CS 475 Parallel Programming: Project 0 - OpenMP Experiement

Conner Rhea
rheac@oregonstate.edu

```
rheac@DESKTOP-NJ3FSKU:/mnt/c/Users/stygi/Desktop/IRL Stuff/OSU Folder/CS 475 Parallel/Project 0$ tr -d '\r' < loop.bash
> loop1.bash
rheac@DESKTOP-NJ3FSKU:/mnt/c/Users/stygi/Desktop/IRL Stuff/OSU Folder/CS 475 Parallel/Project 0$ bash loop1.bash

NUMT = 1
OpenMP version 201511 is supported here
For 1 threads, Peak Performance =   941.32 MegaMults/Sec

NUMT = 4
OpenMP version 201511 is supported here
For 4 threads, Peak Performance =  2427.18 MegaMults/Sec
rheac@DESKTOP-NJ3FSKU:/mnt/c/Users/stygi/Desktop/IRL Stuff/OSU Folder/CS 475 Parallel/Project 0$ uptime
 17:10:54 up 10 min,  0 users,  load average: 0.00, 0.00, 0.00
rheac@DESKTOP-NJ3FSKU:/mnt/c/Users/stygi/Desktop/IRL Stuff/OSU Folder/CS 475 Parallel/Project 0$
```

S = 2427.17 / 941.32 ≈ 2.58

Fp = (4. / 3.) * (1. - (1. / S)) ≈ 0.82

Written Commentary:

1. I ran this on my personal computer at home, here are my relevant specs:
   Processor AMD Ryzen 5 5600X 6-Core Processor 3.70 GHz
   Installed RAM 16.0 GB
   This is the computer I do most of my schoolwork on and play games with my friends as well

2. One Thread gave 941.32 MegaMults/sec
   Four Threads gave 2427.18 MegaMults/sec

3. My speed up value was 2.58 times faster with 4 threads over 1 thread

4. There is likely a component that cannot be done in parallel, meaning that no matter how many threads I throw at it I won't be able to speed up that component.

5. My parallel fraction value was approximately 0.82, which is fairly close to 1.0 as noted by Professor Bailey's notes