Loop invariant: The element searchnum is not present in list[0..... left-1] and in list[right+1, n-1].

Initialization: Before the first iteration of the loop, left=0 and right=n-1. searchnum ∉ list[0....-1] and ∉ list[n, n-1], trivially, because the ranges do not make sense.

Maintenance: Suppose, after a particular iteration of the loop, the loop invariant holds. ∴ searchnum ∉ list[0.....left-1] and searchnum ∉ list[right+1, n-1]. Now, let left<=right holds. In line 5, we compute middle=(left+right)/2. If list[middle]<searchnum, then it must be the case that if searchnum is present in list[], it will be present in list[middle+1, right]. (∵ list[] is sorted in ascending order) ∴ We initialize (left=middle+1) in line 7

Before the start of the next iteration, it's easy to see that searchnum ∉ list[0...... left-1] (Combine our maintenance assumption of prev iteration searchnum ∉ list[0..... old_left-1] and searchnum ∉ list[old_left, old_middle] which we proved above) and searchnum ∉ list[right+1, n-1] (prev assumption). ∴ Loop invariant holds in this case.

- We can similarly prove that loop invariant will hold if list[middle] > searchnum

Termination: The function will terminate in two cases:

1) If list[middle] = searchnum. So, then searchnum has been found, and we return the posn. middle
2) When the condn. left>right is reached. Just before this iteration, left<=right would hold. middle= (left+right)/2.

If, left > $\frac{left+right}{2}$, ∴ then left>right (contradiction)

∴ left <= $\frac{left+right}{2}$ Similarly, $\frac{left+right}{2}$ <= right