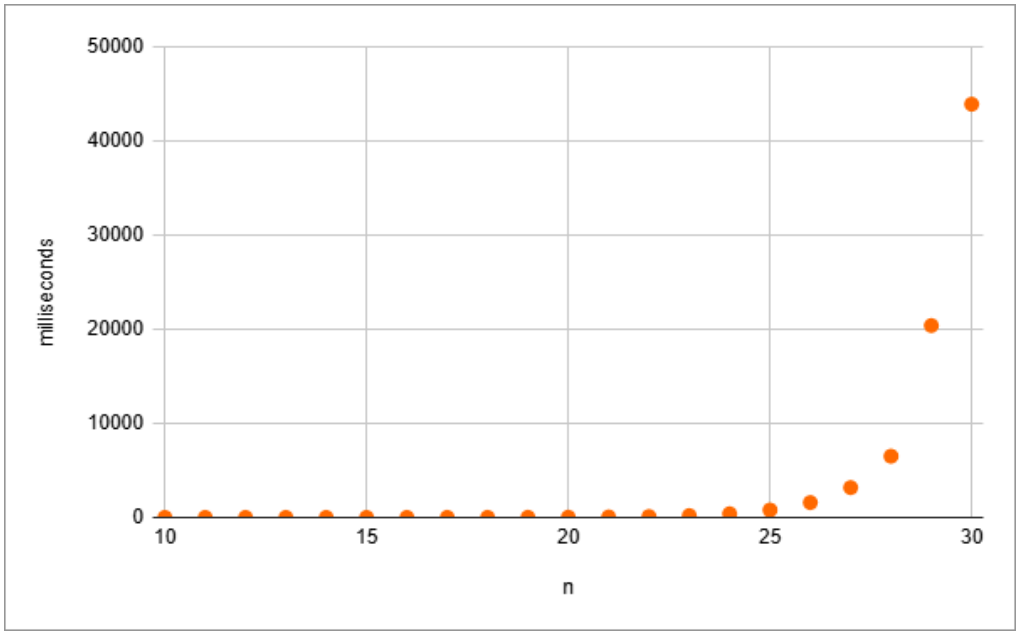
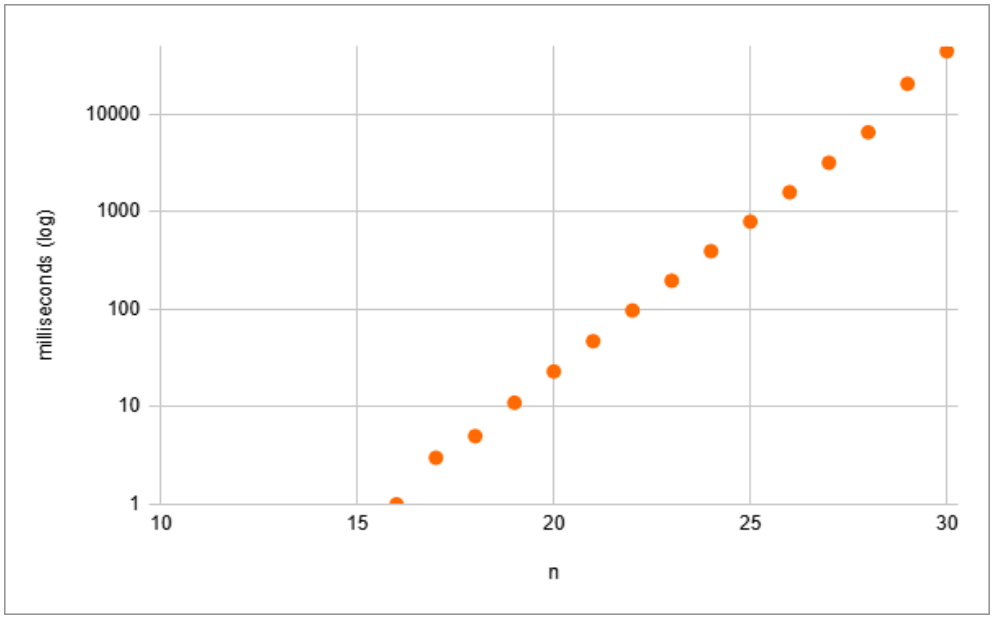
Leah Blasczyk
HW02
GitHub Username: **Lebrra**
(Assignment 2 Path: https://github.com/Lebrra/repo759/tree/main/HW02)

This document holds plots + data for task1 as well as data + analysis for task3.

task1 plots:



^ with linearly-scaled y



^ with log-scaled y
n = 10-15 took a rounded 0 ms, so they aren't visible on the log graph. (see data below)

HW02/Task2_1.out contents:

```
Tue Sep 24 09:48:55 PM CDT 2024
Results:
element count:          1024
time to process:        0 milliseconds
first element:          0.415078
last element:           -14.7482

Results:
element count:          2048
time to process:        0 milliseconds
first element:          -0.725967
last element:           7.44686

Results:
element count:          4096
time to process:        0 milliseconds
first element:          -0.763267
last element:           5.2375

Results:
element count:          8192
time to process:        0 milliseconds
first element:          0.981854
last element:           13.2958

Results:
element count:          16384
time to process:        0 milliseconds
first element:          0.819953
last element:           -16.1637

Results:
element count:          32768
time to process:        0 milliseconds
first element:          -0.848353
last element:           -57.1423

Results:
element count:          65536
```

```
time to process:        1 milliseconds
first element:          -0.0434504
last element:           59.1158

Results:
element count:          131072
time to process:        3 milliseconds
first element:          -0.397422
last element:           154.651

Results:
element count:          262144
time to process:        5 milliseconds
first element:          -0.942284
last element:           178.389

Results:
element count:          524288
time to process:        11 milliseconds
first element:          0.764051
last element:           -518.685

Results:
element count:          1048576
time to process:        23 milliseconds
first element:          0.380201
last element:           137.978

Results:
element count:          2097152
time to process:        47 milliseconds
first element:          0.99983
last element:           531.947

Results:
element count:          4194304
time to process:        97 milliseconds
first element:          0.30884
last element:           2981.13

Results:
```

```
element count:          8388608
time to process:        196 milliseconds
first element:          0.112124
last element:           -1208.65

Results:
element count:          16777216
time to process:        393 milliseconds
first element:          -0.887774
last element:           -1106.53

Results:
element count:          33554432
time to process:        789 milliseconds
first element:          0.263372
last element:           4236.73

Results:
element count:          67108864
time to process:        1580 milliseconds
first element:          0.850295
last element:           -12618.3

Results:
element count:          134217728
time to process:        3170 milliseconds
first element:          0.586998
last element:           2127.44

Results:
element count:          268435456
time to process:        6504 milliseconds
first element:          -0.0293931
last element:           7900.27

Results:
element count:          536870912
time to process:        20392 milliseconds
first element:          -0.621617
last element:           -11645.8
```

```
Results:
element count:          1073741824
time to process:        43914 milliseconds
first element:          0.969748
last element:           19435.8
```

task3 outputs: (2 variants)

```
Matrix size:    1010

matmul.mmul1()
time to process:        3353 milliseconds
last element:           13.4478

matmul.mmul2()
time to process:        835 milliseconds
last element:           13.4478

matmul.mmul3()
time to process:        18579 milliseconds
last element:           13.4478

matmul.mmul4()
time to process:        3368 milliseconds
last element:           13.4478
```

```
Matrix size:    1234

matmul.mmul1()
time to process:        12628 milliseconds
last element:           5.26819

matmul.mmul2()
time to process:        1541 milliseconds
last element:           5.26819

matmul.mmul3()
time to process:        35407 milliseconds
last element:           5.26819

matmul.mmul4()
time to process:        12646 milliseconds
last element:           5.26819
```

task3 analysis:

There are some pretty significant time differences between mmul 1, 2, and 3. Since all arrays are row-ordered, it makes sense that the fastest algorithm is the one that spends the majority of the calculations moving up one index in the space (mmul2) by iterating j in the innermost loop and k in the second innermost loop. k and i are used as multipliers, meaning that whenever they change we are likely needing to jump several spaces to get the value at this index. With this in mind, it shows that mmul1 is the second-fastest because i is still the outermost-loop and mmul3 is the slowest because both multiplier iterators (i and k) are contained within the j iterator, causing very sporadic position calling on every set. As for mmul1 and mmul4, they seemed to run nearly identically.