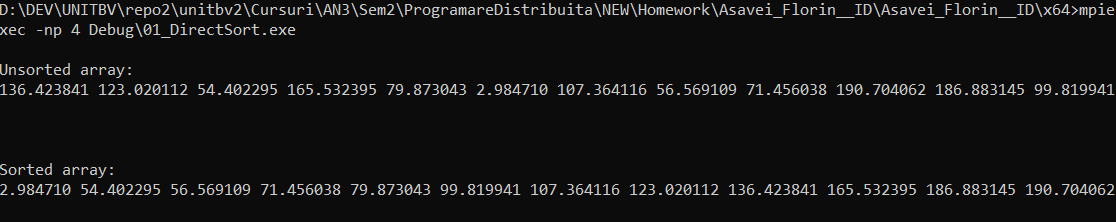
# Observations:

# 1. Direct Sort

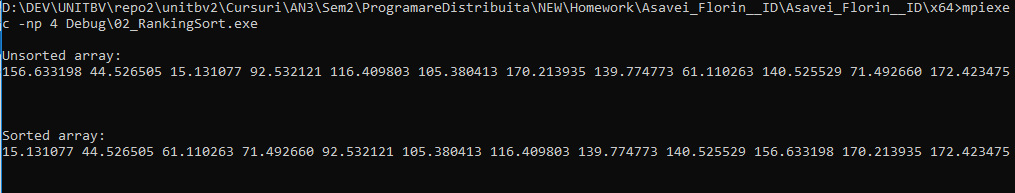
* 1. Check that algorithm works (number of elements = 12) with 4 processors



* 1. Performance Observations:
* The execution time decrease by more than half when using 4 processors.
* Best efficiency is achieved when using 2 processors
* Compared to the other 3 algorithms, it is the fastest in all scenarios

# Ranking Sort

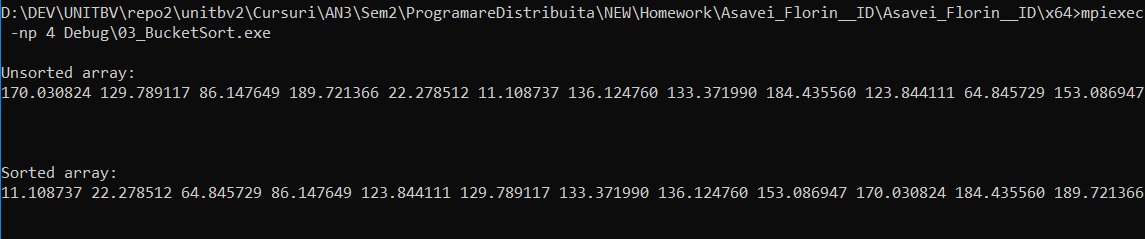
* 1. Check that algorithm works (number of elements = 12) with 4 processors



* 1. Performance Observations:
* This is the slowest algorithm ~ 15.000 times slower than the others
* Execution time is reduced in half just by using 3 processors and it becomes 3 times faster if we use 4 processors
* This algorithm benefits the most out of using Parallel Computing
* It is the only one that achieved more than 100% efficiency (not sure if relevant since it took a lot of time and there are other programs running on the laptop)

# Bucket Sort

* 1. Check that algorithm works (number of elements = 12) with 4 processors

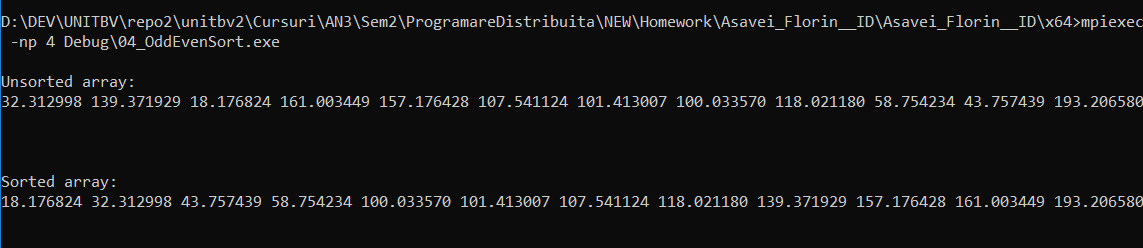


3.2 Performance Observations:

* Second fastest algorithm (after Direct Sort)
* The speedup is most linear out of all algorithms

# Odd Event Sort

4.1 Check that algorithm works (number of elements = 12) with 4 processors



3.3 Performance Observations:

* The second slowest algorithm out of the 4
* It the the lowest overall efficiency
* Best efficiency achieved when using 2 processors
* Not much improvement is achieved if we use more than 2 processors

# Overall Observations:

All algorithms resulted in better execution times when using multi-processing.

It seems that the best Efficiency is achieved when using 2 processors.