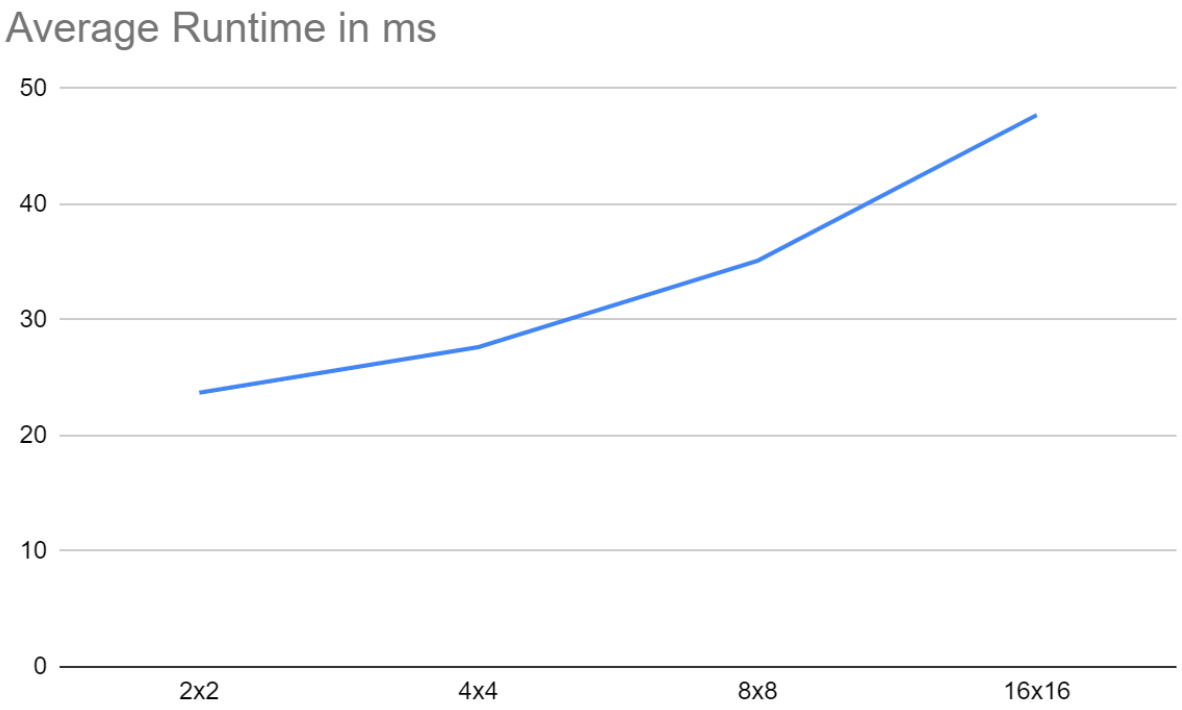
In the following problem, we were given code and told to compute the max, min and average of a NxN matrix array where if each row was added it could cause overflow. We were limited to using Int and floats for calculations, and Double/Long to log the time. This was a practice in Concurrency as well as primitive overflowing and how to get around it.

In my code I managed to compute the min, max and average just like we were told, but what I often found was that the average was off by around 20.25 each time. I believe this has to do with how the float is implemented in java, and how it does not cover every single fractional value, rather, (depending on how it is implemented), it tends to lose some value the further out one goes from [1, -1].

As can be seen clearly in the graph, this line seems to be in between exponential and linear in runtime. This was expected as the algorithm in the program would be $O(n^2)$ if it were implemented on one thread. We also know that by using multiple threads we should get faster processing results as long as there is not a lot of blocking (such as with IO). Therefore, it stands to reason that by using multiple threads we are able to finish a task in a faster amount of time. The standard deviation only seemed a bit high for the 16x16 array so maybe it was affected by background processes.



Array	Standard Deviation
2x2	0.5490167602
4x4	0.3614018774
8x8	0.4522893067
16x16	1.639543678

Code
// // // from http://www.letmeknows.com/2017/04/24/wait-for-threads-to-finish-java/ // // This is a very small set up to get people started on using threads // // // // // Adopted by Shaun Cooper // last updated November 2020 // // We need static variable pointers in the main class so that

```
//  we can share these values with the threads.
//  the threads are address separate from us, so we need to share
//  pointers to the objects that we are sharing and updating

/*
Name: Marco Salazar
Date: 11/13/2020
Assignment: Programming assignment Concurrency
Problem: To experiment with concurrency and creating threads. This does not deal with
deadlock or semaphores, but
introduces us to basics of concurrency, as well as overflowwing of values.

There is the length that the multidimensional matrix will be.
And the output is the max, min, average, and time it took to compute those values.

This code is adapted from Shaun Cooper and everything else added was original code.
*/

import java.util.ArrayList;
import java.util.*;

public class MythreadTest {

   private static ArrayList<Thread> arrThreads = new ArrayList<Thread>();

   // we use static variables to help us connect the threads
   // to a common block
   public static int[][] A;
   //These variables hold the max min and avg for each thread so that we do not have to deal
with race conditions.
   public static int[] max;
   public static int[] min;
   public static int[] sum;

   //main entry point for the process

   public static void main(String[] args) {
      try {
                  int size = Integer.parseInt(args[0]);
                  // create the array from input
                  A = new int[size][size];
      max = new int[size];
      min = new int[size];
      sum = new int[size];

                  //
```

```
                // fill array with random valuesvalues
                //
        for(int i = 0; i < A.length; i++){
            for(int j = 0; j < A[0].length; j++){
                //Essentially the sum of these integers should be Integer.MAX_VALUE if they are
the largest ones.
                A[i][j] = (int) ( Math.random()*(Math.pow(2,32-size) - Math.pow(2, 31-size)) +
Math.pow(2, 31-size));
            }
        }

        //start time.
        long startTime = System.nanoTime();

                // create N threads to work on each row
            for (int i = 0; i < size; i++)
            {
                Thread T1 = new Thread(new ThreadTest(i));
                T1.start();             // standard thread start
                arrThreads.add(T1);
            }

                // wait for each thread to complete
            for (int i = 0; i < arrThreads.size(); i++)
            {
                arrThreads.get(i).join();
            }

                // all the threads are done
                // do final calculations
        // ensure that all numbers will be replaced with the first value of their respective arrays.
        int ma = Integer.MIN_VALUE;
        int mi = Integer.MAX_VALUE;
        float avg =0;
        for(int i = 0; i < size; i++){
            if(max[i] > ma) ma = max[i];
            if(min[i] < mi) mi = min[i];
            //addition of divided sums to keep it from overflowing.
            avg = avg + (sum[i] / ((float)(size*size)));
        }

        //Calculate the time and print out the results.
        long endTime = System.nanoTime();
        long timeElapsed = endTime - startTime;

        System.out.println("max: " + ma);
```

```
        System.out.println("min: " + mi);
        System.out.printf("average: %f\n", avg);
        System.out.println("Time is (ms): " + (timeElapsed /1000000.0));

            //This for loop will not stop execution of any thread,
            //only it will come out when all thread are executed

            System.out.println("Main thread exiting ");
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}

// each thread should access its row based on "ind"
// and leave results I would suggest in a static array that you need
// to create  in MythreadTest

class ThreadTest implements Runnable {
    private int i;
    private int ma;
    private int mi;
    private int su;
    ThreadTest(int ind)
    {
      i = ind;
      ma = Integer.MIN_VALUE;
      mi = Integer.MAX_VALUE;
      su = 0;
    }
    public void run() {
      try
      {

          System.out.println("Thread is started " + i);
          for(int col = 0; col < MythreadTest.A.length; col++){
              ma = Math.max(ma, MythreadTest.A[i][col]);
              mi = Math.min(mi, MythreadTest.A[i][col]);
              su += MythreadTest.A[i][col];
          }
          MythreadTest.max[i] = ma;
          MythreadTest.min[i] = mi;
          MythreadTest.sum[i] = su;
          System.out.println("Thread is exiting " + i);
      }
      catch (Exception e) {
```

```
            System.out.println(e.getMessage());
        }     }
}
```

Output

```
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 2
Thread is started 1
Thread is started 0
Thread is exiting 1
Thread is exiting 0
max: 1036627250
min: 603809053
average: 919791552.000000
Time is (ms): 23.7759
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 2
Thread is started 0
Thread is started 1
Thread is exiting 0
Thread is exiting 1
max: 833509096
min: 704068575
average: 790806592.000000
Time is (ms): 23.0642
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 2
Thread is started 1
Thread is started 0
Thread is exiting 1
Thread is exiting 0
max: 1018050232
min: 558808540
average: 780228224.000000
Time is (ms): 23.6216
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 2
Thread is started 1
Thread is started 0
Thread is exiting 1
Thread is exiting 0
max: 1046449263
min: 718448080
average: 893939584.000000
Time is (ms): 23.5201
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 2
Thread is started 1
Thread is started 0
Thread is exiting 1
Thread is exiting 0
max: 756127211
min: 634918307
average: 691164480.000000
Time is (ms): 24.5695
Main thread exiting
```

```
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 4
Thread is started 3
Thread is started 0
Thread is started 1
Thread is exiting 3
Thread is started 2
Thread is exiting 0
Thread is exiting 1
Thread is exiting 2
max: 254150635
min: 150750124
average: 198003488.000000
Time is (ms): 27.7094
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 4
Thread is started 2
Thread is started 3
Thread is started 0
Thread is started 1
Thread is exiting 3
Thread is exiting 0
Thread is exiting 2
Thread is exiting 1
max: 238016819
min: 144073091
average: 199801104.000000
Time is (ms): 27.6537
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 4
Thread is started 2
Thread is started 3
Thread is started 0
Thread is started 1
Thread is exiting 3
Thread is exiting 0
Thread is exiting 2
Thread is exiting 1
max: 267046885
min: 145018213
average: 216478048.000000
Time is (ms): 27.0435
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 4
Thread is started 1
Thread is started 2
Thread is started 3
Thread is started 0
Thread is exiting 2
Thread is exiting 3
Thread is exiting 1
Thread is exiting 0
max: 267815106
min: 146778144
average: 207946112.000000
Time is (ms): 27.9582
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 4
Thread is started 3
Thread is started 1
Thread is started 2
Thread is started 0
Thread is exiting 1
Thread is exiting 2
Thread is exiting 3
Thread is exiting 0
max: 257873822
min: 140262558
average: 189357856.000000
Time is (ms): 27.8868
Main thread exiting
```

```
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 8
Thread is started 7Thread is started 2Thread is started 6Thread is started 1Thread is started 4Thread is started 3Threa
 is exiting 3Thread is started 5Thread is started 0Thread is exiting 5Thread is exiting 4Thread is exiting 2Thread is e
iting 1Thread is exiting 6Thread is exiting 7Thread is exiting 0
max: 16730813
min: 8617644
average: 13166535.000000
Time is (ms): 34.8303
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 8
Thread is started 7Thread is started 1Thread is started 0Thread is started 4Thread is started 3Thread is exiting 3Threa
 is started 2Thread is started 5Thread is started 6Thread is exiting 5Thread is exiting 2Thread is exiting 1Thread is e
iting 4Thread is exiting 0Thread is exiting 7Thread is exiting 6
max: 16631508
min: 8449781
average: 13177074.000000
Time is (ms): 35.3003
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 8
Thread is started 1Thread is started 5Thread is exiting 1Thread is started 3Thread is started 4Thread is started 6Threa
 is started 7Thread is started 2Thread is started 0Thread is exiting 2Thread is exiting 7Thread is exiting 6Thread is e
iting 4Thread is exiting 3Thread is exiting 5Thread is exiting 0
max: 16684147
min: 8393912
average: 11802582.000000
Time is (ms): 34.47
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 8
Thread is started 3Thread is started 4Thread is started 5Thread is started 6Thread is started 1Thread is exiting 4Threa
 is started 0Thread is started 2Thread is started 7Thread is exiting 2Thread is exiting 0Thread is exiting 1Thread is e
iting 6Thread is exiting 5Thread is exiting 3Thread is exiting 7
max: 16727998
min: 8463083
average: 12426448.000000
Time is (ms): 35.5923
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 8
Thread is started 2Thread is started 4Thread is started 5Thread is started 3Thread is exiting 3Thread is started 0Threa
 is started 1Thread is started 7Thread is started 6Thread is exiting 7Thread is exiting 1Thread is exiting 0Thread is e
iting 4Thread is exiting 5Thread is exiting 2Thread is exiting 6
max: 16687117
min: 8484203
average: 12840603.000000
Time is (ms): 35.3582
Main thread exiting
```

```
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 16
Thread is started 13Thread is started 4Thread is started 12Thread is exiting 13Thread is started 8Thread is started 15
read is started 1Thread is started 9Thread is started 11Thread is started 2Thread is started 14Thread is started 5Thre
 is started 6Thread is started 10Thread is started 0Thread is started 3Thread is started 7Thread is exiting 3Thread is
xiting 0Thread is exiting 10Thread is exiting 6Thread is exiting 5Thread is exiting 14Thread is exiting 2Thread is exi
ng 11Thread is exiting 9Thread is exiting 1Thread is exiting 15Thread is exiting 8Thread is exiting 4Thread is exiting
2Thread is exiting 7
max: 65487
min: 32922
average: 49422.886719
Time is (ms): 46.821
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 16
Thread is started 4Thread is started 7Thread is started 8Thread is started 14Thread is started 3Thread is exiting 8Thr
d is started 9Thread is started 6Thread is started 10Thread is started 15Thread is started 0Thread is started 12Thread
s started 1Thread is started 2Thread is started 5Thread is started 11Thread is started 13Thread is exiting 11Thread is
xiting 5Thread is exiting 2Thread is exiting 1Thread is exiting 12Thread is exiting 0Thread is exiting 15Thread is exi
ng 10Thread is exiting 6Thread is exiting 9Thread is exiting 3Thread is exiting 7Thread is exiting 14Thread is exiting
Thread is exiting 13
max: 65506
min: 32854
average: 48748.968750
Time is (ms): 47.3358
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 16
Thread is started 15Thread is started 10Thread is started 12Thread is started 8Thread is started 5Thread is exiting 10
read is started 0Thread is exiting 0Thread is started 11Thread is started 13Thread is started 7Thread is started 6Thre
 is started 9Thread is started 2Thread is started 3Thread is started 14Thread is started 1Thread is started 4Thread is
xiting 1Thread is exiting 14Thread is exiting 3Thread is exiting 2Thread is exiting 9Thread is exiting 6Thread is exit
g 7Thread is exiting 13Thread is exiting 11Thread is exiting 12Thread is exiting 5Thread is exiting 8Thread is exiting
5Thread is exiting 4
max: 65493
min: 32786
average: 49462.636719
Time is (ms): 48.4224
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 16
Thread is started 13Thread is started 5Thread is started 3Thread is started 9Thread is started 8Thread is started 15Th
ad is started 11Thread is started 14Thread is started 1Thread is started 6Thread is started 2Thread is started 10Threa
is started 0Thread is started 7Thread is started 4Thread is started 12Thread is exiting 4Thread is exiting 7Thread is
iting 0Thread is exiting 10Thread is exiting 2Thread is exiting 6Thread is exiting 1Thread is exiting 14Thread is exit
g 11Thread is exiting 15Thread is exiting 3Thread is exiting 8Thread is exiting 5Thread is exiting 9Thread is exiting
Thread is exiting 12
max: 65513
min: 32790
average: 49302.277344
Time is (ms): 50.125
Main thread exiting
marco@DESKTOP-625N2SQ:/mnt/c/schoollinux/cs471/concurrency$ java MythreadTest 16
Thread is started 1Thread is started 4Thread is started 8Thread is started 7Thread is started 15Thread is exiting 8Thr
d is started 6Thread is started 9Thread is started 10Thread is started 13Thread is started 5Thread is started 0Thread
 started 12Thread is started 3Thread is started 2Thread is exiting 2Thread is started 11Thread is started 14Thread is
iting 11Thread is exiting 3Thread is exiting 12Thread is exiting 0Thread is exiting 5Thread is exiting 13Thread is exi
ng 10Thread is exiting 9Thread is exiting 6Thread is exiting 15Thread is exiting 7Thread is exiting 4Thread is exiting
Thread is exiting 14
max: 65520
min: 32841
average: 49150.847656
Time is (ms): 45.8439
Main thread exiting
```