Kevin Kim
Professor Raheja
CS 4080.03
22 October 2023

Programming Project 1 Report

Comparison Between Four Versions:

The first version of this programming project is a C program that uses a regular stack dynamic two-dimensional array to represent the matrices. After testing the multiplication function on the two 10x10 matrices for this version, I obtained 0.000013 seconds to finish performing matrix multiplication as can be seen toward the end of this report.

Although the second version is also a C program, this version uses a pointer to a pointer that allocates memory in the heap instead of the stack using the malloc function to create dynamic two-dimensional matrices. After testing the multiplication function on the two 10x10 matrices for this version, I obtained 0.000014 seconds to finish performing matrix multiplication, which was extremely close to the result from the first version.

Furthermore, the third version is a C++ program that has a pointer to a pointer as private data in the "Matrix" class. This C++ version also has overloaded operator functions that perform matrix operations like addition, subtraction, and multiplication. In fact, this C++ version was the only version that allowed user-defined overloaded operator functions, which allowed me to utilize two Matrix objects and perform matrix operations by simply using appropriate matrix operation symbols such as "+", "-", and "*" while using the two Matrix objects as the operands. This version definitely took me the longest time to resolve all the segmentation faults and runtime errors as I forgot to overload the assignment operator ("="). Although operator overloading can make code more difficult to debug and lead to errors, I realized that it could be very powerful and help in different aspects, such as readability and reusability. After testing the multiplication function on the two 10x10 matrices for this version, I obtained 0.000079 seconds to finish performing matrix multiplication, which was the slowest result out of all four versions.

Lastly, the fourth version is a Java program that uses two-dimensional arrays and appropriate methods for the matrix operations. After testing the multiplication function on the two 10x10 matrices for this version, I obtained 0.000039594 seconds to finish performing matrix multiplication, which was the slowest result out of all four versions.

After finishing the implementations for all four versions of this programming project, I realized that the first C version which uses a regular stack dynamic two-dimensional array to represent the matrices was the fastest when performing the matrix multiplication on the two 10x10 matrices. Contrarily, the C++ version was the slowest.

Comments on the Timing of Matrix Multiplication on Increasing Size for Versions 2, 3, and 4:

As the time taken by versions 2, 3, and 4 to complete the multiplication operation on increasing NxN matrices was shown in page 6 of this report, I can conclude that the C version which used a pointer to a pointer and allocated memory in the heap for the dynamic two-dimensional matrices seemed to be the fastest as the matrix size increased. Furthermore, the C++ version that used a pointer to a pointer and overloaded operator functions seemed to be the slowest as the matrix size increased. Although I was able to anticipate that the C version would be the fastest, it was unexpected that Version 4 was faster than Version 3 as I assumed Java to be more "inefficient" compared to the other two languages.

Screenshot of the Output for Version 1:

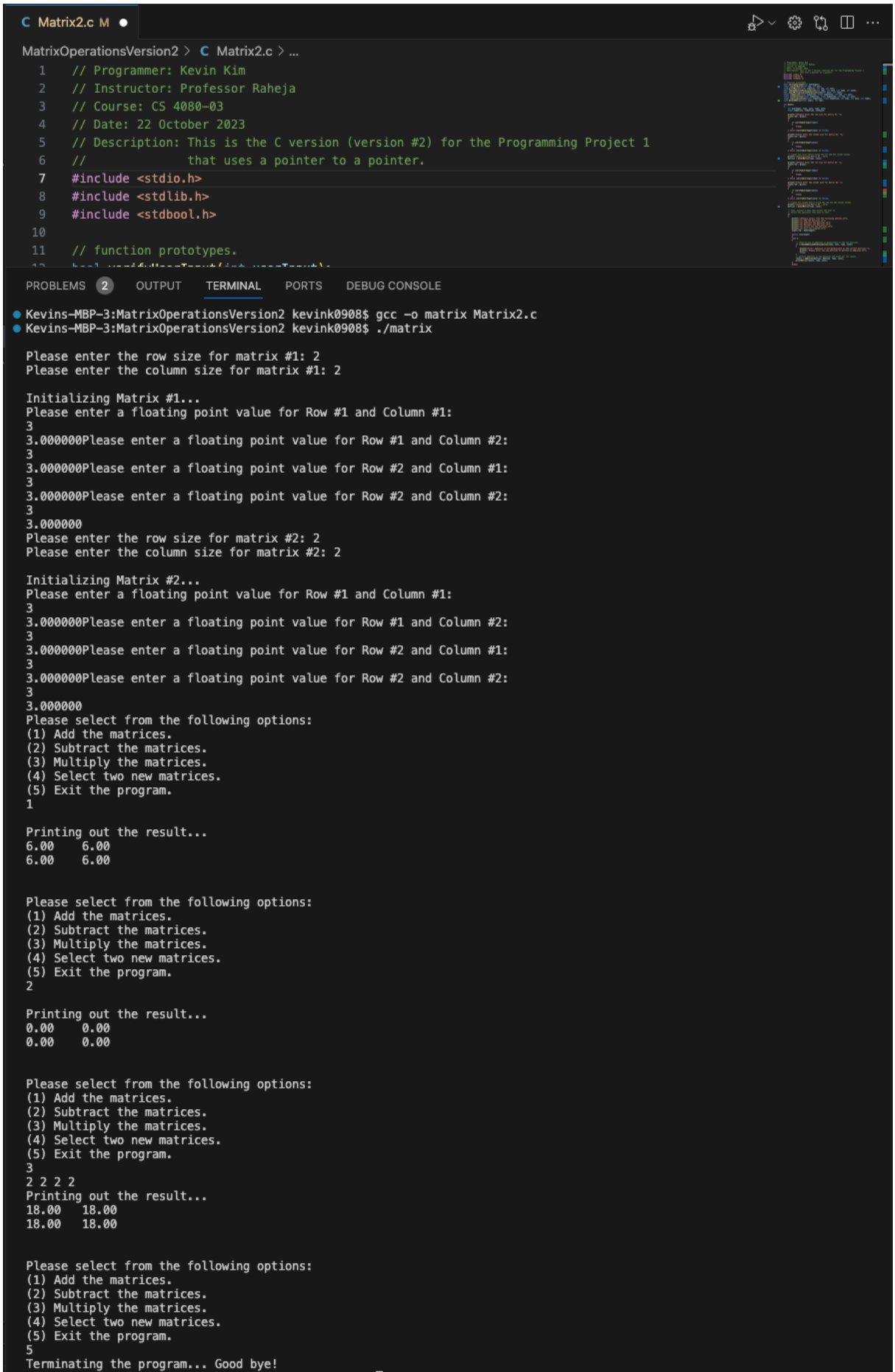
MatrixOperationsVersion1 > C Matrix1.c > main()

```
1   // Programmer: Kevin Kim
2   // Instructor: Professor Raheja
3   // Course: CS 4080-03
4   // Date: 22 October 2023
5   // Description: This is the C version (version #1) for the Programming Project 1
6   //              that uses a regular stack dynamic 2D array for representing matrices.
7   #include <stdio.h>
8   #include <stdlib.h>
9   #include <stdbool.h>
10
11  // function prototypes.
12  bool verifyUserInput(int userInput);
13  void enterMatrix(float matrix[100][100], int row, int col);
14  void printMatrix(float matrix[100][100], int row, int col);
15  bool checkAdditionPermissibility(int row1, int col1, int row2, int col2);
```

PROBLEMS 9 OUTPUT TERMINAL PORTS DEBUG CONSOLE

```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Kevins-MBP-3:Programming Project 1 kevink0908$ cd MatrixOperationsVersion1
Kevins-MBP-3:MatrixOperationsVersion1 kevink0908$ gcc -o matrix Matrix1.c
Kevins-MBP-3:MatrixOperationsVersion1 kevink0908$ ./matrix

Please enter the row size for matrix #1: 2
Please enter the column size for matrix #1: 3

Initializing Matrix #1...
Please enter a floating point value for Row #1 and Column #1:
0
Please enter a floating point value for Row #1 and Column #2:
1
Please enter a floating point value for Row #1 and Column #3:
2
Please enter a floating point value for Row #2 and Column #1:
3
Please enter a floating point value for Row #2 and Column #2:
2
Please enter a floating point value for Row #2 and Column #3:
1

Please enter the row size for matrix #2: 3
Please enter the column size for matrix #2: 2

Initializing Matrix #2...
Please enter a floating point value for Row #1 and Column #1:
2
Please enter a floating point value for Row #1 and Column #2:
1
Please enter a floating point value for Row #2 and Column #1:
4
Please enter a floating point value for Row #2 and Column #2:
3
Please enter a floating point value for Row #3 and Column #1:
6
Please enter a floating point value for Row #3 and Column #2:
5

Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
3

Printing out the result...
16.00   13.00
20.00   14.00


Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
1
Error: addition is not permissible on the current matrices. Please enter two new matrices to perform an addition.

Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
5
Terminating the program... Good bye!
```

Screenshot of the Output for Version 2:



C Matrix2.c M

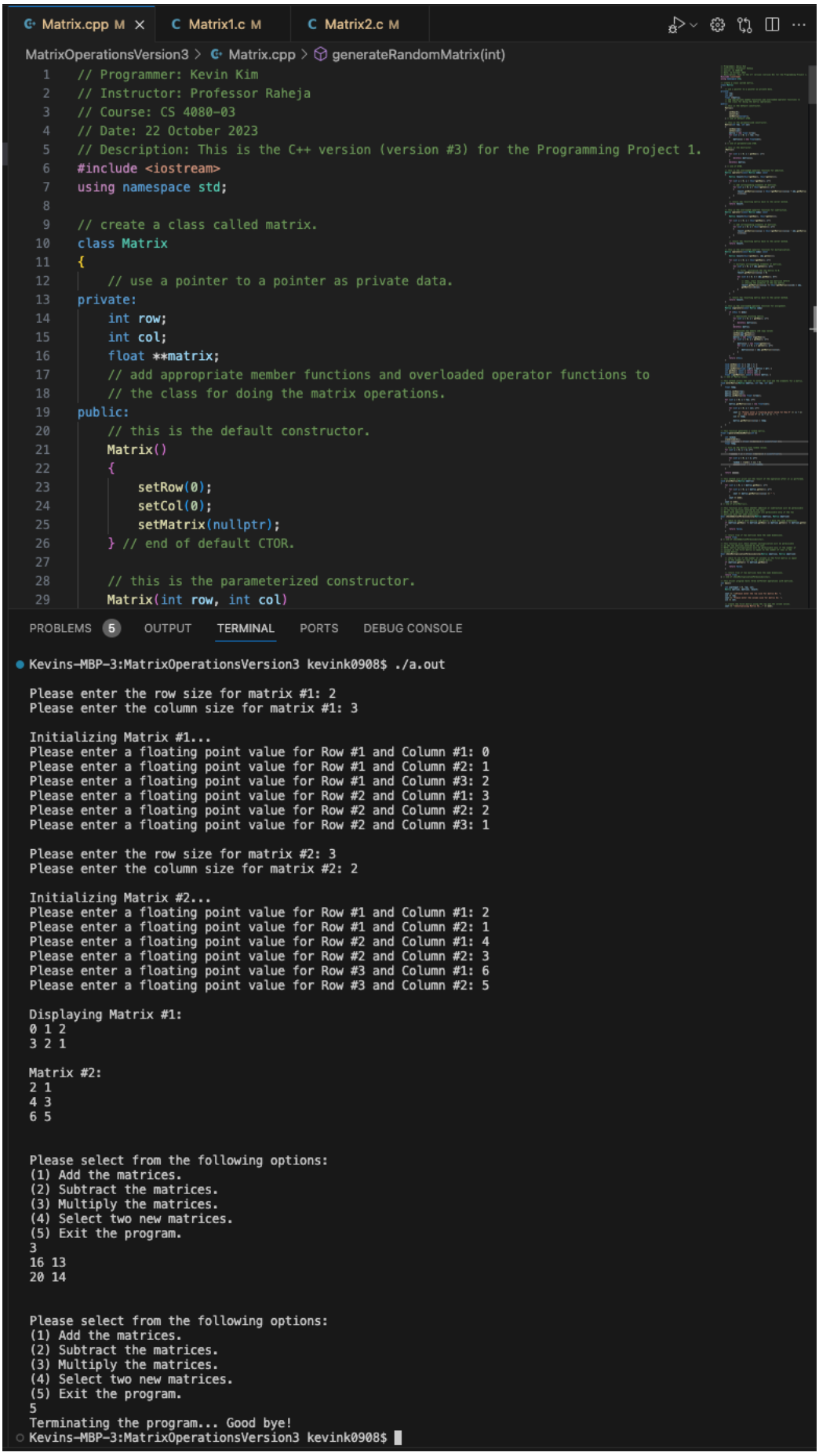
MatrixOperationsVersion2 > C Matrix2.c > ...

```
1   // Programmer: Kevin Kim
2   // Instructor: Professor Raheja
3   // Course: CS 4080-03
4   // Date: 22 October 2023
5   // Description: This is the C version (version #2) for the Programming Project 1
6   //              that uses a pointer to a pointer.
7   #include <stdio.h>
8   #include <stdlib.h>
9   #include <stdbool.h>
10
11  // function prototypes.
```

PROBLEMS 2 OUTPUT TERMINAL PORTS DEBUG CONSOLE

```
Kevins-MBP-3:MatrixOperationsVersion2 kevink0908$ gcc -o matrix Matrix2.c
Kevins-MBP-3:MatrixOperationsVersion2 kevink0908$ ./matrix

Please enter the row size for matrix #1: 2
Please enter the column size for matrix #1: 2

Initializing Matrix #1...
Please enter a floating point value for Row #1 and Column #1:
3
3.000000Please enter a floating point value for Row #1 and Column #2:
3
3.000000Please enter a floating point value for Row #2 and Column #1:
3
3.000000Please enter a floating point value for Row #2 and Column #2:
3
3.000000
Please enter the row size for matrix #2: 2
Please enter the column size for matrix #2: 2

Initializing Matrix #2...
Please enter a floating point value for Row #1 and Column #1:
3
3.000000Please enter a floating point value for Row #1 and Column #2:
3
3.000000Please enter a floating point value for Row #2 and Column #1:
3
3.000000Please enter a floating point value for Row #2 and Column #2:
3
3.000000
Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
1

Printing out the result...
6.00    6.00
6.00    6.00


Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
2

Printing out the result...
0.00    0.00
0.00    0.00


Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
3
2 2 2 2
Printing out the result...
18.00   18.00
18.00   18.00


Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
5
Terminating the program... Good bye!
```

Screenshot of the Output for Version 3:



Matrix.cpp M | Matrix1.c M | Matrix2.c M

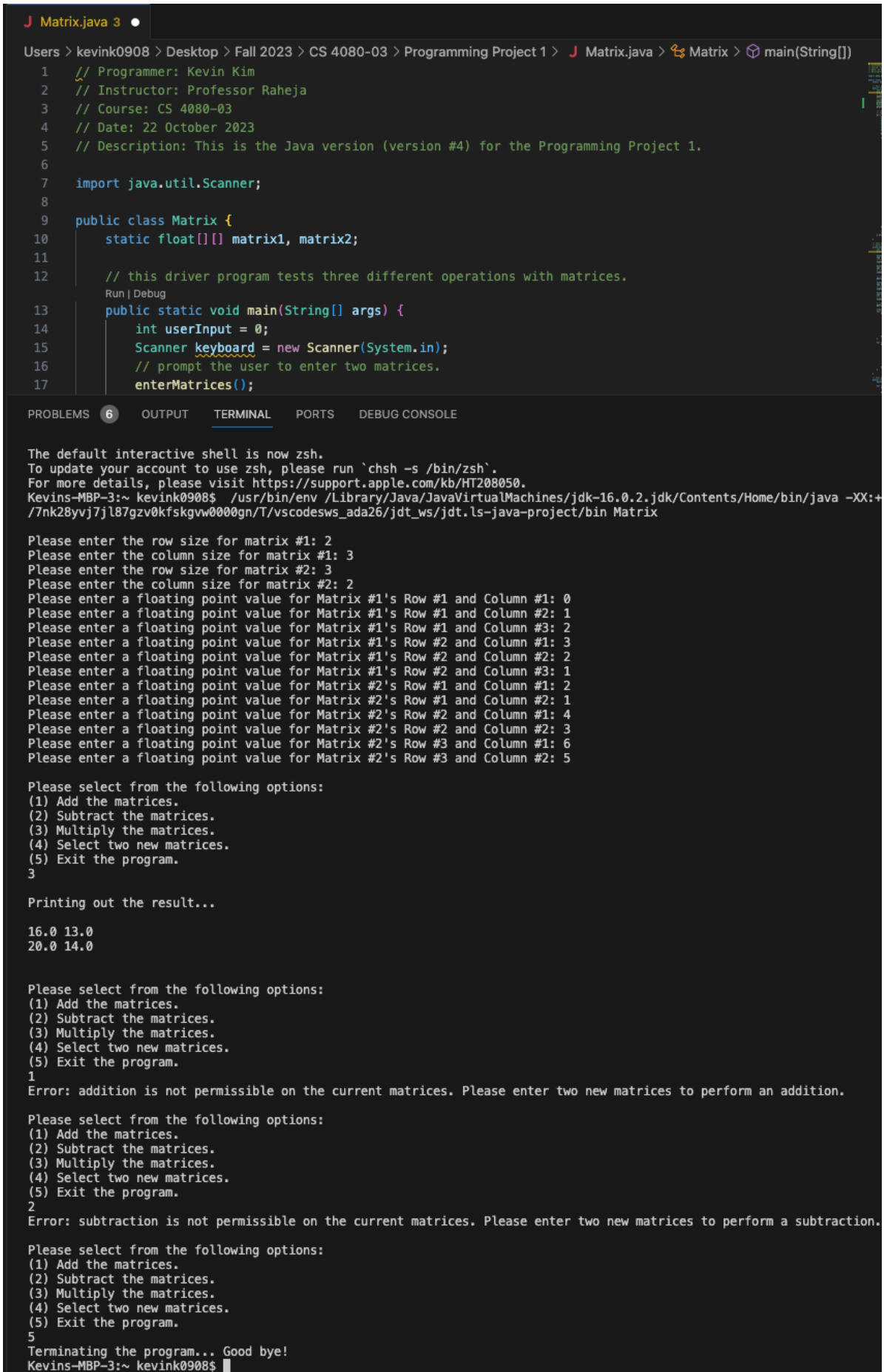
MatrixOperationsVersion3 > Matrix.cpp > generateRandomMatrix(int)

```
1   // Programmer: Kevin Kim
2   // Instructor: Professor Raheja
3   // Course: CS 4080-03
4   // Date: 22 October 2023
5   // Description: This is the C++ version (version #3) for the Programming Project 1.
6   #include <iostream>
7   using namespace std;
8
9   // create a class called matrix.
10  class Matrix
11  {
12      // use a pointer to a pointer as private data.
13  private:
14      int row;
15      int col;
16      float **matrix;
17      // add appropriate member functions and overloaded operator functions to
18      // the class for doing the matrix operations.
19  public:
20      // this is the default constructor.
21      Matrix()
22      {
23          setRow(0);
24          setCol(0);
25          setMatrix(nullptr);
26      } // end of default CTOR.
27
28      // this is the parameterized constructor.
29      Matrix(int row, int col)
```

PROBLEMS 5 | OUTPUT | TERMINAL | PORTS | DEBUG CONSOLE

```
Kevins-MBP-3:MatrixOperationsVersion3 kevink0908$ ./a.out

Please enter the row size for matrix #1: 2
Please enter the column size for matrix #1: 3

Initializing Matrix #1...
Please enter a floating point value for Row #1 and Column #1: 0
Please enter a floating point value for Row #1 and Column #2: 1
Please enter a floating point value for Row #1 and Column #3: 2
Please enter a floating point value for Row #2 and Column #1: 3
Please enter a floating point value for Row #2 and Column #2: 2
Please enter a floating point value for Row #2 and Column #3: 1

Please enter the row size for matrix #2: 3
Please enter the column size for matrix #2: 2

Initializing Matrix #2...
Please enter a floating point value for Row #1 and Column #1: 2
Please enter a floating point value for Row #1 and Column #2: 1
Please enter a floating point value for Row #2 and Column #1: 4
Please enter a floating point value for Row #2 and Column #2: 3
Please enter a floating point value for Row #3 and Column #1: 6
Please enter a floating point value for Row #3 and Column #2: 5

Displaying Matrix #1:
0 1 2
3 2 1

Matrix #2:
2 1
4 3
6 5


Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
3
16 13
20 14


Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
5
Terminating the program... Good bye!
Kevins-MBP-3:MatrixOperationsVersion3 kevink0908$
```

Screenshot of the Output for Version 4:



Matrix.java 3

Users > kevink0908 > Desktop > Fall 2023 > CS 4080-03 > Programming Project 1 > Matrix.java > Matrix > main(String[])

```
1   // Programmer: Kevin Kim
2   // Instructor: Professor Raheja
3   // Course: CS 4080-03
4   // Date: 22 October 2023
5   // Description: This is the Java version (version #4) for the Programming Project 1.
6
7   import java.util.Scanner;
8
9   public class Matrix {
10      static float[][] matrix1, matrix2;
11
12      // this driver program tests three different operations with matrices.
        Run | Debug
13      public static void main(String[] args) {
14          int userInput = 0;
15          Scanner keyboard = new Scanner(System.in);
16          // prompt the user to enter two matrices.
17          enterMatrices();
```

PROBLEMS 6 OUTPUT TERMINAL PORTS DEBUG CONSOLE

```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Kevins-MBP-3:~ kevink0908$  /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-16.0.2.jdk/Contents/Home/bin/java -XX:+
/7nk28yvj7jl87gzv0kfskgvw0000gn/T/vscodesws_ada26/jdt_ws/jdt.ls-java-project/bin Matrix

Please enter the row size for matrix #1: 2
Please enter the column size for matrix #1: 3
Please enter the row size for matrix #2: 3
Please enter the column size for matrix #2: 2
Please enter a floating point value for Matrix #1's Row #1 and Column #1: 0
Please enter a floating point value for Matrix #1's Row #1 and Column #2: 1
Please enter a floating point value for Matrix #1's Row #1 and Column #3: 2
Please enter a floating point value for Matrix #1's Row #2 and Column #1: 3
Please enter a floating point value for Matrix #1's Row #2 and Column #2: 2
Please enter a floating point value for Matrix #1's Row #2 and Column #3: 1
Please enter a floating point value for Matrix #2's Row #1 and Column #1: 2
Please enter a floating point value for Matrix #2's Row #1 and Column #2: 1
Please enter a floating point value for Matrix #2's Row #2 and Column #1: 4
Please enter a floating point value for Matrix #2's Row #2 and Column #2: 3
Please enter a floating point value for Matrix #2's Row #3 and Column #1: 6
Please enter a floating point value for Matrix #2's Row #3 and Column #2: 5

Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
3

Printing out the result...

16.0 13.0
20.0 14.0


Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
1
Error: addition is not permissible on the current matrices. Please enter two new matrices to perform an addition.

Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
2
Error: subtraction is not permissible on the current matrices. Please enter two new matrices to perform a subtraction.

Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
5
Terminating the program... Good bye!
Kevins-MBP-3:~ kevink0908$
```

Table for Time Taken to Complete Matrix Multiplication on Two 10x10 Matrices:

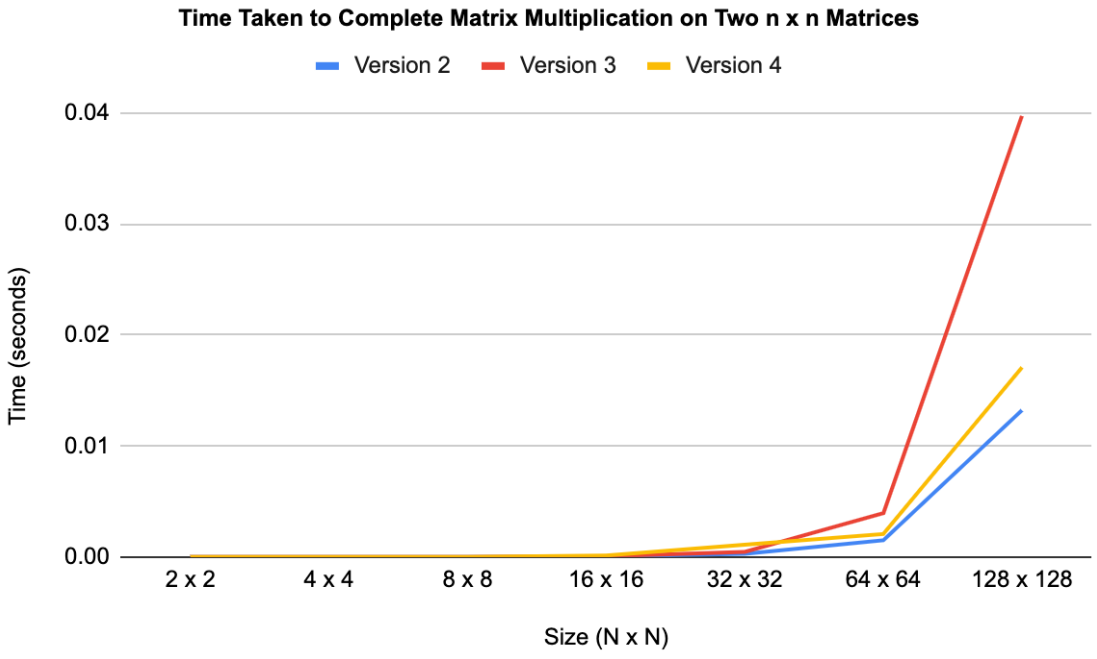
Version Number	Time (seconds)
Version 1 (C)	0.000013
Version 2 (C)	0.000014
Version 3 (C++)	0.000079
Version 4 (Java)	0.000039594

NOTE: Please see additional screenshots near the end of this report for screenshots that contain the time spent on performing matrix multiplication on two 10x10 matrices using different versions.

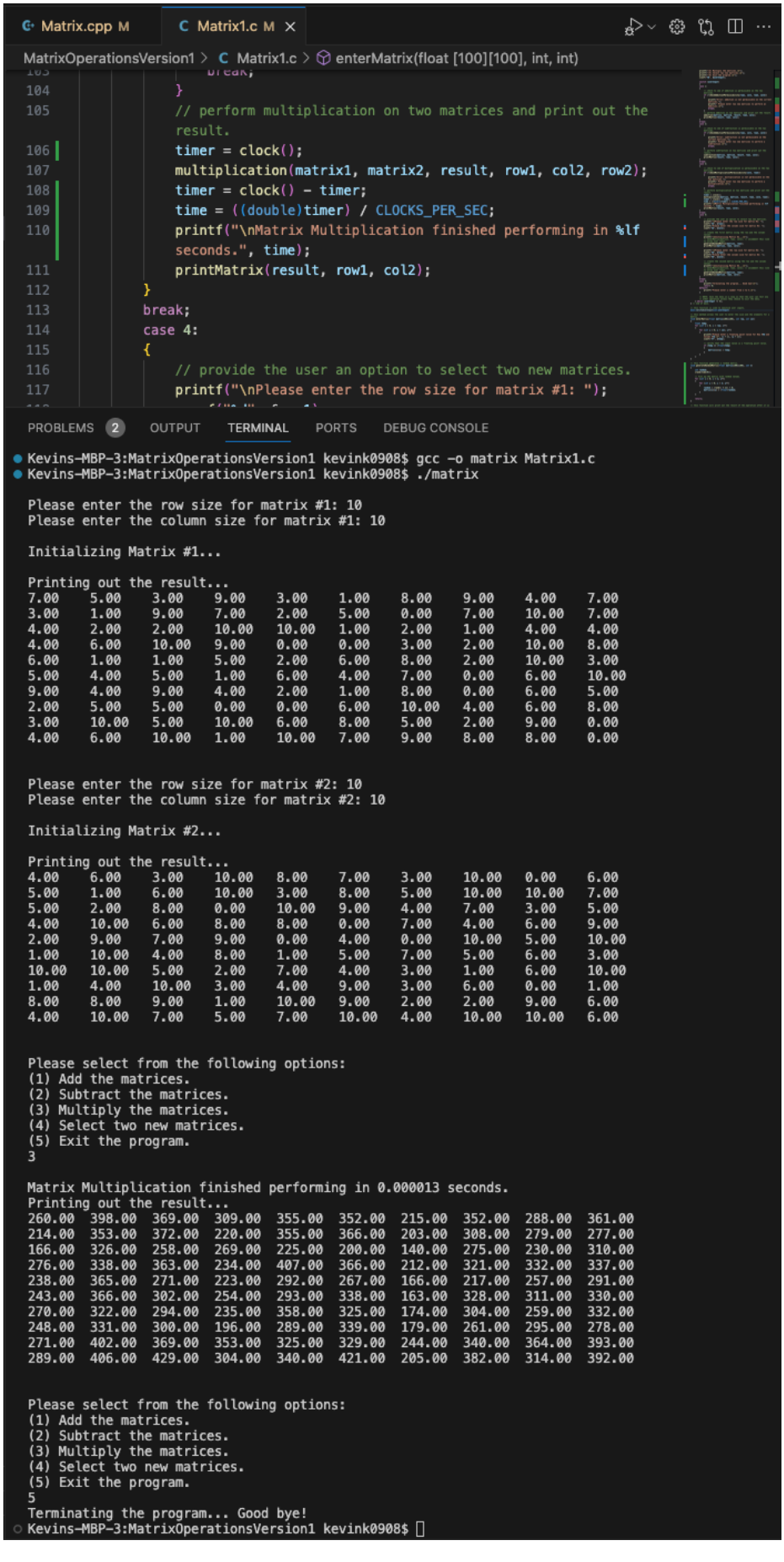
Table for Time Taken to Complete Matrix Multiplication on Two n x n Matrices:

Matrix Size	Version 2 (sec)	Version 3 (sec)	Version 4 (sec)
2 x 2	0.000006	0.000009	0.000004383
4 x 4	0.000007	0.000016	0.000008586
8 x 8	0.000010	0.000021	0.000022469
16 x 16	0.000057	0.000074	0.000141371
32 x 32	0.000295	0.000473	0.001111067
64 x 64	0.001510	0.003956	0.002073391
128 x 128	0.013216	0.039711	0.017077273

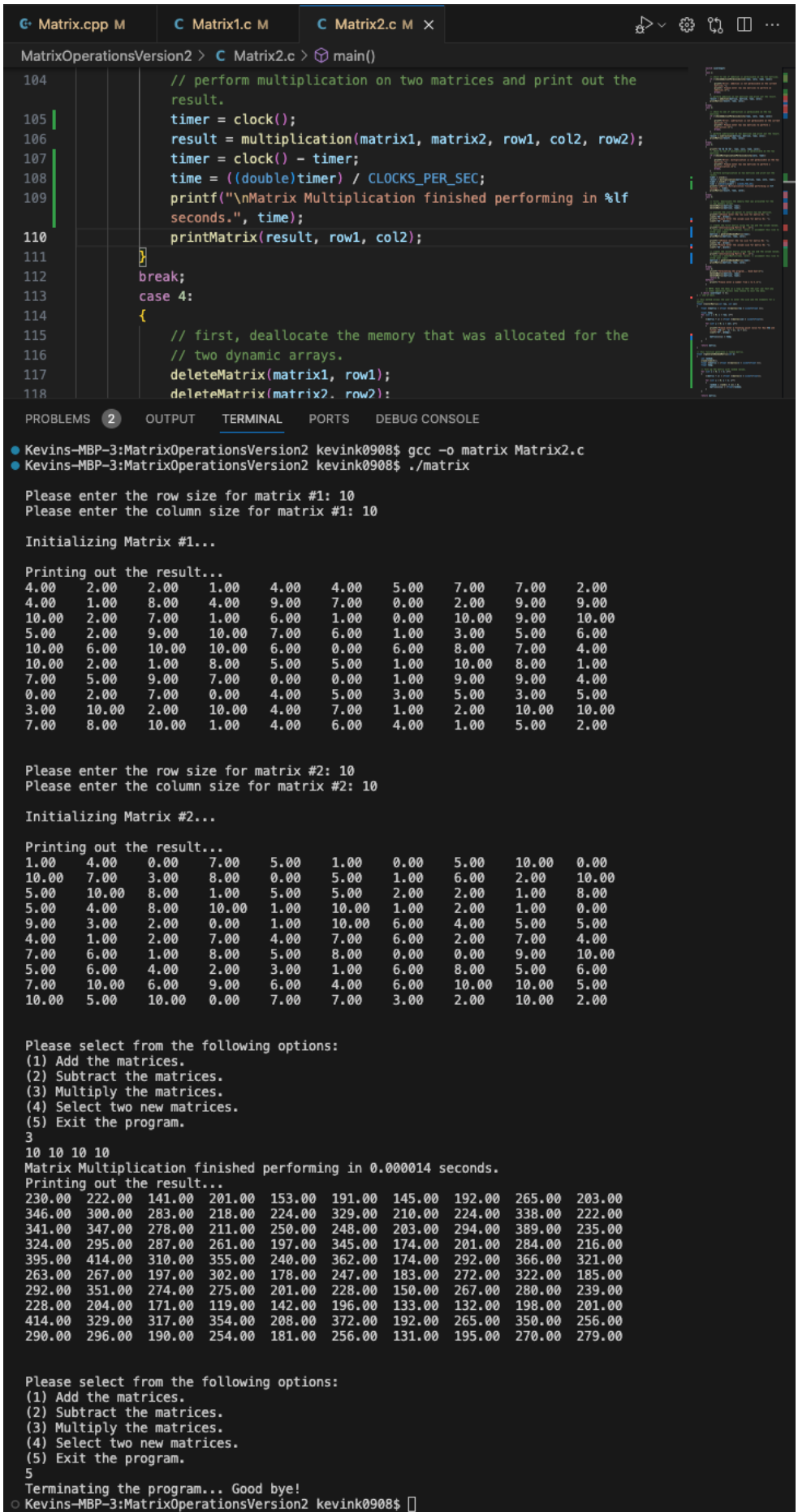
Graph for Time Taken to Complete Matrix Multiplication on Two n x n Matrices:



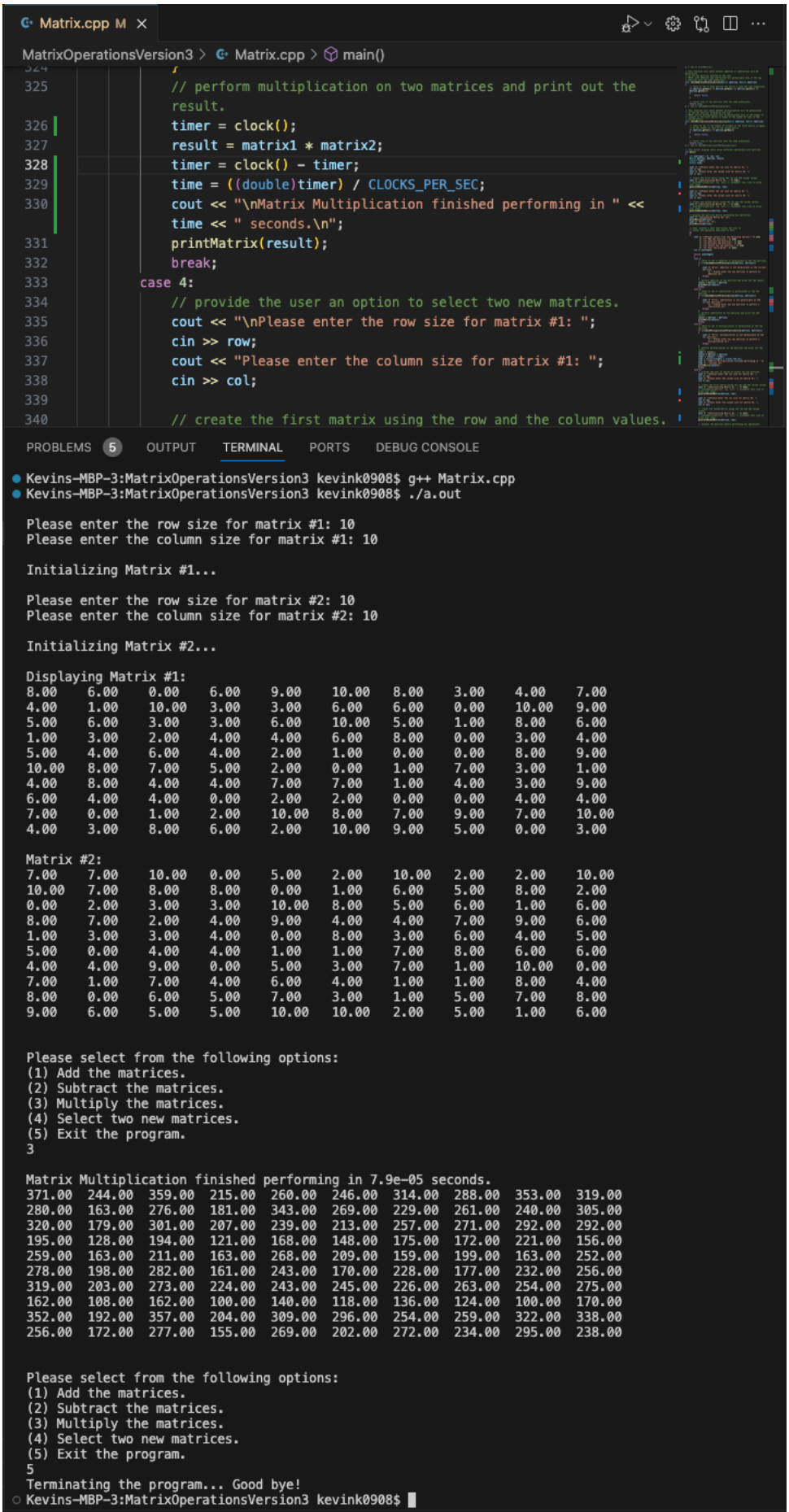
Additional Screenshot #1:



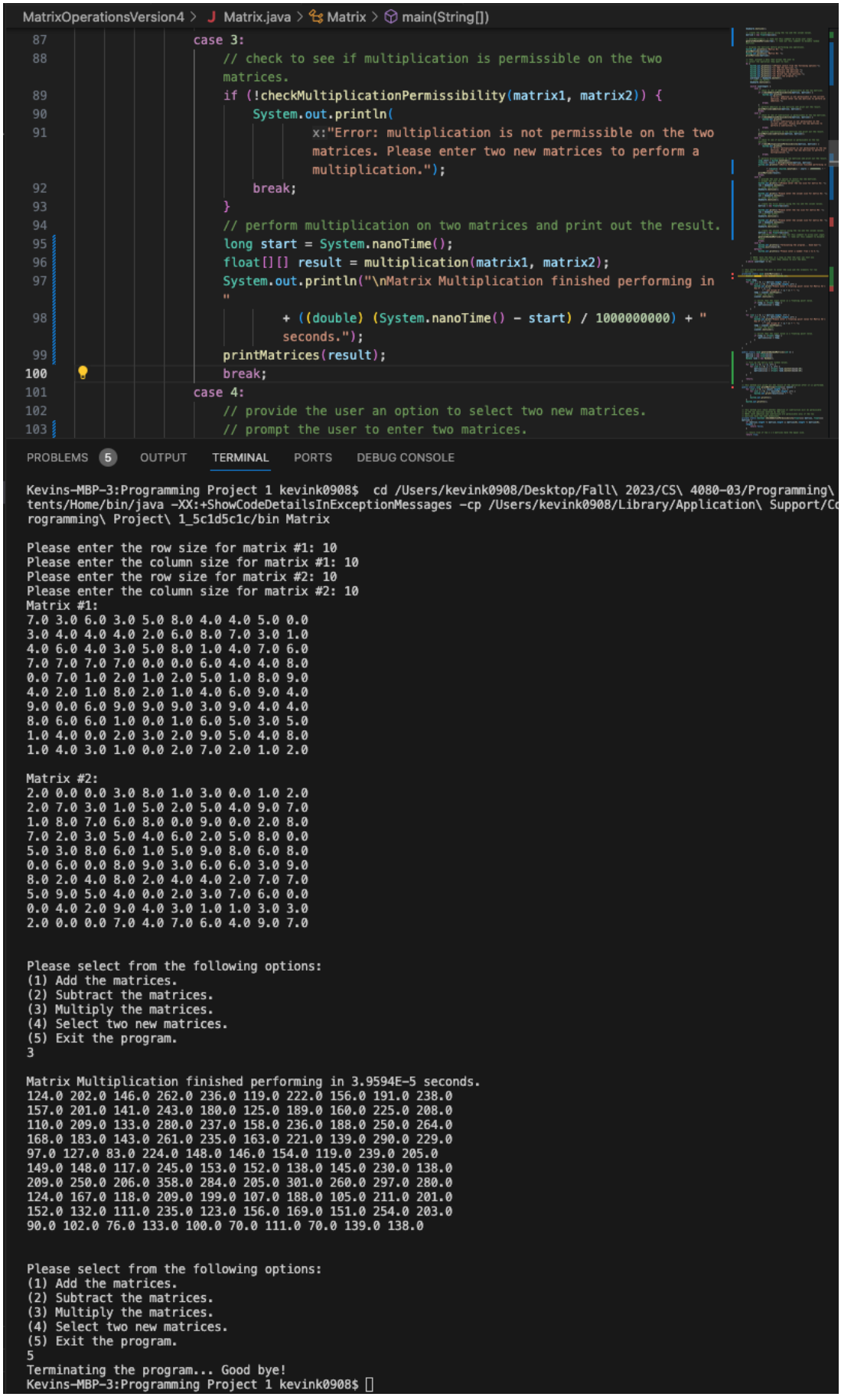
Additional Screenshot #2:



Additional Screenshot #3:



Additional Screenshot #4:



MatrixOperationsVersion4 > Matrix.java > Matrix > main(String[])

```
87                  case 3:
88                      // check to see if multiplication is permissible on the two
                        matrices.
89                      if (!checkMultiplicationPermissibility(matrix1, matrix2)) {
90                          System.out.println(
91                                  x:"Error: multiplication is not permissible on the two
                                    matrices. Please enter two new matrices to perform a
                                    multiplication.");
92                          break;
93                      }
94                      // perform multiplication on two matrices and print out the result.
95                      long start = System.nanoTime();
96                      float[][] result = multiplication(matrix1, matrix2);
97                      System.out.println("\nMatrix Multiplication finished performing in
                        "
98                              + ((double) (System.nanoTime() - start) / 1000000000) + "
                                seconds.");
99                      printMatrices(result);
100                     break;
101                 case 4:
102                     // provide the user an option to select two new matrices.
103                     // prompt the user to enter two matrices.
```

PROBLEMS 5 OUTPUT TERMINAL PORTS DEBUG CONSOLE

```
Kevins-MBP-3:Programming Project 1 kevink0908$  cd /Users/kevink0908/Desktop/Fall\ 2023/CS\ 4080-03/Programming\
tents/Home/bin/java -XX:+ShowCodeDetailsInExceptionMessages -cp /Users/kevink0908/Library/Application\ Support/Co
rogramming\ Project\ 1_5c1d5c1c/bin Matrix

Please enter the row size for matrix #1: 10
Please enter the column size for matrix #1: 10
Please enter the row size for matrix #2: 10
Please enter the column size for matrix #2: 10
Matrix #1:
7.0 3.0 6.0 3.0 5.0 8.0 4.0 4.0 5.0 0.0
3.0 4.0 4.0 4.0 2.0 6.0 8.0 7.0 3.0 1.0
4.0 6.0 4.0 3.0 5.0 8.0 1.0 4.0 7.0 6.0
7.0 7.0 7.0 7.0 0.0 0.0 6.0 4.0 4.0 8.0
0.0 7.0 1.0 2.0 1.0 2.0 5.0 1.0 8.0 9.0
4.0 2.0 1.0 8.0 2.0 1.0 4.0 6.0 9.0 4.0
9.0 0.0 6.0 9.0 9.0 9.0 3.0 9.0 4.0 4.0
8.0 6.0 6.0 1.0 0.0 1.0 6.0 5.0 3.0 5.0
1.0 4.0 0.0 2.0 3.0 2.0 9.0 5.0 4.0 8.0
1.0 4.0 3.0 1.0 0.0 2.0 7.0 2.0 1.0 2.0

Matrix #2:
2.0 0.0 0.0 3.0 8.0 1.0 3.0 0.0 1.0 2.0
2.0 7.0 3.0 1.0 5.0 2.0 5.0 4.0 9.0 7.0
1.0 8.0 7.0 6.0 8.0 0.0 9.0 0.0 2.0 8.0
7.0 2.0 3.0 5.0 4.0 6.0 2.0 5.0 8.0 0.0
5.0 3.0 8.0 6.0 1.0 5.0 9.0 8.0 6.0 8.0
0.0 6.0 0.0 8.0 9.0 3.0 6.0 6.0 3.0 9.0
8.0 2.0 4.0 8.0 2.0 4.0 4.0 2.0 7.0 7.0
5.0 9.0 5.0 4.0 0.0 2.0 3.0 7.0 6.0 0.0
0.0 4.0 2.0 9.0 4.0 3.0 1.0 1.0 3.0 3.0
2.0 0.0 0.0 7.0 4.0 7.0 6.0 4.0 9.0 7.0


Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
3

Matrix Multiplication finished performing in 3.9594E-5 seconds.
124.0 202.0 146.0 262.0 236.0 119.0 222.0 156.0 191.0 238.0
157.0 201.0 141.0 243.0 180.0 125.0 189.0 160.0 225.0 208.0
110.0 209.0 133.0 280.0 237.0 158.0 236.0 188.0 250.0 264.0
168.0 183.0 143.0 261.0 235.0 163.0 221.0 139.0 290.0 229.0
97.0 127.0 83.0 224.0 148.0 146.0 154.0 119.0 239.0 205.0
149.0 148.0 117.0 245.0 153.0 152.0 138.0 145.0 230.0 138.0
209.0 250.0 206.0 358.0 284.0 205.0 301.0 260.0 297.0 280.0
124.0 167.0 118.0 209.0 199.0 107.0 188.0 105.0 211.0 201.0
152.0 132.0 111.0 235.0 123.0 156.0 169.0 151.0 254.0 203.0
90.0 102.0 76.0 133.0 100.0 70.0 111.0 70.0 139.0 138.0


Please select from the following options:
(1) Add the matrices.
(2) Subtract the matrices.
(3) Multiply the matrices.
(4) Select two new matrices.
(5) Exit the program.
5
Terminating the program... Good bye!
Kevins-MBP-3:Programming Project 1 kevink0908$
```