CS 403 PA1 REPORT

I used lock mechanism of Python thread library to ensure syncronization. ConSet class has 4 functions; insert,pop, printSet and init. Insert and printset functions syncronized by acquiring lock at the begining and releasing the lock at the end of the function. Pop function works in a infinite while loop, if there is no element to pop it releases and acquires the lock until it can find an element to pop.

I felt that my approach may not be the best, when I have executed the leader algorithm. It felt a bit slow but eventually it works. In the algorithm I used a global lock called print_lock. It helps the print statements to be printed sequentially. After delivering the message and receiving the whole messages the process starts to find max_pair. Then checks if it is unique, if so process prints that it is unique and changes the flag to stop the process.

For the pop process I have first tried the below code. It releases the lock for 0.1 second then hopes for another thread to insert an element. However, it caused deadlock for the leader election algorithm. I did not really understand the problem. When I used “with self.lock” it worked and I did not touch it.

```python
def pop(self):
        while True:
            self.lock.acquire()
            if self.state != {}:
                for item, val in self.state.items():
                    if val:
                        self.state[item] = False
                        self.lock.release()
                        return item
            else:
                self.lock.release()
                time.sleep(0.1)
```