead costs of parallelism.

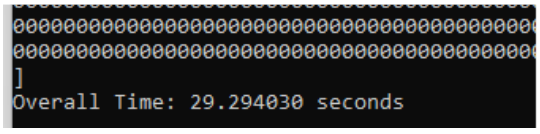
Task 2

Python code in the accompanied py file.

Output
2 cores



1 core



Lessons learned

1. Pool size of 2 is faster than pool size of one due to parallelism allowing for more CPU usage
2. At lower calculation sizes the time difference between pool size one and two is not as dramatic. I.e parallelism has greater merit for larger calculations.
3. At lower calculation sizes multicore processing may actually take longer because of the associated overhead with higher parallelism.
4. At very large sizes the speedup may be limited by the serial parts of the processing which cannot be made parallel.