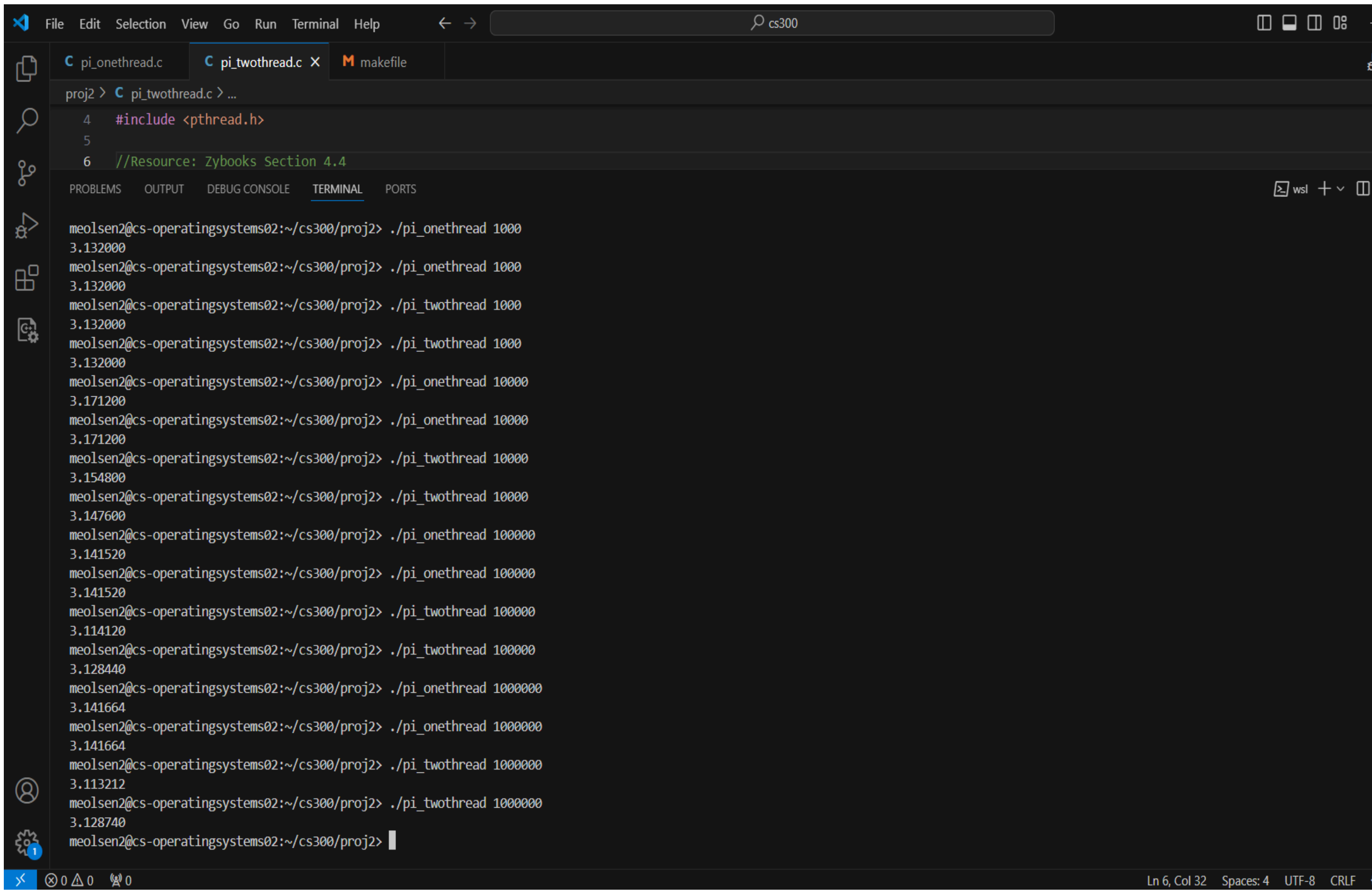
Outputs:


File Edit Selection View Go Run Terminal Help
cs300
pi_onethread.c
pi_twothread.c
makefile
proj2 > pi_twothread.c > ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
wsl
Ln 6, Col 32 Spaces: 4 UTF-8 CRLF

```c
#include <pthread.h>

//Resource: Zybooks Section 4.4
```

```
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_onethread 1000
3.132000
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_onethread 1000
3.132000
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_twothread 1000
3.132000
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_twothread 1000
3.132000
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_onethread 10000
3.171200
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_onethread 10000
3.171200
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_twothread 10000
3.154800
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_twothread 10000
3.147600
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_onethread 100000
3.141520
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_onethread 100000
3.141520
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_twothread 100000
3.114120
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_twothread 100000
3.128440
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_onethread 1000000
3.141664
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_onethread 1000000
3.141664
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_twothread 1000000
3.113212
meolsen2@cs-operatingsystems02:~/cs300/proj2> ./pi_twothread 1000000
3.128740
meolsen2@cs-operatingsystems02:~/cs300/proj2> 
```

a. Generally, the higher the total number of points, the more accurate the pi estimation is. In testing by increments of powers of 10, the most accurate results start at 100,000. From my testing, the pi estimations are all between 3.14152 and 3.14166 from 100,000 total points and beyond.

b. With using a small total number of points like 10,000, pi_onethread and pi_twothread are close in accuracy and speed, with pi_twothread actually being more accurate in some iterations. Once the total number of points reaches 100,000, the differences in performance emerge, with pi_onethread almost guaranteed to have a better estimation and faster execution time. In terms of speed, the discrepancy is caused by thread synchronization. Using pthread_join() blocks the calling thread to wait for the completion of the other thread. Since pi_twothread has more threads to join, synchronization takes longer than with one thread. In terms of estimation accuracy, synchronization also comes into play because if multiple threads are all trying to access and edit the same data, there could be inconsistencies.