Python:

Dining philosopher:

import threading

import time

import random

num\_philosophers = 6

forks = [threading.Semaphore(1) for \_ in range(num\_philosophers)]

def philosopher(index):

left\_cs = forks[index]

right\_cs = forks[(index + 1) % num\_philosophers]

while True:

print(f"Philosopher {index} is thinking")

time.sleep(1)

left\_cs.acquire()

right\_cs.acquire()

print(f"Philosopher {index} is eating")

time.sleep(1)

left\_cs.release()

right\_cs.release()

if \_\_name\_\_ == "\_\_main\_\_":

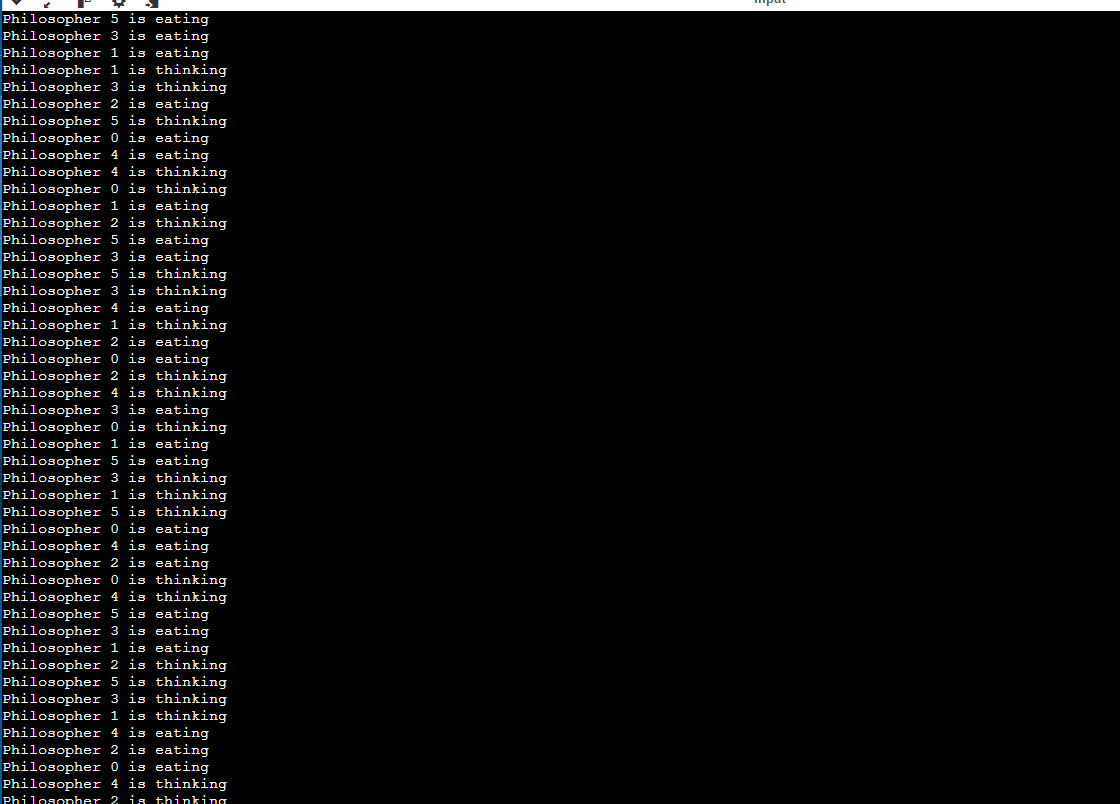
threads = [threading.Thread(target=philosopher, args=(i,)) for i in range(num\_philosophers)]

for thread in threads:

thread.start()

for thread in threads:

thread.join()

  
  
 Used mutex:  
  
import threading

import time

num\_philosophers = 2

# Create a mutex for each fork (chopstick)

forks = [threading.Lock() for \_ in range(num\_philosophers)]

def philosopher(index):

left\_cs = forks[index]

right\_cs = forks[(index + 1) % num\_philosophers]

while True:

print(f"Philosopher {index} is thinking")

time.sleep(1) # Simulate thinking

with left\_cs: # Lock the left chopstick

with right\_cs: # Lock the right chopstick

print(f"Philosopher {index} is eating")

time.sleep(1) # Simulate eating

if \_\_name\_\_ == "\_\_main\_\_":

threads = [threading.Thread(target=philosopher, args=(i,)) for i in range(num\_philosophers)]

for thread in threads:

thread.start()

for thread in threads:

thread.join()

