Finiteness of BinarySearch

1 Question

The following BinarySearch algorithm has been introduced in the class. Prove that the algorithm satisfies the *finiteness* property. That is, the while only runs for a finite number of iterations for an ordered array arr.

BinarySearch(integer array arr, integer len, integer key)
$left \leftarrow 0$, $right \leftarrow len - 1$
while $left \leq right$ **do**
  $mid \leftarrow \text{floor}((left + right)/2)$
  if $key = arr[mid]$ **then**
    return mid
  else if $key < arr[mid]$ **then**
    $right \leftarrow mid - 1$
  else
    $left \leftarrow mid + 1$
  end if
end while
return NOTFOUND

2 Answer

Claim 0: In every iteration, either $key = mid$ and the algorithm returns, or the range $right - left + 1$ strictly decreases.
Proof: If the algorithm does not return in the iteration, then either $right \leftarrow mid - 1$ or $left \leftarrow mid + 1$. Let (l_0, r_0) denote the old pair of $left$ and $right$, and (l_1, r_1) denote the new pair. Then, in the former case of $right \leftarrow mid - 1$,

$$\begin{aligned} r_1 - l_1 + 1 &= \left\lfloor \frac{l_0 + r_0}{2} \right\rfloor - 1 - l_0 + 1 \\ &\leq \frac{l_0 + r_0}{2} - l_0 \\ &= \frac{r_0 - l_0}{2} \\ &= r_0 - l_0 + 1 - (\frac{r_0 - l_0}{2} + 1) \end{aligned}$$

Because $r_0 - l_0 \geq 0$ (the condition of WHILE), the term $\frac{r_0 - l_0}{2} + 1$ is strictly positive. Thus, $r_1 - l_1 + 1$ strictly decreases.

Claim 1: The algorithm runs for a finite number of iterations.

Proof: The range $right - left + 1$ needs to be a positive integer because the WHILE loop needs $left \leq right$. Initially, $right - left + 1$ is simply len. From **Claim 0**, the range cannot remain positive after at most len iterations. Thus, the algorithm runs for a finite number of iterations.