Loop Invariant for Linear Search:
Consider an array $A[0 \ldots n-1]$ and a value v to search within this array. The loop invariant is as follows:

- **Initialization:** Before the loop starts, the subarray $A[0 \ldots -1]$ is empty, and thus does not contain v. This satisfies the invariant that the subarray has been searched and v has not been found.
- **Maintenance:** At the start of each iteration i of the loop, the subarray $A[0 \ldots i-1]$ has been searched, and v has not been found. During the iteration, the element $A[i]$ is checked. If $A[i]$ equals v, the search terminates successfully. If $A[i]$ does not equal v, the loop continues to the next iteration. Thus, the invariant is maintained because the subarray $A[0 \ldots i]$ is now the searched subarray without finding v.
- **Termination:** The loop terminates when $i = n$. At this point, the entire array $A[0 \ldots n-1]$ has been searched. If v is not found within the array, the algorithm concludes that v is not present. The invariant holds because the entire array has been searched without finding v.

Explanation:
The loop invariant does more than just describe the minimal states of the subarray; it provides a logical framework that explains the conditions necessary to maintain those states throughout the execution of the loop.

- **Initialization**: The invariant holds true before the loop starts. It describes the initial state and ensures that the algorithm starts from a valid state.
- **Maintenance**: The invariant must hold true before and after each iteration of the loop. This means that every time the loop progresses (from one iteration to the next), the conditions described by the invariant are preserved. This ensures the loop is making progress towards its goal without violating any key assumptions.
- **Termination**: When the loop ends, the invariant guarantees that the desired properties (like sorting or searching) have been achieved. It also often helps in proving that the loop terminates correctly, ensuring that the algorithm reaches its final, correct state.

In summary, the loop invariant is a powerful tool in algorithm design and analysis because it not only describes the minimal states of the loop but also provides the reasoning and conditions required to maintain those states and ensure the algorithm works as intended.