For the design of my program, I used the lecture slides as the main source of help, following its monitor solution that it provided. I also had to do additional research in understanding the use of pthreads and the different methods associated with them, such as how to initialize conditional variables and mutex locks. The basic implementation of my program was to first define some variables that would be helpful in giving a name, such as the states of each philosopher. I then created the monitor which kept track of all of the philosopher's states and made sure each philosopher was able to pick up available forks.

In my constructor, I initialized all of the philosopher states to thinking as well as the condition variables and lock which served in guaranteeing that the processes satisfy all the requirements to solve the critical section problem. The test function invoked a signal if both forks were available for a given philosopher i, and if so the philosopher would be able to begin eating. The pickup_fork and putdown_fork functions are self explanatory, however the putdown_fork also tests the neighbors to see if the forks are available for them to pick up as well.

After creating the monitor, I created a simple function that would be called by my pthread named “philosopher”. This function laid out the series of steps that each philosopher had to follow. The function would loop for the amount of time allocated, which can be changed by changing the “times” variable at the top of the program.

After implementing the monitor and interface, all that was left to do was create five pthreads for each philosopher and invoke the “philosopher” method for each of them to simulate them thinking/eating.