Proving Conditions to Critical Section Problem

If you argue that the condition is not satisfied, it is enough to give a specific execution sequence that will violate the condition.
It doesn't matter if the algorithm works 99%, if there is one situation that violates the condition then we say that the condition is violated.

For proving that a condition is satisfied you must be more general in your argument in order to cover all possible situations. A simple execution sequence will not be enough.

Mutual Exclusion:
There are two ways of proving ME.

1. Bring a process P_i in the CS. If ME is satisfied, any process P_j that attempts to enter CS will be blocked (in Peterson Solution or hardware support implementation it blocks by Busy Waiting). Show that Pj cannot exit the Busy Waiting or blocked state unless Pi exits CS and executes its exit section. This means that Pj cannot catch-up with Pi in CS, therefore Mutual Exclusion satisfied.
2. By Contradiction. Assume by contradiction that ME is violated and more than one process can be in the CS at a given time. Show that this hypothesis will result in a contradiction;

No Starvation: no process will be postponed indefinitely long (forever).
Prove that there is no possible situation by which a process will wait forever (in the entry section) while the other process uses the CS over and over.

Progress Condition:
This condition has two parts: No Delay and No Deadlock
No Delay: if CS is empty and there are processes that attempt to enter CS, one of THESE processes will be able to use the CS right away (decision on which process enters CS is done on processes attempting to enter the CS at that moment), it will not be delayed by a process that is in the remainder at that moment.
Proof: keep a process in the remainder and show that the other process can use the CS as many times it needs.

No Deadlock: decision as to which process will use the CS next is taken in finite time.

1. Prove that processes cannot block in the entry section while the CS is empty. Usually to show this you move with processes at the same pace, this creates higher probability for a deadlock.
2. Another way of proving this is by contradiction. Assume by contradiction that processes will by blocked in the entry section while the CS is empty. Proceed until you get to a contradiction.