Project 0 Writeup

1. **Machine**

   I ran this program on the flip3 engr server.

2. **Performance**

   I had very inconsistent results for each execution of the code for small values of ARRAYSIZE (1000 - 10000). Only after I tried the code with arrays of length 100 million did it produce consisent results. Moreover, I noticed that my speedup never exceeded 2, which was suspicious. I did some digging and found that I did not have four cores available, so it was not actually delegating four threads for the code. After I ran it on the school server, I had much better results for the speedup.

3. **Times**

   For my code, I created three integer arrays each with 10,000,000 items. I ran the benchmarks 100 times and kept only the peak performance for each. I also calculated the average performance to ensure that my result was not an outlier. The output of my code is below:

   ```
   Using 1 threads
   Peak Performance =  2183.41 Mega Mults/Sec
   Avg Performance =  2032.73 Mega Mults/Sec
   Using 4 threads
   Peak Performance =  7020.86 Mega Mults/Sec
   Avg Performance =  6615.97 Mega Mults/Sec
   ```

4. **Speedup**

   The above unit of measurement is in terms of how many times the parallelized part can be executed in 1000000 seconds: Mega Mults / Sec = ARRAYSIZE / (1000000 * (time1 - time0)). Since the speedup is equal to the time for single threaded run divided by the time for the nth threaded run, we can calculate this result by dividing Mega Mults for the four-threaded divided by the Mega Mults for the single threaded:

   $$\frac{7020.86}{2183.41} = \mathbf{3.22}$$

   My guess is that the reason the speedup is less than 4 is that there is some time lost in delegating the data of the for loop to four separate threads and then recombining the results. Moreover, there is some error in measurement from executing one program after another, so we would similarly expect some deviation from the theoretical speedup of 4.

5. **Parallel Fraction:**

   $(\frac{4}{3}) * (1 - (\frac{1}{3.22})) = \mathbf{0.92}$