1. Drew Sadler
2. void* adder(void* args){
```
        for(int a=0;a<N;a++){
                race++;
        }
}
void* subtractor(void* args){
        for(int a=0; a<N;a++){
                race--;
        }
}
```
3. I think they would make the variable equal to 2 million, and or the negative of that.
4. I think the value of race would end up being 0
5. [asadler1@hopper3 Studio_10]$ ./race
   final value is: -338152
   [asadler1@hopper3 Studio_10]$ ./race
   final value is: 993511
   [asadler1@hopper3 Studio_10]$ ./race
   final value is: -965857
   [asadler1@hopper3 Studio_10]$ ./race
   final value is: 1014174
   [asadler1@hopper3 Studio_10]$ ./race
   final value is: 61487
   [asadler1@hopper3 Studio_10]$ ./race
   final value is: 508451
6. Random values of executing adder or subtractor, I feel the minimum value would be -2 million or the maximum value is 2 million
7. It continues with the same behavior up until somewhere around 6000 iterations, where it just starts to print out 0 for everything below that, (probably a larger number that still prints 0 just).
8. Yes, as single processors still have hyperthreading so that multiple functions can be executed
9. [asadler1@hopper3 Studio_10]$ taskset -c 0 ./race
   final value is: 0
10. [asadler1@hopper3 Studio_10]$ taskset -c 1 ./race
    final value is: -3594885
11. It does not function for 0 processor cores running since it fails being either adder or subtractor since it can't pick 1 or both. While the process that was allowed on 1 core is able to use multithreading and can have both the processes be used interchangeably finally allowing us a value other than 0.