Example: Correctness of Binary Search

Consider again the recursive version of binary search:

function RBS(a,b)
    if a<b then begin
        m := ⌊(a+b)/2⌋
        if x ≤ A[m] then return RBS(a,m) else return RBS(m+1,b)
    end
    else if x=A[a] then return a
    else return –1
end
Proof by induction of the correctness of the function $RBS$:
We show that $RBS$ correctly finds the index of $x$ in $A[a] \ldots A[b]$ by using induction on the quantity $(b-a)$.

(basis)
If $(b-a)=0$, and hence $a=b$, then the if statement is not executed and the else if and else statements correctly return $a$ if $x=A[a]=A[b]$ or $-1$ otherwise.

(inductive step)
For some $k>0$, assume that the algorithm works correctly if $(b-a)<k$, and consider the case $(b-a)=k$.
Since $k>0$, it must be that $a<b$ and hence the body of the first if statement is executed, which begins with computing the midpoint:
\[ m := \lfloor (a+b)/2 \rfloor \]
Since \( a < b \) and the computation of \( m \) rounds \( (a+b)/2 \) downward, it must be that \( a \leq m < b \), and hence both halves defined by \( m \) are smaller than the original; that is, \( (m-a) < (b-a) \) and \( (b-(m+1)) < (b-a) \).

Since \( A \) is sorted in increasing order, if \( x \leq A[m] \), it cannot be that \( x = A[i] \) for some \( m < i \leq b \), unless \( x = A[m] \) (i.e., it could be that \( A[m] = A[m+1] = ... = A[i] \)), and hence it is ok to restrict the search to \( A[a] ... A[m] \). Similarly, if \( x > A[m] \) it cannot be that \( x = A[i] \) for some \( a \leq i \leq m \), and hence it is ok to restrict the search to \( A[m+1] ... A[b] \).

Thus, the recursive calls are made correctly to problems that are at least one element smaller (that is when the recursive calls are made, the quantity \( (b-a) \) will be less than \( k \)). Hence, correctness follow by induction.