Loop Invariant

• Def’n: A boolean condition that is true immediately before every evaluation of the loop guard.
• It is true even if the loop runs 0 times (i.e. is skipped).
• It is true immediately before each evaluation of the loop guard, including the last evaluation if the loop terminates.
• It is true immediately after the loop terminates, if the loop terminates.
while (c) {
    //@loop_invariant I;
    {
        /* loop body */
    }
    //@assert P;
}
1. **INIT**
Show that the loop invariant $I$ is true immediately before the first evaluation of the loop guard $C$. 

```
P (postcondition)
```
2. PRESERVATION
Show that if the loop invariant I is true immediately before the evaluation of the loop guard C, then I is true immediately before the next evaluation of the loop guard C.
3. EXIT
Once we have a valid loop invariant, we can show that the logical conjunction of the loop invariant I and the negation of the loop guard C implies the desired postcondition P:

\[ I \land \neg C \implies P \]
4. TERMINATION
Show that the loop will always terminate (i.e. that C must eventually be false).