Lauren Fromm
404751250

Homework 6 Brief

To create a multithreaded function, I first want to include <pthread.h> in main, then I include the -plthread in the Makefile.

I then want to create an array of threadids so I can make the threads using pthread_create.

I create a new function that holds the for loops that creates the pixel coordinates. I then use pthread_create to call this functions. Next, I make a for loop to join all the loops.

After that, I print all of the coordinates using an array that holds all of the coordinates that were created in the function called by pthread_create.

I first ran into issues printing the image, since I want to join the threads before I print them. I solved this by created a global 3 dimensional array that keeps track of the values that need to be printed, then after the threads join, I print the array.

I also ran into an issue of not being able to get to the function I made when pthread_create was called. This was because I forgot to make the function a pointer. Once I did that, my program started working.

The biggest problem I faced was my program started taking longer when I was using more threads and I couldn't understand why. Finally, I figured out that I needed to make the number of threads a global variable, and then when going through the loops

of creating the pixel coordinates,
split it up by how many threads there
were. This allowed by function
to work and I saw the results I wanted.

The results I saw were:

1 thread:
real 0m41.492s
user 0m41.485s
sys 0m0.002s

2 threads:
real 0m21.789s
user 0m43.328s
sys 0m0.002s

4 threads:
real 0m11.280s
user 0m44.621s
sys 0m0.002s

8 threads:
real 0m5.820s
user 0m44.913s
sys 0m0.002s

So obviously, using threads makes
the program go a lot faster. It takes
the original real time and divides it
by how many threads there are.