Name: Sakshi Sayankar

Roll No.: 67

**Token\_ring.py:**

import threading

import time

class TokenRingMutex:

def \_\_init\_\_(self, n):

self.tokens = [threading.Event() for \_ in range(n)]

self.tokens[0].set()

self.n = n

self.queue = []

def request\_critical\_section(self):

self.queue.append(threading.current\_thread().ident)

while True:

token\_idx = self.queue.index(threading.current\_thread().ident)

self.tokens[token\_idx % self.n].wait()

if token\_idx == 0:

return

def release\_critical\_section(self):

token\_idx = self.queue.index(threading.current\_thread().ident)

self.tokens[(token\_idx + 1) % self.n].set()

self.queue.remove(threading.current\_thread().ident)

def worker(mutex, id):

while True:

print(f"Worker {id} is outside the critical section")

mutex.request\_critical\_section()

print(f"Worker {id} is inside the critical section")

time.sleep(1)

mutex.release\_critical\_section()

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

mutex = TokenRingMutex(3)

workers = []

for i in range(3):

worker\_thread = threading.Thread(target=worker, args=(mutex, i))

workers.append(worker\_thread)

worker\_thread.start()

for worker\_thread in workers:

worker\_thread.join()

