```java
class FilterLockMExclusion implements Lock {
        int[] level;
        int[] victim;

        int numberOfAllowedThreadsInCriticalSection;

        // We take an M, which represents how many threads we want to let
        // inside of the critical section. If you wanted the typical
        // behavior of a Filter lock, m would just be 0
        //
        // For example, a Filter(64, 2) would let 2 threads inside of the
        // critical section.
        public Filter(int n, int m) {
                // We use n -m - 1 because we want m threads in the critical
                // section
                numberOfAllowedThreadsInCriticalSection = n - m - 1;

                level = new int[numberOfAllowedThreadsInCriticalSection];
                victim = new int[n];

                for (int i = 0; i < numberOfAllowedThreadsInCriticalSection; i++) {
                        level[i] = 0;
                }
        }

        public void lock(int me) {
                // We lock all levels up until level n-m-1, which will make it so
                // a minimum of m locks can be obtained. Since only m locks can
                // be obtained, m threads will be able to enter the critical
                // section.
                for (int i = 1; i < numberOfAllowedThreadsInCriticalSection; i++) {
                        level[me] = i;
                        victim[i] = me;

                        while ((∃k != me) (level[k] >= i && victim[i] == me)) {};
                }
        }

        public void unlock(int me) {
                level[me] = 0;
        }
}
```