Individual Report:

Austin Celestino

CS 4080

Time Taken for the 10x10 Matrices:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Array Size/Time Taken (Microseconds) | C: Dynamic Stack Array | C: Pointer to Pointer | C++: Pointer Matrix Class | Java: Matrix Arrays |
| 10x10 | 0 | 0 | 12 | 0 |

With the 10x10 array, the array size is too short for any meaningful time comparison.

Test Sets:

Test set 1:

|  |
| --- |
| 11.88268 46.10042 82.22092 91.8576 80.97411 95.58794 63.3352 17.77189 61.34262 67.89339 |
| 20.83487 85.9746 53.37753 99.99346 50.70716 92.29951 62.76533 53.34921 45.75474 84.14875 |
| 73.83357 15.00873 78.10734 60.80909 42.6504 27.70327 63.54046 96.93647 64.17813 3.20995 |
| 89.36193 14.50199 95.45718 9.82123 21.76149 68.00308 54.87832 15.90527 70.41325 97.6751 |
| 88.28802 86.14418 77.35746 19.24965 92.8457 36.07177 90.90753 65.58483 27.27671 82.51533 |
| 56.95119 13.96925 15.93588 93.17759 32.77599 48.5471 12.25644 18.45217 12.16192 40.86183 |
| 2.50172 71.01943 43.44132 38.0733 98.07915 22.61079 11.88235 79.74879 16.57686 5.3438 |
| 91.70107 46.93692 61.83292 32.63616 97.54681 56.32348 24.17256 45.41052 91.81788 11.30964 |
| 51.95377 14.89266 38.46202 29.78476 82.38908 47.80285 26.82352 38.22248 23.32787 59.55963 |
| 92.89609 27.80292 8.31818 57.04672 8.12068 68.1574 24.37482 77.05061 84.8368 4.43678 |

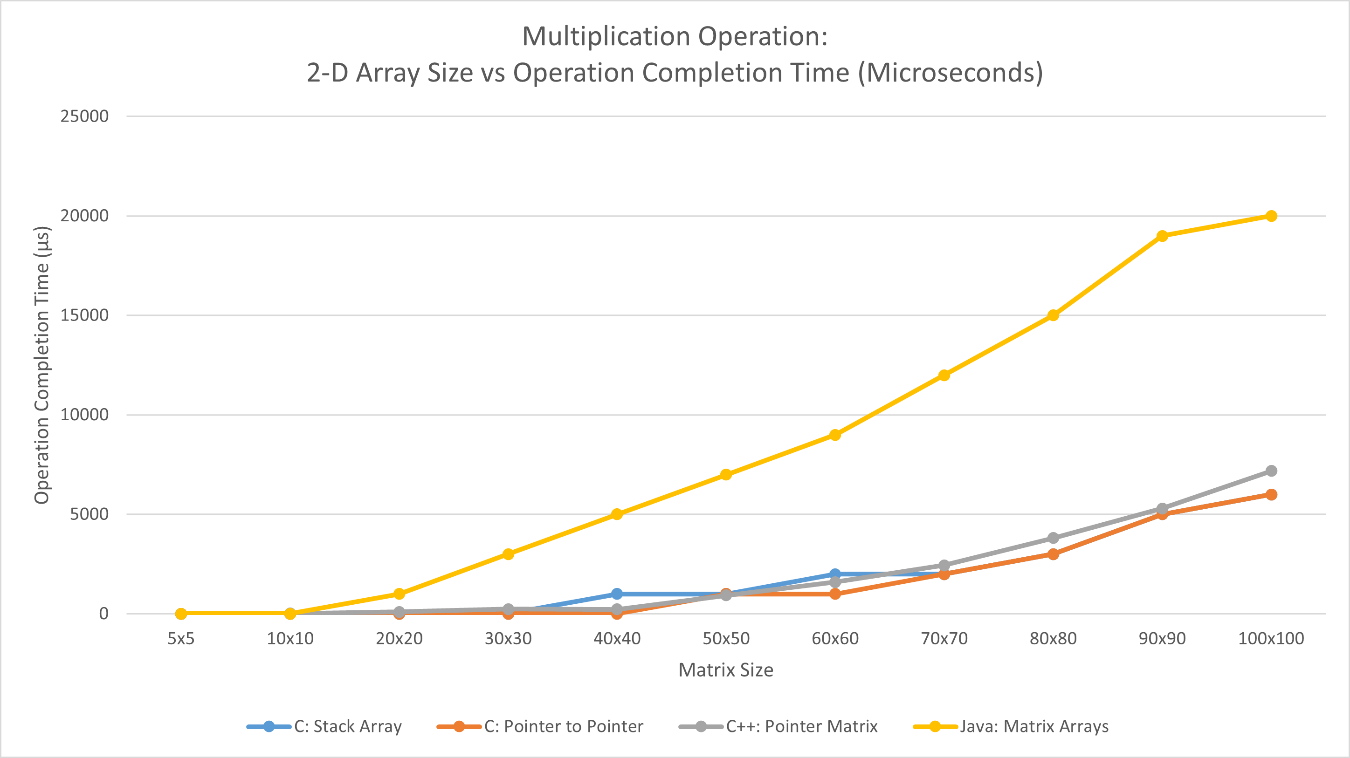
Test set 2:

|  |
| --- |
| 86.48636 86.16865 52.04844 99.75933 47.19867 48.90859 15.561 79.38734 68.72028 41.15845 |
| 99.08433 88.22627 63.21038 10.88784 63.64212 48.60664 89.23099 24.03482 93.31114 9.90954 |
| 31.66073 34.84695 8.83532 54.81295 41.72836 21.12854 30.32627 19.55102 68.4551 87.90889 |
| 73.7971 51.7875 80.67757 71.3871 30.84036 8.30006 9.07632 58.06401 93.0466 22.05416 |
| 2.96409 78.24219 11.44885 90.27423 52.06236 13.31218 31.57608 6.28243 87.60377 73.84575 |
| 18.92453 83.61288 29.81885 23.82887 57.79616 23.0907 91.16988 52.3222 77.34898 88.6468 |
| 28.66603 44.97459 1.3009 55.47194 6.73261 25.57206 58.91473 14.07259 47.01491 4.6729 |
| 23.17579 13.07976 18.15159 92.04391 26.3286 58.50494 49.05568 50.99361 19.13075 83.29056 |
| 56.40352 65.49528 95.67238 11.91739 52.8756 44.29958 20.58172 82.19063 18.72535 46.04959 |
| 18.70554 42.62981 35.81741 41.57701 71.91385 17.21051 62.2998 29.24116 38.52525 32.80444 |

Array Size vs Time taken to complete multiplication:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Array Size/Time Taken (Microseconds) | C: Dynamic Stack Array | C: Pointer to Pointer | C++: Pointer Matrix Class | Java: Matrix Arrays |
| 5x5 | 0 | 0 | 4 | 0 |
| 10x10 | 0 | 0 | 12 | 0 |
| 20x20 | 0 | 0 | 99 | 1000 |
| 30x30 | 0 | 0 | 236 | 3000 |
| 40x40 | 1000 | 0 | 225 | 5000 |
| 50x50 | 1000 | 1000 | 927 | 7000 |
| 60x60 | 2000 | 1000 | 1600 | 9000 |
| 70x70 | 2000 | 2000 | 2445 | 12000 |
| 80x80 | 3000 | 3000 | 3814 | 15000 |
| 90x90 | 5000 | 5000 | 5291 | 19000 |
| 100x100 | 6000 | 6000 | 7185 | 20000 |

Time taken Graph:



Test Matrices for Output:

Matrix 1:

|  |  |  |
| --- | --- | --- |
| 81.82924 | 20.89393 | 82.06377 |
| 49.09642 | 16.95748 | 68.78876 |
| 41.01637 | 9.49074 | 25.07789 |

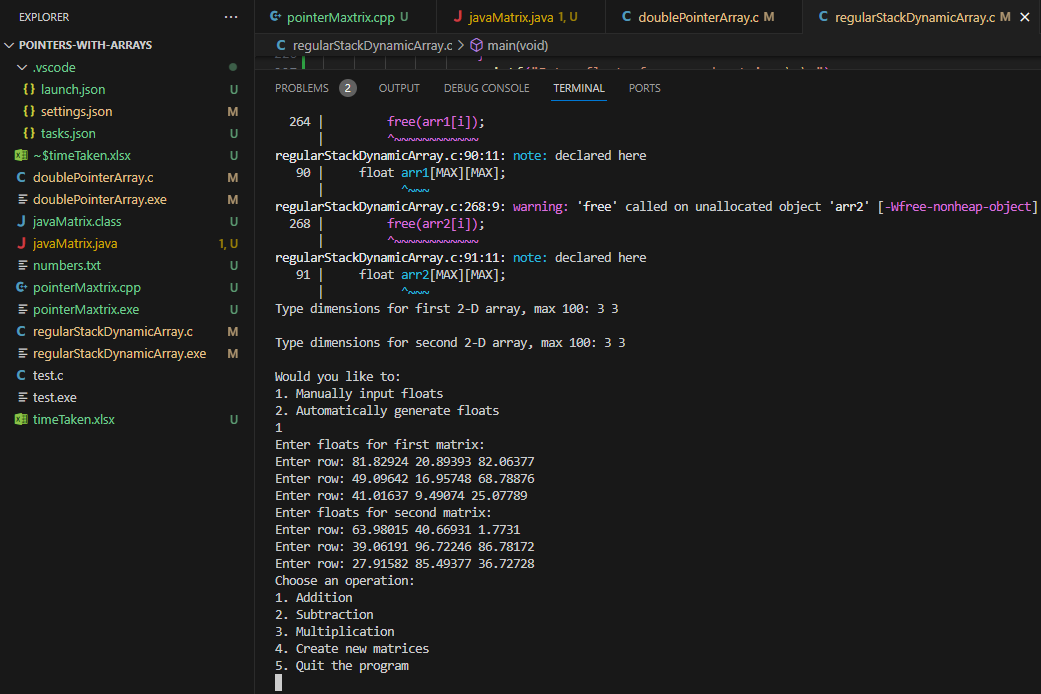
Matrix 2:

|  |  |  |
| --- | --- | --- |
| 63.98015 | 40.66931 | 1.7731 |
| 39.06191 | 96.72246 | 86.78172 |
| 27.91582 | 85.49377 | 36.72728 |

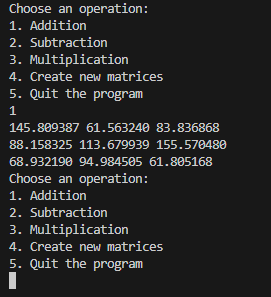
Outputs for each program:

C - Regular Stack Dynamic Array:

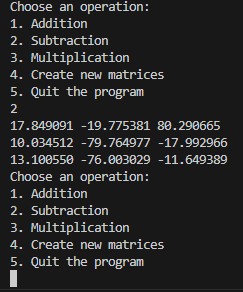
Startup:



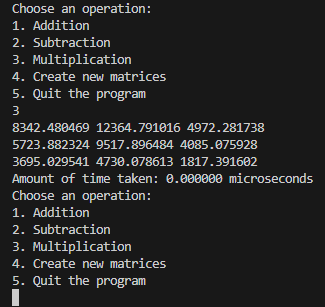
Addition:



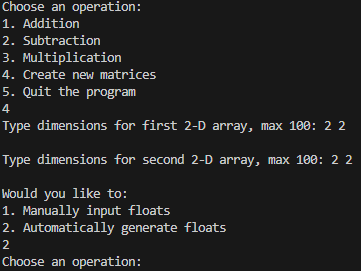
Subtraction:



Multiplication:

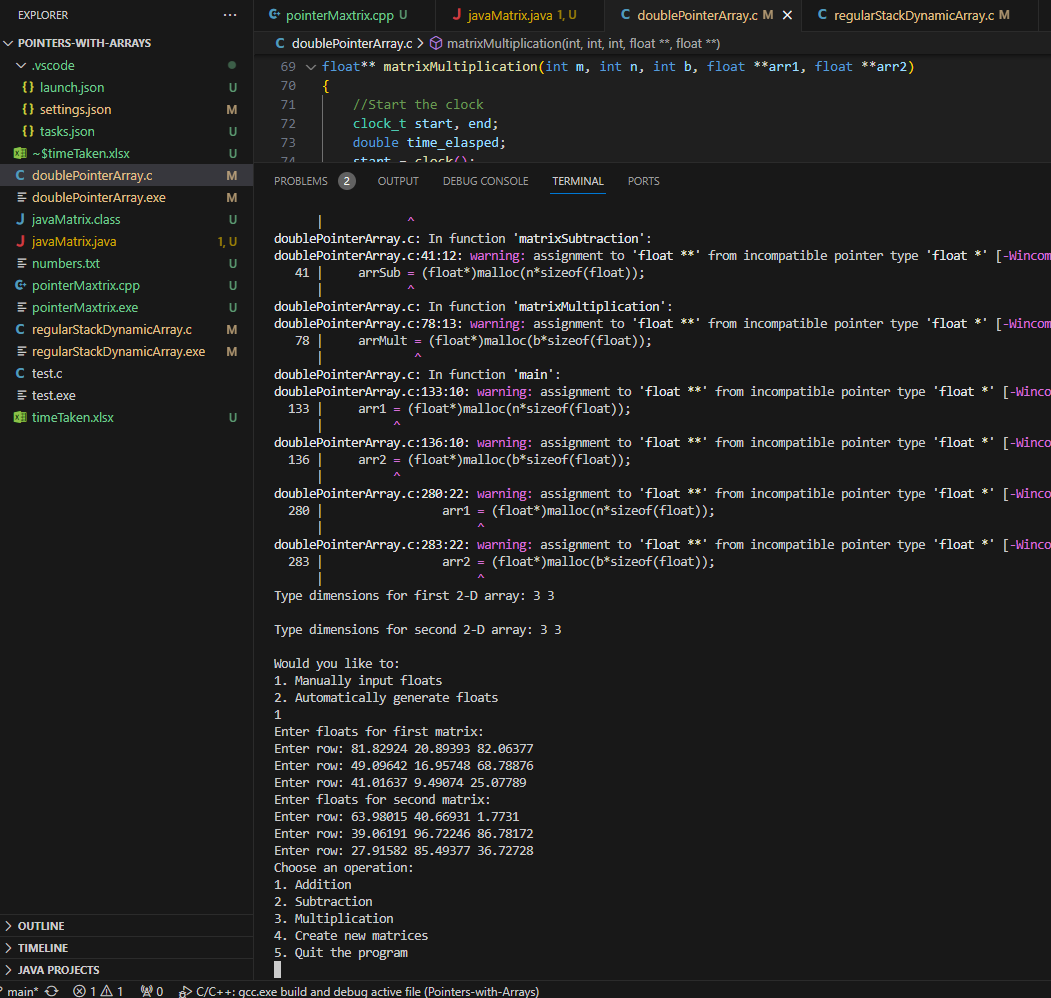


Creating new matrices:

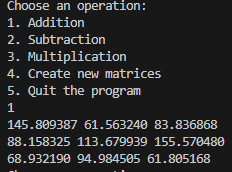


C – Pointer to Pointer:

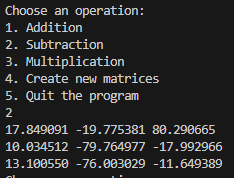
Startup:



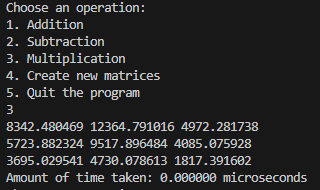
Addition:



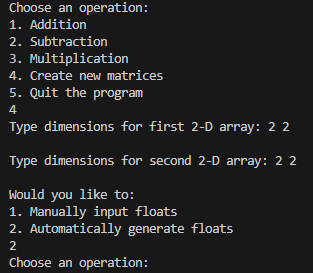
Subtraction:



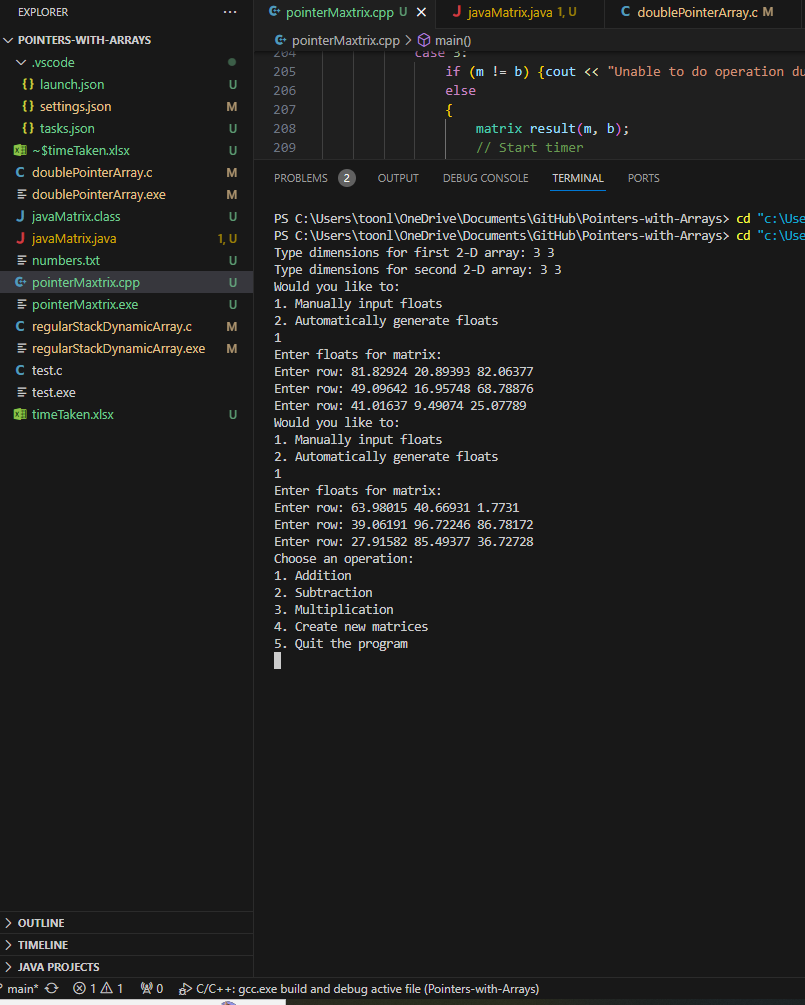
Multiplication:



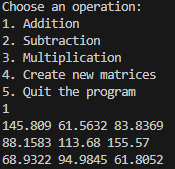
Creating new Matrices:



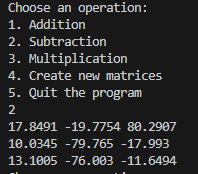
C++ - Pointer Matrices:

Startup:

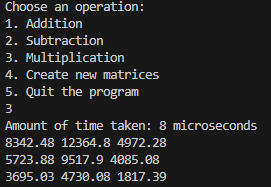
Addition:



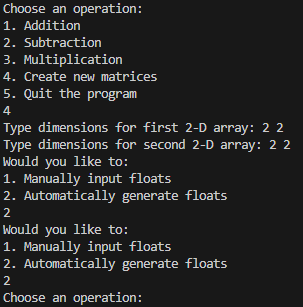
Subtraction:



Multiplication:

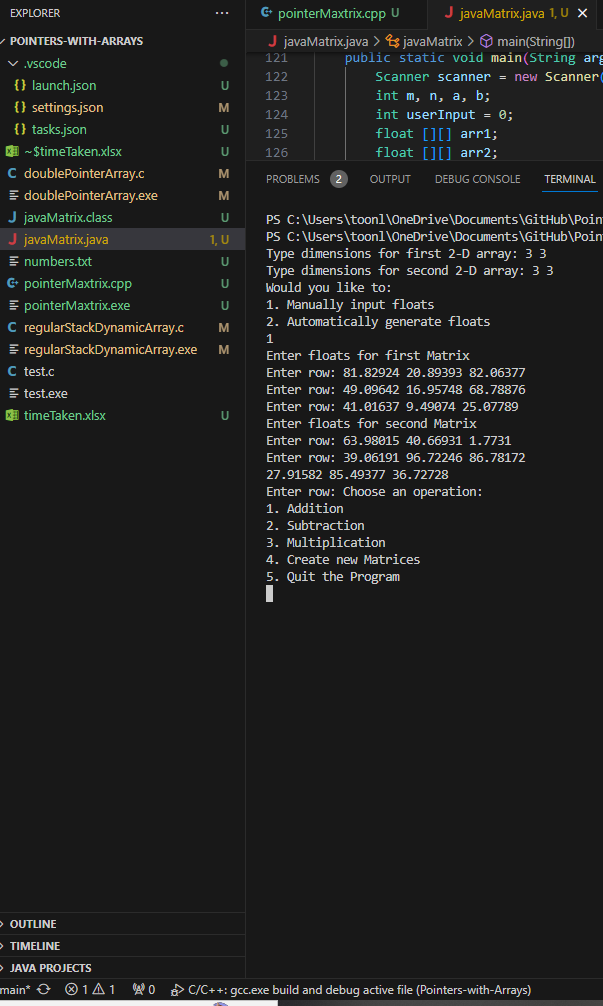


Creating new matrices:

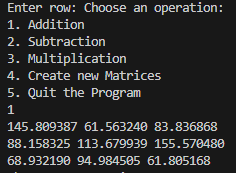


Java – Matrices with Java Classes:

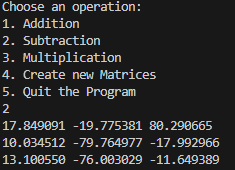
Startup:



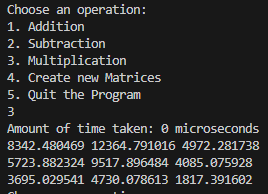
Addition:



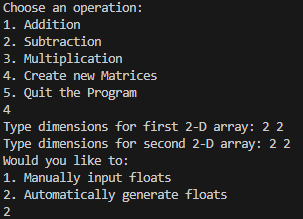
Subtraction:



Multiplication:



Creating new matrices:



Comments:

C was the fastest among the languages, but C++ had the cleanest implementation for me. Java was by far the slowest amongst all of the methods by a large margin. C++ also seemed to have the best accuracy of time amongst all of the languages that were tested.